Designing Interactions to Empower Thoughtful Human-AI Co-Creation

Frederic Gmeiner

CMU-HCII-26-107
July 2026

Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee

Nikolas Martelaro (Co-Chair)
Kenneth Holstein (Co-Chair)
Aniket Kittur
Christopher McComb (Dept. of Mechanical Engineering, CMU)

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Copyright © 2026 Frederic Gmeiner

This research was sponsored by the National Science Foundation (Grant No. 2118924). The views and conclusions contained in this document are those of the author and do not necessarily reflect the views of NSF or any sponsoring agency.

Keywords: Human–AI Interaction, Generative AI, Human–AI Co-Creation, Metacognition, Cognitive Support, Interaction Design, AI-Assisted Design

Abstract

Despite Generative AI's potential to support complex, open-ended creative and analytical tasks, such as designing multi-objective mechanical parts or analyzing data, professionals often struggle to integrate these systems effectively into their workflows. Key challenges include misalignment between outputs and intentions, uncertainty about how to guide the system, and reduced cognitive engagement when tasks are overly delegated to automation, often resulting in insufficient problem exploration and limited ability to evaluate generated outcomes. I argue that to better leverage GenAI's capabilities for complex, open-ended problem-solving, we need advances in interaction and system design that deliberately support people's critical thinking and reflection, alongside continued improvements to the underlying models.

My research addresses this challenge by rethinking human-AI interaction for cognitively demanding professional workflows. I study how people use GenAI in domains such as mechanical design, slide presentation authoring, and data analysis, and develop interaction techniques that scaffold reflection, sharpen problem formulation, and support deliberate cognitive engagement where human judgment is critical. Building on cognitive and learning science principles, such as metacognition and Socratic dialogue, and design theory concepts, such as reflection-in-action, this thesis investigates how cognitively supportive interactions can improve human-AI co-creation.

In the first research thread, I examine how mechanical and architectural designers engage with novel AI-based design tools in multi-objective manufacturing tasks. From these studies, I identify support strategies, such as reflective questioning and thought externalization, that improve designers' cognitive engagement and outcomes. I further demonstrate how voice-based support agents can facilitate cognitive support in GenAI workflows. Additionally, I develop novel AI prototyping methods to more effectively evaluate real-time support agent behavior. In a second thread, I present Intent Tagging, a chat-alternative input technique that supports granular and reflective control of large language models during slide presentation authoring through lightweight, structured, and adaptive graphical interface elements. Finally, through MindKicks, I introduce a modular plug-in architecture for augmenting existing AI-assisted workflows with proactive cognitive "sidekicks." Through a study with data analysts, I show how a MindKicks-based support system can complement chat-based coding agents, increase analytical engagement with AI outputs, and help analysts discover more data issues.

In sum, this thesis contributes interaction techniques, systems, prototyping methods, and empirical insights that advance human-AI interaction by enabling people to think more deeply, create more thoughtfully, and solve more complex problems with the help of AI.

Acknowledgements

This dissertation would not have been possible without the support of many collaborators, mentors, co-authors, colleagues, friends, and family. They all deserve credit for the work presented here, and I am deeply grateful to them. My foremost thanks go to my advisors, Nikolas Martelaro and Ken Holstein. I could not have asked for better mentors. You gave me room to develop my own research direction while remaining closely involved when I needed help. You read drafts shortly before deadlines, attended practice talks, helped when participant recruitment stalled, and made conference travel and research opportunities possible. Nik, you often understood what I was trying to do before I could explain it clearly. You helped me turn early prototypes and sketched ideas into research questions that could be studied. Ken, your questions repeatedly forced me to reconsider what problem I was actually addressing and what evidence my claims required. Your detailed comments sometimes took time to work through, but the work was always stronger afterward. More than anything, both of you made me feel that I had two superheroes always on my side. Working with you has been one of the most rewarding parts of this PhD. I am equally grateful to Aniket Kittur and Christopher McComb for serving on my dissertation committee. Your guidance helped me see the broader implications of this work and pushed me to grow as a researcher.

Becoming part of the Human-Computer Interaction Institute also meant a great deal to me. The many talks, seminars, critiques, hallway conversations, and chance encounters shaped how I think about research. I am especially grateful to John Zimmerman, Lining Yao, Dominik Moritz, Raelin Musuraca, Brad Myers, Jodi Forlizzi, Laura Dabbish, Jeffrey Bigham, Paulo Carvalho, Sauvik Das, Motahhare Eslami, Sarah Fox, Scott Hudson, Geoff Kaufman, Ken Koedinger, David Lindlbauer, Adam Perer, Hirokazu Shirado, John Stamper, and Haiyi Zhu.

I was fortunate to conduct part of this research in collaboration with industry labs that I admire for their passion for interaction design. At Microsoft Research and Office AI, my special thanks go to my mentors Nathalie Riche, Nicolai Marquardt, Michael Bentley, and Vishal Chowdhary. Thank you for trusting me with an ambitious project and for creating an encouraging, generous, and collegial environment that made our work feel like a playful but equally serious exploration of future AI interfaces. I also valued the hallway, lunch, and coffee conversations at Microsoft with Michel Pahud, David Brown, Ken Hinckley, Bala Kumaravel, Gonzalo Ramos, Hugo Romat, and Steven Drucker. At Autodesk Research, I am extremely grateful to John Thompson, Ye Wang, Justin Matejka, George Fitzmaurice, Jo Vermeulen, and Qian Zhou. Thank you for welcoming me into the group, sharing your expertise, and providing spot-on feedback whenever you could.

Throughout my time at CMU, I was unusually lucky to be part of two outstanding research groups, the CoALA lab and the Augmented Design Capability Studio. Tzu-Sheng Kuo, Wesley Hanwen Deng, Alicia DeVrio, Anna Kawakami, David Chuan-En Lin, Charvi Rastogi, Alice Tang, Howard Han, Howie Wang, Chengyi Shen, and Zeda Xu, thank you for piloting

countless studies, listening to practice talks, commenting on ideas, and stepping in whenever help was needed. I often felt that I had a small team of extraordinary specialists beside me, each appearing with exactly the skill, comment, or encouragement that was needed.

The wider HCII PhD peer community mattered just as much. I am grateful to Tianying Chen, Hyunsung Cho, Lisa Egede, Yi Fei Cheng, Kimi Wenzel, Alexander Wang, Jordan Taylor, Cella Sum, Sanika Moharana, Yunzhi Li, Hao-Ping "Hank" Lee, Seyun Kim, Ji-Youn Jung, Faria Huq, Noor Hammad, Catarina Gonçalves Fidalgo, Nathan DeVrio, Ezra Awumey, Riku Arakawa, Adinawa Adjagbodjou, Duarte Sousa, Hugo Simão, Conrad Borchers, Katelyn Morrison, Will Epperson, Dinesh K. Patel, Tate Johnson, Tucker Rae-Grant, Toby Jia-Jun Li, Umut Civici, Humphrey Yang, Lea Albaugh, Pranav Khadpe, Kexin Bella Yang, Stephanie Valencia, Jaemarie Solyst, Venkatesh Sivaraman, Samantha Reig, Steven Moore, Franklin Mingzhe Li, Mary Beth Kery, Anna Fang, Frank Elavsky, and Julia Cambre. Thank you for the conversations, shared meals, feedback, and mutual support that made the PhD feel less solitary.

A special thank you goes to Nur Yildirim, my fellow designer-turned-HCI-researcher. You had already found your way through the PhD when I arrived and helped me find mine. Our weekly coffee conversations at La Prima became an informal introduction to how the HCI and CHI communities worked. You offered honest advice whenever I was unsure about a project or decision, introduced me to people at conferences, and encouraged me to review papers early in my PhD. These conversations helped me grow as a researcher, but they also became an important part of my life in Pittsburgh. Over the years, you and Sam became close friends of our family. We shared many lovely evenings together, culminating in the wonderful experience of attending your wedding with our daughter as your flower girl. Thank you for welcoming us so warmly into your lives. My thanks also go to Luke Guerdan. Our runs through Pittsburgh in every season, bike trips, dinners, drinks, and late-night conversations became a steady part of my life during the PhD. We rarely ran out of things to discuss, even when we probably should have gone home hours earlier. I am very glad that CMU brought us together.

I am equally grateful to the talented undergraduate and graduate students who supported this research as research assistants and collaborators: Anita Sun, Linda Xue, Sophia Timko, Jamie Conlin, Eric Tang, Kaitao Luo, Claire Mallela, and Anna Xu. Your care, persistence, and ideas had a huge influence on the studies in this dissertation. It was a privilege to work with you and to watch you develop as researchers. Behind all the visible PhD research also always lies administrative support that rarely appears in the final document. Leah Buffington, Queenie Kravitz, Eric Davidson, and Kate Schaich, thank you for your patience, responsiveness, and help throughout these years.

My initial decision to pursue a PhD in HCI at Carnegie Mellon was greatly shaped by Anijo Mathew, my former colleague and department head at the American University of Sharjah. Anijo, thank you for encouraging me to pursue a PhD in the United States when this idea still felt improbable and far away. Your confidence in me mattered, and I remain deeply grateful for your advice and support ever since. I am grateful as well to former colleagues and friends at

AUS, such as Kathryn Best, Lars Harmsen, Zlatan Filipovic, Robert Ziegler, Lucy Ziegler, Jason Carlow, Varkki Pallathucheril, and Martin Giesen. My time in Sharjah and the conversations and collaborations with you shaped me as a designer, educator, and researcher, and prepared me for much of what followed. Earlier, during my undergraduate and graduate studies at the Berlin University of the Arts and my time as a research assistant at Bauhaus University Weimar, I met many inspiring and talented people working where design, research, and technology meet. I especially want to thank Joachim Sauter, Kora Kimpel, Erik Spiekermann, Felix Beck, Benjamin Frenzel, Michael Markert, Clemens Winkler, and Jens Geelhaar. Our early encounters shaped my understanding of approaching technology through design, treating experimentation as a serious form of inquiry, and keeping people and their experiences at the center of technical work.

Beyond CMU, Pittsburgh became our family's home during my PhD, and we were fortunate to build lasting friendships there. I am especially grateful for the friendship of Maria Chroneos, who became godmother to both of our children and an indispensable part of our family. I am deeply thankful for the care and neighborly love we received from Robin Thurston and Ralph Vituccio. You made Squirrel Hill feel like a place where we belonged, and knowing that you were nearby gave us a sense of security and community that is difficult to leave behind.

Across all places I have lived, I have been fortunate to find friends who have remained close, regardless of distance or circumstance, over these years. Johannes Fertmann, Simon Deeg, Nils Frahm, Sepp Singwald, Lasse Teubner, Johannes Humann, Leif Gütschow, and Jason Shatilla, thank you for the countless conversations and emotional support, which gave me a steady feeling of confidence and trust. I want to thank my parents, Hayko and Waltraut Gmeiner, for supporting every path I chose to take and for encouraging my curiosity about any topic or field I was interested in exploring. I am deeply grateful as well to the Gürtürk and Subasi families, especially Nurgül and Furkan, for welcoming me into their family and supporting us across the distance throughout these years.

Finally, my deepest thanks go to my wife, Sümeyye Subasi Gmeiner, and our two children, Lina and Luis. This PhD required far more from you than appears in this dissertation. Sümeyye, you supported the decision to begin it, carried much of our family life when the work consumed my attention, and endured deadlines, travel, uncertainty, and my repeated promises that things would calm down after the next submission. During these years, we lived through a pandemic, built a home far from our families, raised our children, welcomed Luis, and moved several times. You three made it possible for me to focus on this work while ensuring that research never became more important than the lives around it. I know how much you all sacrificed and carried. This dissertation exists because of your love, patience, honesty, and strength. *Seni çok seviyorum.* Thank you for going through all of it with me and believing in our family. I am truly excited about everything ahead in our years together.

Contents

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Thesis Overview . . . 6
1.3 Contributions of This Thesis . . . 10

2 Background & Related Work **13**
2.1 Design and Creation Processes . . . 13
2.2 Cognitive Challenges of AI-Assisted Creation Workflows . . . 15
2.2.1 Overview of GenAI Approaches and Opportunities . . . 15
2.2.2 Human-GenAI Interaction Barriers in Human-AI Co-Creation . . . 16
2.3 The Need for Thoughtful Human-AI Interactions . . . 18
2.4 Strategies to Support Thoughtful Design and Problem Solving . . . 20
2.5 Summary . . . 23

3 Investigating Strategies for Supporting Designer-AI Co-Creation **25**
3.1 Overview . . . 25
3.2 Background . . . 26
3.2.1 Learning Complex Software . . . 26
3.2.2 Human-Human Collaboration as a Lens for Studying Co-Creative Systems 27
3.2.3 Team Learning . . . 28
3.3 Study 1: Think-Aloud Design Sessions . . . 29
3.3.1 Participants . . . 29
3.3.2 Study Context: AI-Based Design Tools and Tasks . . . 31
3.4 Study Design . . . 35
3.4.1 Study Procedure . . . 35
3.4.2 Analysis . . . 36
3.5 Results . . . 40
3.5.1 Design Tasks Outcome Summary . . . 40
3.5.2 Challenges Designers Faced When Learning to Co-Create with AI (RQ3.1a) . . . 42
3.5.3 Learning Strategies Among Successful Designers (RQ3.1b) . . . 45

3.6 Study 2: Learning with a Peer Guide (RQ3.2) . . . 47
3.6.1 Method . . . 47
3.6.2 Results – Guided Sessions . . . 48
3.6.3 Designers' Needs and Expectations for Co-Creating with AI-Based Design Tools (RQ3.3) . . . 50
3.7 Discussion . . . 54
3.7.1 New Learning Challenges for Human–AI Co-Creation . . . 54
3.7.2 Toward Models of Human Collaboration as Lenses for Studying and Designing Co-Creative Systems . . . 55
3.7.3 Design Opportunities and Future Work . . . 56
3.7.4 Human–AI Co-Creation Beyond Manufacturing . . . 59
3.8 Limitations . . . 60

4 Exploring Metacognitive Support Agents for Human-AI Co-Creation 61
4.1 Overview . . . 61
4.2 Background . . . 62
4.2.1 Multimodal and Collaborative CAD Systems . . . 62
4.2.2 Cognitive Challenges of AI-Assisted Mechanical Design Tasks in Fusion 360 Generative Design . . . 64
4.3 Constructing Support Agent Design Probes . . . 64
4.4 Study Design . . . 68
4.4.1 Participants . . . 69
4.4.2 Design Task and System . . . 69
4.4.3 Procedure . . . 69
4.4.4 Wizard of Oz Setup . . . 70
4.4.5 Measures and Analysis . . . 72
4.5 Findings . . . 75
4.5.1 Impact of Different Support Strategies on Design Process (RQ4.1) . . . 76
4.5.2 Perceived Benefits and Challenges of Metacognitive Support Agents (RQ4.2) . . . 84
4.6 Discussion . . . 89
4.6.1 Toward Metacognitive Design Support Systems . . . 89
4.6.2 Opportunities For Agent-based CAD Support . . . 91
4.7 Limitations . . . 92

5 Prototyping Multimodal GenAI Real-Time Agents 95
5.1 Overview . . . 95
5.2 Background . . . 97
5.2.1 Prototyping Generative AI Applications . . . 97

5.2.2 User-centered Evaluation of AI Applications . . . 99
5.3 Methods . . . 100
5.3.1 Prototyping Scenario: Designing and Implementing SocraBot . . . 101
5.4 Findings . . . 104
5.4.1 Overview . . . 105
5.4.2 How Can We Rapidly Evaluate Multimodal Prompt Strategies Using *Limited* Data in an Experiential Way? (RQ5.1) . . . 106
5.4.3 How Can We Evaluate Multimodal Prompt Strategies *With Live Session Data* in an Experiential Way? (RQ5.2) . . . 110
5.4.4 Summary of Prototyping Methods and Insights . . . 116
5.5 The Counterfactual Replay Prompt Evaluation Toolkit . . . 117
5.6 Discussion . . . 118
5.6.1 Taming Complexity with Iterative Prompt Decomposition Through Interactive Human-in-the-loop Steering and Evaluation . . . 118
5.6.2 Supporting Designerly and User-centered Prompt Engineering Approaches . . . 120
5.6.3 Future Synergies with Existing Methods and Tools . . . 121
5.6.4 Limitations . . . 122

6 Intent Tagging: Enabling Thoughtful Co-Creation with LLMs 125
6.1 Overview . . . 125
6.2 Background . . . 127
6.2.1 Intent Elicitation in Human-AI Interaction . . . 127
6.2.2 Systems for Supporting Text-based Prompting and Steering GenAI . . . 129
6.2.3 Non-Linear Content Creation and Iterative Design Workflows . . . 130
6.2.4 Systems for Supporting Slide Deck Creation . . . 131
6.3 Exploring Design Principles for GenAI-supported Rich Content Creation . . . 132
6.3.1 Design Principles . . . 132
6.4 IntentTagger: An Intent Tagging-based Slide Creation System . . . 134
6.4.1 Example Use Case Scenario . . . 135
6.4.2 Interface Features . . . 137
6.4.3 Implementation Details . . . 141
6.5 User Study . . . 141
6.5.1 Participants . . . 142
6.5.2 Procedure and Tasks . . . 142
6.5.3 Collected Data, Measures, and Analysis . . . 143
6.6 Study Findings . . . 144
6.6.1 Key differences: Chat vs Intent Tag-based Interactions (RQ6.1) . . . 144
6.6.2 Observed Content Creation Workflows Using Intent Tags (RQ6.2) . . . 147

6.6.3 Users' Perceived Benefits and Challenges for Working with Intent Tags (RQ6.3) . . . 152
6.7 Intent Tagging Interactions Beyond Slide Deck Creation . . . 158
6.8 Discussion and Design Considerations . . . 159
6.8.1 Intent Tagging Enables Non-Linear GenAI Workflows . . . 159
6.8.2 Intent Tagging Promotes Human-AI Co-Creation and Reflection-In-Action 160
6.8.3 Blending Steering Interactions and Manual Content Editing . . . 162
6.8.4 Trade-offs between UI Customization and UI Management . . . 162
6.8.5 Limitations of the Evaluation . . . 163

7 MindKicks: Pluggable Cognitive Sidekicks for Enhancing AI-Driven Workflows 165
7.1 Overview . . . 165
7.2 Background . . . 167
7.2.1 Supporting Cognitive Engagement in AI-Assisted Programming and Data Analysis . . . 167
7.2.2 Parallel Support and Multi-Agent Systems . . . 169
7.2.3 Design Tensions of Proactive Mixed-Initiative Systems . . . 169
7.3 MindKicks: A Universal Plug-In Architecture for Embedding Cognitive Support Into Existing AI-Assisted Workflows . . . 172
7.3.1 Design Rationale . . . 172
7.3.2 Design Principles . . . 173
7.3.3 MindKicks Plug-In Architecture . . . 175
7.3.4 MindKicks' Cognitive Support Strategies . . . 178
7.4 MiKiData: A MindKicks-based Data Analysis Support System . . . 178
7.4.1 Example Use Case Scenario . . . 179
7.4.2 Interface Features . . . 181
7.5 User Study . . . 186
7.5.1 Participants . . . 186
7.5.2 Task Design . . . 186
7.5.3 Dataset . . . 187
7.5.4 Tiered Finding Structure . . . 188
7.5.5 Memo Deliverable . . . 189
7.5.6 Procedure . . . 189
7.5.7 Collected Data . . . 190
7.5.8 Measures and Analysis . . . 190
7.6 Study Findings . . . 194
7.6.1 How MindKicks Expands Data Issue Discovery and Exploration (RQ1) . 194
7.6.2 How Analysts Integrate MindKicks into Their Workflow (RQ2) . . . 198

7.6.3 How MindKicks Impacts Analytical AI Engagement (RQ3) 205
7.7 Outlook: Alternative MindKicks Adapters MiKiProd and MiKiMech 210
7.8 Discussion 211
7.8.1 Supporting Cognitive Engagement and Discovery with MindKicks 212
7.8.2 Implications for Expanding Chat-Based LLM Interactions 213
7.8.3 Design Considerations for Cognitive Sidekicks and Adaptive UI 214
7.8.4 Limitations of the Study 215

8 Conclusion and Future Directions 217
8.1 Future Directions 221
8.1.1 How Can We Reimagine *Software Tools* as Cognitively Augmented Workflows? 221
8.1.2 How Can We Reimagine *Learning* With and Through AI-Augmented Work? 222
8.2 Conclusion 224

A Additional Materials – Chapter 3 225
A.1 Task Outcomes 225
A.2 Interview Protocol 226

B Additional Materials – Chapter 4 227
B.1 Study Participants Demographics 227
B.2 Wizard Training and Guides 228

C Additional Materials – Chapter 5 237
C.1 Study Participants Demographics 237
C.2 Open-Source Toolkit Details 237
C.2.1 **Architecture and Components** 239
C.2.2 Details on Hybrid-WoZ User Study Methodology 239

D Additional Materials – Chapter 6 241
D.1 Study Participants Demographics 241
D.2 Further System Implementation Details 241
D.3 Semi-structured Task Outcomes 248

E Additional Materials – Chapter 7 255
E.1 Study Participants Demographics 255
E.2 Study Materials 256
E.2.1 Case Brief and Task Prompt 256
E.2.2 Data Dictionary 257

E.2.3 Fitness Assessment Memo Template 258
E.3 Chapter 7 Coding Schemes and Measure Details 259
E.3.1 Finding Inventory Coding and LLM-Assisted Audit 260
E.3.2 Process Coding Scheme 260
E.4 Coding Agent Request Events 263
E.5 MindKicks Sessions Timelines 275
E.6 Analytical AI-Engagement Scores (AIES) 276

List of Figures

3.1 The Fusion 360 design task and workflow. The task involves (A) designing an engine bracket that connects the engine to a damper. (B) A starter file containing connection holes and bounding dimensions is provided to the users to initiate the design in (C) Fusion 360. The user is prompted to create (D) a viable design while minimizing weight. (E) The workflow involves five steps, and based on the AI system's solutions, the user may iterate the design by adjusting the design constraints and criteria to produce new solutions. 31

3.2 The SimuLearn design task. (A) The user is prompted to design a bike bottle holder using the provided (B) starter file and (C) the SimuLearn tool to create (D) morphing grid structures. (E) The transformed grid should assemble into the holder and fit around the bottle and bike frame. (F) The SimuLearn tool provides three types of design workflows. After initializing a morphing grid design, the user can choose between AI-assisted optimization functions or rapid simulations to iterate the design. 33

3.3 Overview of Study 1 think-aloud design sessions procedure. Participants were first introduced to the design tool and task, then worked while thinking aloud across multiple sessions. They completed an interview after turning in their design. 35

3.4 Overview of engine bracket designs created by participants. P-F05 met both the shape and structural requirements. P-F01 only met the shape requirements (the bracket has holes for bolts with sufficient clearance and respects the specified dimensions but is structurally too weak). All other brackets missed both requirements and were either too heavy, weak, larger than the specified dimensions, or did not have enough bolt clearance. For simplicity, I only show one design option per participant. Please see the Appendix for all submitted design options. 39

3.5 Overview of bottle holder designs created by participants. Participants came up with different designs using the shape-changing grid material in various ways. Participant P-S01 was not able to control the shape as intended and improvised by stacking grids together. While some designs (P-S06, P-S07) would not be able to hold a bottle, some designers (P-S04, P-S10) utilized the shape-changing grid structure to create promising bottle holder designs. Participant P-S11 stopped working on the task out of frustration and did not submit a design. 39

4.1 Overview of the Fusion 360 design task (A-E), workflow (F-G), common user mistakes (H), and cognitive challenges (I). Note that this task matches the "bracket design" task from the previous Chapter 3. The task involves (A) designing an engine bracket that connects the engine to a damper. (B) A starter file containing connection holes and bounding dimensions is provided to the users to initiate the design in (C) Fusion 360. The user is prompted to create (D) a viable design while minimizing weight and avoiding (E) infeasible features. (F–G) The workflow involves six steps, and based on the AI system's solutions, the user may iterate the design by adjusting the design constraints and criteria to produce new solutions. 63

4.2 Process diagram of Wizard of Oz setup. The remotely located wizard (right) followed the designer's actions (left) by listening to their verbalizations and observing their screen and webcam stream. Using a web interface, the wizard could (1) type messages and send these as (2) synthesized voice messages to the user as agent messages. (3) All agent messages were logged with timestamps. 71

4.3 Plot showing the design outcome scores between agent-supported groups and no support. 75

4.4 Overview of engine bracket designs created by participants grouped by quality score (1–5). White IDs indicate participants from the *No Support* group (all scored one point). 75

4.5 Plot illustrating the total number of messages per support topic category and agent type (unsaturated colored bars) and the percentage of observable impactful messages that triggered observable new considerations (saturated areas). 77

4.6 Timeline excerpts visualizing participant and agent interactions throughout the design task; timelines are divided into lanes, each showing (in)correct GenAI input specifications (diamond shapes) for (1) forces, (2) constraints, (3) bolt and (4) dampener clearances, (5) bounding box (from top to bottom); black and orange vertical lines represent exchanged agent and user messages with purple and black triangles indicating an observable impact on the design process. 78

5.1 The Counterfactual Replay Prompt Evaluation Toolkit. The toolkit enables iteratively testing and refining prompt strategies for multimodal real-time agent systems by replaying user session videos. (1) Prompt designers can see recorded or live user session videos and transcripts; (2) optionally refine the model instructions and (3) generate candidate LLM responses based on the prompt and replayed video context. (4) Responses can be evaluated and optionally labeled (rating and comment). (5) In live hybrid Wizard-of-Oz sessions, approved responses can be sent directly to the user's interface. This workflow supports immersive, in-context evaluation of prompt strategies for real-time multimodal support agents. 96

5.2 SocraBot's front-end user interface. (1) While a designer works on a mechanical design task in Autodesk Fusion 360 (2) SocraBot follows the user's verbalizations and screen actions and proactively sends support messages that appear in the overlay GUI window; new messages will also be automatically read as a synthesized voice message to the user. 102

5.3 Process diagram illustrating the sequence of my prototyping steps (1—5), employed evaluation approaches, and reflections, alongside revealed prompt issues and improvements. The bottom row shows the resulting evolution of my prompt strategy over the course of the prototyping process (A—C). 104

5.4 Interface of the Counterfactual Replay Prompting tool used during prototyping; using the video player (1), one can explore different situations of recorded user sessions along with the synchronized transcription (2), and then generate a model response for a specific situation (3); and review (4) and evaluate (5) the generated message. *Note: This interface differs from the open-source version.* 107

5.5 The Hybrid Wizard-of-Oz setup; (1) Wizards can follow along with user actions in the video call app; (2) using the wizard interface, they can trigger various model inference calls such as (3) task phase classification and (4) message generation; (5) wizards can optionally send generated messages to the user's SocraBot interface; (6) all generated messages can be reviewed and rated in the message history. *Note: This interface differs from the open-source version.* 111

6.1 *IntentTagger*, an intent tag-based LLM-driven slide creation system: Users can steer slide generation with (1) keyword-based concept tags by (2) adding them directly, (3) choosing from suggestions, or (4) adding existing content. (5) A user-editable outline and slide deck are generated based on all selected tags. (6) For fine-grained changes, the user can change individual slides with the same (scoped) intent tagging mechanism. Each tag allows exploring alternative properties in a dynamically generated (7) drop-down list, and tag values can be fine-tuned through the (8) opposite slider widget. 126

6.2 Screenshots of the *IntentTagger* system: (Left window) A user creates a slide presentation using the Slide Deck Steering Canvas with (1) active tags in the three circular tag groups surrounded by (2) inactive system-suggested tags. (3) The generated outline and slides are displayed in the tabbed sidebar. (Right window) The user makes (scoped) adjustments to a single slide via the Slide Steering Overlay and (4) explores generated slide variations. (5) Then, local slide changes can be accepted or applied to the entire deck. 134

6.3 Examples of steering GenAI slide generation with Intent Tags: (A) Influencing the detail level of the displayed information, (B) steering the "mood" from "bright" to "dark," or (C) gradually blending the layout between text- and image-heavy. 139

6.4 (Left) Participants' responses when rating the 6-point Likert statements for IntentTagger and Chat-based Interface, ranked from largest to smallest effects; (Right) Dots show the mean difference of IntentTagger compared to Chat-based Interface; Bars are the 95% CIs calculated with the studentized bootstrap method. 146

6.5 Timeline plots visualizing participants' interaction events of the semi-structured task (phase 3); upper lanes show events related to outline and slide generation; center lanes show interactions with concept tags; lower lanes show interactions with image reference tags; tag modifications via the *Opposite Slider Widget* are omitted since only a few participants occasionally explored slider options but without applying these to the slide generation (see section 6.6.2) *Note: During P04's session, the image suggestions failed due to an API outage. 148
6.6 UX interface sketches illustrating alternative use-cases for Intent Tagging-based steering of GenAI content creation; (Left) Blog post creator: (1) Steering of a post generation through intent tags in a side panel; (2) new tag suggestion can be requested per group; (middle) Video creator: Steering of a selected video clip through intent tags; (4) additionally, Keyframe Tags allow to control temporal aspects of the clip generation in the timeline view; (Right) 3D scene creator: (5) Steering of a 3D scene generation through intent tags. 158

7.1 Four design probes used in formative expert sessions to explore adaptive UI mechanisms for proactive cognitive support: (A) a board view for spatially organizing parallel support streams with semantic zooming/progressive disclosure, (B) a fisheye view for focus+context browsing, (C) a confidence stream for dynamically prioritizing recent items, and (D) semantic clusters for grouping related suggestions across topics. 171
7.2 Diagram of MindKicks plug-in architecture. The host application exposes the user's evolving artifact and available UI outlets. Inside the MindKicks plug-in, the domain context adapter adds task specifications, domain guidance, and task-phase information, while the tool adapter adds artifact diffs, host events, and UI outlet availability. The general support strategies remain domain-agnostic; the parallel agent runtime combines the strategy, domain context, and tool context into an LLM call and routes the generated support item back to the host UI outlets. . 175
7.3 Screenshot of MiKiData — MindKicks deployed inside Marimo. 181
7.9 Discovery outcomes by condition. Each dot is one participant; black bars mark group means. 194

7.10 Post-task Likert comparison between MindKicks and Baseline (Coding Agent only) groups. Left: 6-point Likert response distributions for each condition. Right: MindKicks–Baseline mean differences with 95% bootstrap percentile confidence intervals. Items are ordered from largest to smallest observed mean difference; reported p-values are BH-adjusted Welch two-sample tests. 195
7.11 Ratings of MindKicks-related items. Percentages show the distribution of 6-point Likert responses among MindKicks-condition participants ($n = 10$). 196
7.12 Representative MindKicks timeline examples illustrating the three usage patterns: P03 as a *High MindKicks Integration* participant, P29 as a *Low MindKicks Integration* participant, and P18 as a *Low AI Use* participant. Blue diamonds indicate MindKicks card interactions, blue circles indicate analytical reactions to MindKicks cards, light blue ticks indicate system-generated MindKicks cards/merges, orange squares indicate coding agent requests, and orange circles indicate reactions to coding agent responses. Outlined markers indicate lighter engagement events (e.g., glanced cards, surface/observation reactions, or Execute/Verify coding agent requests), while filled markers indicate deeper uptake or more analytically engaged events (e.g., Read/CodeGen/CodeAdded cards, Interpretation/Critical/Synthesis reactions, or Explore/Synthesize/Challenge coding agent requests). See Appendix Figure E.2 for all participants' session timelines. 199
7.13 MindKicks Interaction Score versus discovered data issues (discovery total) within the MindKicks condition. Points represent individual participants; the trend line shows the positive within-condition association between higher MindKicks integration and the number of discovered data issues. 201

7.14 Number of discovered data issues (discovery total) by coding agent request volume, separated by condition. Point color in the MindKicks panel indicates each participant's MindKicks integration pattern. In the baseline, the highest discovery rate occurs among participants who sent more coding agent requests than High MK Integration users, yet they were at or below the High MK users' discovery level. This suggests that MindKicks helped, especially for these High MK users, to broaden their analytical investigation and identify more data issues by combining MindKicks with the coding agent. 202
7.15 Mean discovery total by Coding Agent (CA) Usage Style and condition; error bars are ± 1 SEM. 203
7.16 Analytical AI-Engagement Score by condition. (Left) AIES-TOT (CA + MK responses) per participant; black bars mark group means. (Right) Per-participant decomposition of AIES-TOT into MK-mediated (purple) and CA-mediated (orange) components, ordered within condition by total (Baseline participants appear as CA-only bars). 205
7.17 Active MindKicks card engagement versus Combined Analytical AI Engagement (AIES-TOT) within the MindKicks condition. Point color indicates the MK Integration Pattern groups from Table 7.5. Dashed horizontal lines indicate the MindKicks and Baseline mean AIES-TOT scores. 207
7.18 MiKiProd — a MindKicks adapter for AI-assisted product planning on an infinite canvas. 209
7.19 MiKiMech — a MindKicks adapter for generative mechanical design in Fusion 360. 210

List of Tables

2.1 Contrasting views of design activity by Simon and Schön, adapted from Dorst and Dijkhuis [90]. 14

3.1 Overview of study 1 participants. P-F11* and P-S11* are the same participant who had experience in using both Fusion 360 and SimuLearn. Explanation for gaps in participant IDs: Some participants had dropped out after the first design session, or participants were assigned to Study 2 (see Section 3.6). 30
3.2 Evaluation of design outcomes for the engine bracket design task (left) and bottle holder design task (right). The designer's satisfaction with the outcome is rated with green=satisfied, yellow=neutral, red=unsatisfied. For the engine bracket task, meeting structural and shape requirement checks are rated as X=fail, check mark=pass. 38
3.3 Schematic overview of each participant's learning process of design task-relevant features of the AI system. Left: Engine Bracket Task. Right: Bottle Holder Task. Correct use of input parameters or AI tools is shown with check marks. Increasing numbers of check marks from the first iteration to the last iteration suggest participants learning to work with the AI system. 41
3.4 Overview of Study 2: Learning with a Peer Guide participants. Six additional participants (3xFusion 360, 3xSimuLearn) were paired with peer guides who had worked successfully with the AI design tools in Study 1. 47
3.5 Overview of design opportunities and example applications in relation to group cognition and team learning. 53

4.1 Overview of agent probes' support strategies and behaviors. 65
4.2 Table summarizing participants' outcome design scores across five criteria (checkmarks) and process statistics by support group. 75
4.3 Overview of design considerations and key learnings. 88

5.1 Example data from the initial prompt evaluations for comparing messages generated by human wizards during formative user testing and the GPT-generated messages in a tabular format (given the same user session context up until the timestamp). 106
5.2 Summary of participants' outcome design scores across five criteria (checkmarks) and process statistics by support group (see Appendix C.2.2 for analysis method details). Note: the *"Without SocraBot"* group is identical to the unsupported group from the study in Chapter 4. . . . 113

6.1 Outlining the trade-offs between four intent elicitation methods (GUI-based/Wizard Dialogues, Text Prompting, Chat Dialogues, and Intent Tagging); comparing the factors Open Prompt Format (allowing users to express intent freely), System Guidance (guiding users through predefined options), GUI-based Option Manipulation (adjusting single parameters via GUI elements), and Continuous Option Representation (persistent, single mutable objects). 127
6.2 Overview of semi-structured task results (phase 3); reporting the self-chosen topic of the slide deck and task process summary statistics; *Note: During P04's session, the image suggestions failed to load due to an API outage. 147

7.1 Cognitive support strategies employed by MindKicks. 177
7.2 Pre-identified finding inventory used to score participant discoveries. Tier 1 were surface-level findings (column-inferable); Tier 2 were deeper-level findings (requiring computational evidence). †Injected anomaly. . 188
7.3 MindKicks interaction codes and score weights. 191
7.4 Analytical reaction codes and inclusion in the AI-Engagement Score. . 192
7.5 Per-participant process statistics, grouped by condition and by computed usage style/pattern. MK Interaction Intensity columns report card-level engagement (Glanced / Read / CodeGen / CodeAdded). CA Req Types report the count of each request subtype (Execute / Verify / Challenge / Explore Direction / Synthesize), grouped as *Execute* and *Explore* request families. Conditional intensity shading in # CA Req and MK Interact. indicates relative value within each measure; categorical color codes link participants to the MK Integration Pattern and CA Usage Style groups discussed in §7.6.2. 197

7.6 Overview of participants' MindKicks integration patterns. 198

Chapter 1

Introduction

> *"People have got to become more effective at handling complex problems – at their daily struggle with complex and urgent issues. The survival of man seems dependent upon it. Any reasonable possibility seen by society for increasing that effectiveness should warrant serious investigation."*
>
> *— Doug Engelbart, 1968*

1.1 Motivation

As the complexity of problems facing humanity increases—from climate change to global health crises to rapidly evolving socio-technical ecosystems—computation and artificial intelligence (AI) offer tremendous potential to augment people's intellectual capabilities and enhance our capacity for complex problem-solving. Over recent years, especially with advancements in machine learning and generative AI (GenAI), there has been an unprecedented shift in how AI can support people in tackling more complex and open-ended problems and tasks than previous rule-based computational technologies. For example, in the context of writing, programming, and design, modern GenAI models can support a wide range of complex creation tasks by generating text, code, images, videos, and mechanical parts based on users' specified requirements. While historically AI has been used to primarily solve optimization-based problems, a novel aspect of GenAI lies in its ability to also support specifically those open-ended creation-related tasks that tackle more ill-defined or so-called "wicked problems"— problems and tasks for which there is typically no single correct solution [304]. In these cases, as in many design- and creation-related settings, the same problem or task is heavily shaped by individuals' goals and judgment, with the potential for different people to frame the same problem differently and for multiple solutions to be equally valid and co-exist.

While this open-endedness makes it harder to formalize and thus effectively support such tasks with traditional rule-based computational approaches [304], recent advancements in large-scale machine learning enable the development of AI systems that can more effectively support such open tasks by flexibly adapting to new and previously unseen situations and contexts [46, 317, 377].

Besides their adaptability and versatility, modern GenAI systems, such as Large Language Models, also have the capability of generating outputs that appear to reflect intermediate reasoning while interacting through natural-language dialogue with users [270, 378]. Their ability to mimic reasoning processes and engage conversationally allows users to iteratively specify intentions, evaluate outputs, and redirect the system as their work unfolds. These qualities increasingly evoke the notion of GenAI systems as "co-creators" that can take on a more active role in creative workflows than previous computerized generative software tools [80, 214, 229].

In a broader historical context, this "co-creation" shift directly connects to the long-standing HCI vision of using computation to augment human intellect [104]. A core aspect of this vision reflects a complementary division of labor in which AI speeds up and automates routine steps, while humans bring judgment, framing, and critique. However, this vision extends far beyond simply "AI as automation" by anticipating novel forms of human–AI symbiosis or collaboration in which computational systems expand people's cognitive capabilities and empower them to tackle more complex problems and larger challenges.

Today, we already see this shift starting to unfold across many design, creation, and knowledge work domains. For example, in mechanical design, working with AI can enable designers to co-create designs that would be extremely tedious or even infeasible without AI support, such as reducing the weight of an electric wheelchair component [119] or generating parts using emerging manufacturing processes, such as 3D printed shape-shifting materials [388], with 3D geometric GenAI solvers. Similarly, in programming and data analysis, working with Large Language Models (LLMs) enables developers and analysts to create code and computational workflows from natural language instructions [62], such as producing deployable large-scale mobile apps from text prompts or turning high-level questions about a data set into complex data analysis and visualization code.

However, despite the growing promise of GenAI tools to augment open-ended complex tasks such as design and programming, many professionals today experience mixed

benefits from using GenAI and often struggle to integrate it effectively into their workflows [97, 130, 202, 208, 402]. While some studies found that AI can enhance complex open-ended tasks such as drug discovery [277, 338], programming [62, 279], or data-intensive scientific analysis [97, 374], other studies support a growing concern that integrating AI into professional workflows might, at best, help people complete their tasks faster, but often actually decrease people's self-efficacy and the quality of their task outcomes [130, 202, 404].

Specifically, recent research (including my own studies in this dissertation) has identified a set of unique cognitive challenges that impede the effectiveness of current GenAI-supported workflows:

1) **Intent elicitation and alignment:** Misalignment of AI-generated content with user intentions, which can lead to a perceived disconnect between user input and the AI system's interpretation [130, 323, 351];
2) **Intent and prompt formulation:** User uncertainty around the AI system's capabilities and how to best communicate their intents to the AI system, which can lead to unproductive trial-and-error interactions [130, 402]; and
3) **Cognitive offloading and overreliance on AI outputs:** Users trust AI outputs without critical evaluation, leading to reduced self-reflection, insufficient problem exploration, and failure to notice AI errors and insufficient results [47, 130, 181, 202, 367, 404].

These cognitive and interaction challenges result in user frustration, infeasible outcomes, and a failure to help people think more effectively or solve problems better, ultimately limiting GenAI's capacity to fulfill its promise of augmenting complex human work. Within this scope, a fundamental underlying issue in current generative AI-assisted interactive workflows has been attributed to people frequently offloading their cognition to external AI supports [349] with a resulting negative impact on users' metacognitive processes [109, 289] and critical thinking [202]. Metacognition refers to a person's ability to monitor, evaluate, and regulate their own cognitive processes [115] and is essential for open-ended, creative problem-solving [28, 194, 365]. In many cases, offloading cognition to an AI system can be beneficial for routine work, for example, when using AI to complete certain well-defined repetitive tasks [303]. However, when using AI in the context of open-ended creation and problem-solving tasks, cognitive offloading hinders users' metacognition and critical thinking, resulting in lower self-efficacy and task performance [109, 202, 289]. At the same time, recent research,

including my study described in Chapter 3, has identified that AI workflows often limit people's metacognition and that metacognitive processes are essential for effectively working with generative AI systems [130, 349].

In response, a growing body of HCI work has begun to investigate strategies for improving human-AI interaction by supporting users' metacognition and critical thinking (as for example reflected in recent CHI workshop efforts around "tools for thought" [348]). Specific support approaches include metacognitive cues and Socratic question prompts designed to enhance critical thinking and mitigate cognitive offloading in AI-assisted tasks [77, 99, 107, 157, 276]. Domain-wise, most prior work has investigated mechanisms for improving human-AI interaction through interventions that force users' critical thinking in contexts such as AI-assisted reasoning [109], decision-making [47], learning [99, 107, 171], and search and sensemaking [334].

However, while promising, these tasks differ from open-ended human-AI co-creation tasks, and little work has investigated how we can better support complex, open-ended AI-assisted design and creation tasks, such as AI-assisted mechanical engineering, slide presentation authoring, or code creation. A major contribution of this dissertation is therefore to adapt and extend metacognitive support strategies to the distinct demands of such AI-assisted design and creation-focused workflows.

While many overlaps exist between creation-focused and other human-AI interaction workflows, one major distinction of design and creation-centered work is that it highly depends on *"reflection-in-action"*: the situated process through which creators iteratively make sense of emerging possibilities by acting on, interpreting, and responding to the "material at hand" in the immediate flow of their work [186, 320, 322]. For AI-assisted creation practice, this means designing new interactions and interfaces that help people iteratively critique, refine, and redirect their work as it unfolds, while keeping them engaged in the flow of their ongoing task.

Within this creation practice-focused context, this dissertation investigates strategies that foster users' critical thinking, metacognition, and reflection while also exploring how to design, prototype, and implement novel interactive AI systems that move beyond conventional chatbot interactions. While elements of these strategies have begun to appear in contemporary and widespread conversational AI systems, such as ChatGPT or Claude, they largely remain confined to chat-based interactions rather than being integrated into the broader interfaces and workflows through which complex creative work unfolds. Ultimately, through the systems developed in this dissertation, cognitive support becomes an integrated core part of AI-assisted interactive

work, leading to a more reflective and effective human-AI co-creation practice.

Consequently, my work throughout this dissertation explores how we can achieve more *"thoughtful"* human-AI co-creation by designing interaction techniques and systems to assist people in better thinking through and specifying complex problems while enabling more effective human-AI co-creation workflows. By "thoughtful" human-AI co-creation, I specifically envision AI-assisted work practices in which people are empowered to more deliberately frame complex multi-dimensional problems, explicitly specify tacit intentions, explore hidden alternatives, critically evaluate outcomes, and decide what is worth pursuing, thereby guiding generative AI systems more effectively and ultimately producing results that neither the user nor an AI system could accomplish by themselves.

In essence, grounded in extensive observation and research across AI-assisted work domains such as mechanical and product design, presentation creation, and data science, this thesis argues that

> *In complex open-ended design and creation tasks, we can achieve more effective and thoughtful use of interactive AI by deliberately supporting people's critical thinking, metacognition, and reflection-in-action through novel interfaces and interactions.*

Consequently, this thesis statement raises a sequence of research questions that progress from identifying challenges and support strategies, to designing and prototyping cognitively supportive interactions, and ultimately to integrating such support across complex AI-assisted work practices. Specifically, this dissertation is guided by five research questions:

RQ1 *What challenges do designers face when learning to co-create with computational AI tools, and what strategies can support them in learning to work with these tools?*

RQ2 *To what extent might metacognitive support, such as reflective questioning, planning, and sketching, improve designers' cognitive engagement and interactions with GenAI tools, and to what extent can such support be facilitated through interactive systems?*

RQ3 *How can we prototype and evaluate multimodal real-time AI support systems more rapidly and experientially?*

RQ4 *How might we design new interaction patterns and interfaces that make co-creation with Large Language Models more controllable, steerable, and reflective for open-ended creative workflows?*

RQ5 *How might we integrate different cognitive support strategies in parallel into existing AI-assisted workflows to promote deeper analytical engagement and enhance outcomes in complex open-ended tasks?*

Guided by the thesis statement and these research questions, this dissertation presents insights from observational user studies, novel AI prototyping methods, and hands-on system-building activities conducted within a research-backed user-centered design process. I [1] have primarily built and studied systems for human-AI co-creation in domains such as mechanical and product design for advanced manufacturing tasks, rich content creation, such as in slide presentation authoring, and AI-assisted coding for data analysis. Across these domains, I demonstrate how cognitive support strategies can be translated into novel interfaces and interactions that help people think more effectively during complex AI-assisted tasks. In doing so, this dissertation advances human-AI interaction by establishing cognitive support as a core part of interaction design, providing empirical evidence about its benefits and tradeoffs, and demonstrating how more reflective and effective forms of human-AI co-creation can be achieved through novel interfaces and interactions.

1.2 Thesis Overview

This dissertation is structured around three primary research approaches: (1) investigating key challenges and support opportunities in GenAI workflows; (2) exploring and developing new interaction techniques to improve GenAI workflows; and (3) developing new methods for researchers and practitioners to prototype and evaluate novel GenAI applications. In this section, I outline the research approaches I have taken to investigate the dissertation's research questions and provide an overview of how the thesis progresses through these investigations.

In **Chapter 2**, grounded in theories of design and creation processes, I first highlight unique aspects of open-ended creative tasks compared to more structured task domains. I then present an overview of the opportunities and challenges introduced by generative AI systems in supporting creation and design-related activities. Lastly, I argue for the importance of thoughtfulness in response to the cognitive and interactional

[1] This dissertation represents work from research projects that I have personally led, and I acknowledge the valuable contributions of many collaborators who have supported and shaped this work. While I predominantly use the pronoun "I" throughout this dissertation to maintain clarity and consistency, this choice primarily reflects my personal responsibility for the content rather than diminishing the collaborative nature of the research.

challenges identified in recent human-AI interaction and HCI literature and portray key cognitive strategies that support thoughtful design practice and problem-solving.

I initialy began my research by investigating the current key challenges of professionals working in GenAI-based creation workflows and potential support strategies to help them overcome these challenges. Specifically, in **Chapter 3**, I sought to understand (**RQ1**[2]) *What challenges do designers face when learning to co-create with computational AI tools? What are effective strategies to support designers in learning to co-create with computational AI tools?* To answer these questions, I conducted a series of empirical think-aloud studies and interviews with trained mechanical and architectural designers without prior AI co-creation experience to observe how they learn to work with AI CAD tools in the context of multi-objective manufacturing design tasks. This formative work provided preliminary evidence that designers faced many challenges working with AI assistance that prevented them from creating feasible and satisfying results—not because of the tools' interfaces but because it was a *new kind* of interaction. While most designers worked on their own, some received support during the task from experienced peer guides to observe their interactions and the guides' support strategies. This empirical approach also helped me uncover cognitive support strategies, such as prompting designers' reflection and guiding their thinking and thought externalization through sketching, which seemed to help them better think through the design task, specify the design problem more accurately, and thus work more effectively in the AI-assisted tasks. Reflecting on these findings, I discuss opportunities to improve designer-AI co-creation by integrating targeted metacognitive support into AI design tools to help designers better think through design tasks and frame problems, which present promising strategies to help them achieve more satisfying design outcomes.

After identifying key challenges in GenAI workflows and promising support strategies, I continued exploring interaction techniques to enhance designers' cognitive engagement and outcomes when working with AI tools. Specifically, in **Chapter 4**, I sought to explore (**RQ2**) *To what extent might metacognitive support, such as reflective questioning, planning, and sketching, improve designers' cognitive engagement and interactions with GenAI tools? And to what extent can we facilitate such metacognitive support through interactive systems?* To investigate this, I explored a design space of metacognitive support agents using three support strategies and observed how each influenced designers' processes and outcomes in a GenAI-based design task. I used

[2]Throughout this introduction, I use RQ1–RQ5 to refer to the dissertation-level research questions; within individual chapters, I use chapter-specific numbering for more detailed study questions.

the "Wizard of Oz" (WoZ) technique, in which a human operator controlled the agent probes. Each agent followed a different support approach: (1) *SocratAIs*, a Socratic agent that asked reflective questions to prompt self-explanation and reflection on design decisions; (2) *HephAIstus*, an agent that promoted planning and sketching activities supported by suggestions for design strategies and software operation; and (3) a third condition where experts in mechanical and generative design from Autodesk acted as wizards to provide their own interpretations of metacognitive support strategies. This study revealed that participants with metacognitive agent support often produced more feasible, higher-quality designs than those without support. Specifically, I found that prompting mental simulations through questions and supporting thought externalization through sketching helped designers with intent formulation, problem exploration, and GenAI input specification. However, I also found that asking questions alone could be less effective when designers had already formed strong, incorrect assumptions. Overall, these findings highlighted the potential of metacognitive support to improve AI-assisted design processes while also underscoring the need for adaptive and blended support strategies. I conclude this chapter by discussing how integrating adaptive metacognitive support into AI-driven design workflows can help designers navigate complex design tasks more effectively by providing timely guidance and promoting deeper cognitive engagement.

Encouraged by these insights, I built a functional prototype of *SocraBot*, a voice-based real-time interactive multimodal support system that helps designers think through GenAI-based mechanical and manufacturing design tasks by providing metacognitive support through reflective prompts and promoting planning and sketching. However, while building SocraBot atop a pre-trained multimodal Large Language Model (LLM), I realized that existing prototyping and evaluation methods were insufficient for anticipating the nuanced situational complexity and contextual immediacy required. Motivated by this gap, in **Chapter 5**, I asked (**RQ3**) *How can we prototype and evaluate multimodal real-time AI support systems more rapidly and experientially?* To address this question, I developed two new prototyping methods: *Counterfactual Replays* and an adaptation of the *Hybrid Wizard-of-Oz* approach. *Counterfactual Replays* enabled me to replay video recordings of user sessions and interactively test how different prompts influenced AI model outputs in various contexts. Similarly, *Hybrid Wizard-of-Oz* allowed real-time steering of LLM system behavior during live user sessions. By pairing these methods with empirical user studies in an iterative design process, I gained a granular and more nuanced understanding of how to refine prompting strategies to improve

SocraBot's user experience and support impact. I conclude this chapter by reflecting on how these methods offer a novel and practical approach for prototyping complex multimodal real-time AI agent behaviors, bridging the gap between automated benchmark testing and interactively exploring model behaviors within dynamic and complex real-world user scenarios.

In parallel to building SocraBot, I expanded my exploration of thoughtful co-creation interactions across additional creative open-ended task domains and input modalities. While my previous investigations focused on supporting engineering designers through voice-based agents in geometry-based generative design tools, I wanted to further explore metacognitive support interactions across other task domains and interaction modalities. Specifically, in **Chapter 6**, I asked (**RQ4**) *How might we design new interaction patterns and interfaces that make co-creation with Large Language Models more controllable, steerable, and reflective for open-ended creative workflows?* To address this question, I developed a novel input strategy called *Intent Tagging*—an interaction technique for AI-based rich content creation, such as slide presentation authoring. *Intent Tagging* actively supports users in articulating and refining their intent through granular, adaptive graphical UI elements. To further explore the feasibility of this technique, I then built *IntentTagger*, an interactive LLM-based system that supports users in iteratively generating and modifying slide decks through intent tags. In an evaluative lab study with professional PowerPoint users, *IntentTagger* demonstrated several benefits. Participants valued Intent Tagging as a more controllable and reflective alternative to chat-based LLM interactions, and appreciated graphical tags, contextual suggestions, and previews to help them articulate intent, explore alternatives, and iteratively refine slide decks. Furthermore, I also highlight potential applications of Intent Tagging for other rich content generation tasks, such as creating blog posts, videos, and 3D scenes. Condluding this chapter, I reflect on the broader implications and trade-offs of Intent Tagging for guiding diverse GenAI-enabled content-creation workflows.

Lastly, I explored how different cognitive support strategies could be generalized and integrated into already existing AI-assisted work and tools. Specifically, in **Chapter 7**, I asked (**RQ5**) *How might we integrate different cognitive support strategies in parallel into existing AI-assisted workflows to promote deeper analytical engagement and enhance outcomes in complex open-ended tasks?* To investigate this, I developed *MindKicks*, a plug-in architecture for proactive, context-aware cognitive support, and instantiated it in an AI-assisted data analysis programming environment. Building on the cognitive support strategies developed in earlier chapters, MindKicks enables the

integration of these universal strategies into domain- and user-specific work contexts and across different host-application UI affordances. A study with data analysts found that MindKicks effectively complemented chat-based coding agents, increased analytical AI engagement, and helped participants identify more data issues. Based on the findings, I highlight design considerations around proactivity, suggestion management, user agency, and coordination across MindKicks and chat-based LLM agents. Overall, MindKicks extends my prior work on individual support strategies and interaction techniques toward a universal cognitive support plug-in architecture that enables more reflective and effective AI-assisted work across domains and applications.

In **Chapter 8**, I synthesize the dissertation's central learnings and discuss how its contributions advance more cognitively supportive forms of human-AI interaction. Finally, I outline directions for reimagining software tools and learning environments as AI-supported workflows that will expand people's capacity to think, create, and better solve more complex problems in the future.

1.3 Contributions of This Thesis

This dissertation makes the following contributions:

- *A theoretical framing of thoughtful human-AI co-creation* that connects research on design cognition, metacognition, reflection-in-action, and the emerging challenges of AI-assisted creation workflows (**Chapter 2**). This framing provides the conceptual foundation for the dissertation's empirical and design investigations.
- *An empirical characterization of the challenges and support opportunities involved in learning to co-create with AI* in complex mechanical design tasks (**Chapter 3**). This work identifies cognitive and interaction challenges that limit effective AI use and documents support strategies that help designers better specify requirements and work through design problems in AI-assisted workflows.
- *A design space and empirical account of voice-based metacognitive support agents* for computational design tasks (**Chapter 4**). This work provides evidence about the benefits and tradeoffs of reflective questioning, planning, sketching, and expert-guided support.
- *Two novel prototyping methods for developing multimodal, real-time GenAI agents*: *Counterfactual Replay Prompting* and *Hybrid Wizard-of-Oz* (**Chapter 5**). These

methods support experiential testing and refinement of AI support agent behaviors using recorded and live user-session data.

- *Intent Tagging*, a novel interaction technique that uses graphical micro-prompting to support granular, controllable, and non-linear co-creation with Large Language Models (**Chapter 6**). The design and evaluation of *IntentTagger* further contribute empirical insights and design considerations for AI-assisted rich content creation.
- *MindKicks*, a modular architecture for integrating proactive, context-aware cognitive support "sidekicks" into existing AI-assisted expert workflows (**Chapter 7**). Its implementation and evaluation demonstrate how multiple support strategies can complement chat-based coding agents while revealing tradeoffs and design considerations around attention, agency, and coordination.
- *Cross-cutting interaction-design knowledge for cognitively supportive human-AI co-creation* synthesized across the dissertation's studies, systems, and methods (**Chapters 3–7**). Together, these contributions show how critical thinking, metacognition, and reflection-in-action can be supported through novel conversational, graphical, and workflow-integrated interactions, leading to a more reflective and effective human-AI co-creation practice.

Chapter 2

Background & Related Work

In this chapter, I first review theories of design and creation processes, then outline the opportunities and cognitive challenges introduced by generative AI systems in these contexts. Next, I discuss prior literature addressing the growing need to design human-AI interactions that more deliberately support thoughtful user engagement, and end by highlighting cognitive support strategies to foster thoughtful problem-solving and design thinking.

2.1 Design and Creation Processes

Design and creation processes are typically characterized by their open-ended, iterative, and reflective nature. Unlike well-structured problem-solving domains, design operates in contexts that are frequently ill-defined and resistant to prescriptive procedures [304]. Researchers in design theory and HCI have offered diverse frameworks to describe the epistemological and methodological foundations of design practice. One of the most widely discussed distinctions in design theory contrasts the paradigms of Herbert Simon [333] and Donald Schön [320], who offer fundamentally different and complementary interpretations of the nature of design. Simon frames design as a form of rational problem-solving akin to scientific inquiry and optimization within an objectively defined problem space. In contrast, Schön views design as a *"reflective conversation with the situation,"* emphasizing the designer's subjective role in co-constructing meaning through their actions and reflections. These two paradigms are grounded in two opposing ways of looking at the world, positivism and constructionism [90]. Table 2.1 summarizes key differences between these paradigms, as highlighted by Dorst and Dijkhuis [90].

Item	Simon	Schön
designer	information processor (in an objective reality)	person constructing his/her reality
design problem	ill-defined, unstructured	essentially unique
design process	a rational search process	a reflective conversation
design knowledge	knowledge of design procedures and 'scientific' laws	the artistry of design: when to apply which procedure/piece of knowledge
example/model	optimization theory, the natural sciences	art/the social sciences

Table 2.1: Contrasting views of design activity by Simon and Schön, adapted from Dorst and Dijkhuis [90].

This contrast is frequently used to distinguish between design and engineering—portraying engineering as focused on structured problems with explicit goals and constraints, and design as oriented toward ambiguity, emergence, and problem framing. However, this binary framing is overly reductive [290]. In practice, engineering tasks frequently involve navigating evolving or ill-structured problem spaces, while design work often incorporates procedural knowledge and established conventions. Despite these differences and overlaps, a distinguishing feature of all such creation and design-related processes lies in their frequent engagement with ill-defined problems that resist straightforward decomposition, often requiring abductive reasoning, iterative synthesis, and situated reflection [75, 89, 186]. Design can thus also be described as a cyclical and sensemaking process [283], where creators alternate between divergent exploration and convergent exploitation phases [26, 91]. Designers continuously reframe problems, evaluate evolving ideas, and adjust their plans in response to material and contextual feedback [187, 320]. This reflective engagement is not merely introspective but is often materially mediated: the interaction between the designer and their tools, sketches, and prototypes gives rise to new understandings and design directions [178]. Similar sensemaking cycles also appear in other open-ended knowledge work domains, such as data analysis and programming, where people iteratively form hypotheses, inspect artifacts, and revise their understanding in response to emerging evidence [97, 283]. This broader view is also essential for this thesis because AI-assisted

creation workflows often combine design-like (re)framing work with analytical evaluation and iterative specification.

Based on this understanding of how designers navigate such open-ended, materially grounded processes, the following section examines the cognitive strategies they employ to manage complexity, reflect in action, and guide problem-solving.

2.2 Cognitive Challenges of AI-Assisted Creation Workflows

In the field of *Human-Computer Interaction* (HCI), the concept of *Human-AI interaction* [12] and *AI as co-creators* [79, 80, 214, 229, 264] highlight both the opportunities and challenges presented by generative AI (GenAI) models. Opportunities lie in automating repetitive and tedious tasks, such as data transformation or formatting, and augmenting creative processes, such as code, text, image, or 3D part creation [46, 62, 294, 311], while challenges include aligning AI outputs with user intent, helping users formulate effective prompts and task specifications, and preventing users from overrelying on AI outputs without sufficient evaluation [62, 202, 349, 402]. The following sections provide an overview of existing GenAI techniques and their opportunities to support creation and design workflows, followed by their interaction challenges.

2.2.1 Overview of GenAI Approaches and Opportunities

Historically, generative AI (GenAI) systems have emerged to generate novel outputs for specialized tasks within distinct domains. For example, in design, AI tools have been developed on top of computational methods to generate 2D and 3D design options based on user-defined constraints and objectives [9, 358]. Specifically, in 3D architectural, industrial, and mechanical design, generative design tools have helped designers create CAD sketches [274], building layouts [256], 3D-printable mechanical actuators [207], optimized ship hull designs [173], and lightweight automotive and airplane components [88, 139, 253, 261]. Such AI tools employ various techniques to generate designs tailored to specific goals and requirements. Techniques include constraint-based solvers [359], style transfer [17, 125], simulation and optimization [388], shape grammars [138], genetic algorithms [253], or deep learning for 3D point clouds [140]. Increasingly, these AI-based techniques are becoming available in commercial 3D CAD

design tools, including Siemens NX, Solidworks, nTopology, and Autodesk Fusion 360 [20, 78, 262, 331].

Besides these specialized and domain-specific systems, the advancements of GenAI *foundation* models in recent years, such as large language or diffusion models [40, 307], have enabled the creation of much more versatile and task-agnostic AI models. These models are typically pre-trained on large-scale data for various downstream applications [319]. Contemporary chat-based interfaces for ML foundation models, including ChatGPT [266] and Claude [15], allow users to steer the generation of text, code, images, and reasoning-oriented agentic tasks through open-ended natural language dialogue rather than through fixed domain-specific controls. Because of this versatility, foundation models enable a single model to handle a wide range of tasks and inputs; however, they also require users to specify what context, constraints, and evaluation criteria the model should consider.

2.2.2 Human-GenAI Interaction Barriers in Human-AI Co-Creation

While recent advancements in GenAI models offer unprecedented capabilities for augmenting creative workflows, they also introduce interaction challenges that hinder effective human–AI co-creation. For example, in the context of parameter-based generative design systems, where designers specify distinct parameters to communicate their design goals to the system, the design process demands a shift in designers' attitudes, skills, and mental processes compared with traditional (CAD modeling) practices. This shift requires fluency in computational design thinking and parametric modeling, introducing new cognitive demands [241, 384]. Additionally, many GenAI systems function as black boxes—users specify parameters, then review generated outputs—making it difficult to infer how the system arrived at a particular result or to develop an actionable mental model. This opacity can undermine users' confidence and limit creative exploration.

These challenges are further amplified in the case of GenAI foundation models, such as Large Language Models (LLMs). While these models are often celebrated as major advances in creative AI, particularly for their versatility and generative capabilities, their interactional complexity introduces new challenges. As Schellaert et al. [317] argue, the properties that make foundation models powerful—*input flexibility*, *generality*, and *originality*—also make them harder to control and interpret. Input flexibility enables users to communicate their intent through open-ended natural language, but it often means fewer scaffolds to help them clarify which information the system requires.

Within this context, generality allows the same model to support many different tasks, but therefore, users must infer which capabilities are reliable in the current context. Originality allows models to produce novel outputs, yet the relationship between user inputs and model outputs often remains difficult to predict. Overall, unlike traditional design tools, which support direct manipulation and feedback, text- or prompt-based AI interfaces require users to externalize and specify their goals in open-ended ways.

At the beginning of my research journey described in this thesis, only a few empirical studies (including the study described in Chapter 3) examined these interactional and cognitive dynamics in detail. In parallel to my explorations, a growing body of work has begun to surface the core interaction challenges of GenAI-supported workflows. Based on this emerging literature and my own investigations, the following sections elaborate on the key themes that have crystallized as particularly critical for understanding the frictions and breakdowns in GenAI-assisted workflows:

1) Misalignment Between AI Outputs and User Intentions

A recurring challenge in GenAI-assisted workflows is the misalignment between AI-generated content and users' underlying intentions. This misalignment can lead to a perceived disconnect between what the user provides as input and what the system generates as output [62, 351]. Such misalignments are often exacerbated by the black-box nature of many GenAI systems, which limits users' (and sometimes also the model developers') ability to anticipate or influence how their inputs will be interpreted.

2) Prompt Formulation and Upfront Design Specification

Users frequently struggle to formulate or specify effective instructions and to understand the capabilities and limitations of GenAI systems, resulting in trial-and-error interactions and unpredictable outcomes [223, 339, 402]. In the context of design, many tools require users to articulate their design goals and constraints upfront, shifting the design process toward early, precise specification rather than the iterative exploration typically found in design and creation workflows [130, 339]. Especially in creative tasks, this can be problematic where users' needs and goals are often emergent and not fully formed at the outset [50, 321, 327, 352]. Consequently, prompt-based GenAI workflows demand a shift in designers' skills, attitudes, and mental models compared with conventional design practices, such as CAD modeling [241, 384].

3) Cognitive Offloading and Miscalibrated Engagement

GenAI systems often encourage users to "offload" cognitive effort by producing seemingly confident and fluent outputs. This can reduce critical thinking and promote overreliance on AI [127, 130, 202, 303, 349, 404]. Evidence also suggests that cognitive offloading can affect metacognitive accuracy. Users may overestimate either their own performance or the quality of AI-assisted outcomes, particularly during parts of the task where the AI contributes substantially [109, 147, 202, 225, 289, 404]. Similarly, studies of AI-assisted programming and other knowledge work find that people adopt generated suggestions more quickly but are less likely to detect errors [58, 181, 251]. This miscalibration is further reinforced by placebo effects and users' overexpectations about AI capabilities [180, 190, 367].

4) Insufficient Problem Exploration and Framing

Effective design and problem-solving work require a deep understanding of the problem space, including both explicit and implicit constraints. However, when cognitive engagement is reduced, for example, due to cognitive offloading, it becomes more difficult for users to explore alternative problem framings and iterate meaningfully on design goals [203, 402, 404]. For instance, empirical studies have shown that designers using traditional geometric modeling tools engage more frequently in semantic-level exploration and uncover unexpected solutions, whereas those using parametric systems tend to follow top-down processes with limited goal revision [52].

5) Limited Evaluation of AI-Generated Outcomes

To effectively utilize GenAI, users have to be able to critically evaluate AI-generated outcomes in relation to their evolving design goals and contextual constraints. However, if their engagement with the problem space itself is shallow, for example, due to cognitive offloading, their ability to meaningfully assess generated outcomes may be compromised [349, 404].

2.3 The Need for Thoughtful Human-AI Interactions

As outlined in the previous section, recent studies have shown that GenAI systems can impair critical thinking and encourage overreliance, particularly when users defer reasoning to the system's confident and fluent outputs. In response, a growing body of HCI

and human factors literature has called for more deliberately crafted human–AI interactions that sustain and scaffold users' cognitive engagement. Early cautions come from automation theory: Bainbridge's "ironies of automation" [25] anticipated many of these issues in the context of industrial control systems, including role shifts and operator disengagement, while more recent work revisits these risks in GenAI contexts [332]. Recent research has proposed that AI systems should provoke reflection rather than merely assist with task completion: Some argue that GenAI should actively challenge users with counter-suggestions to promote critical reasoning [314, 316], while others call for interaction designs that foster higher-order thinking rather than substituting for it [394]. These ideas are echoed in calls to make metacognitive thinking a central component of future AI literacy and training efforts [330, 349]. Recent workshop efforts in HCI on the topic of "tools for thought" extend this argument by framing GenAI systems as environments that should support and augment people's thinking processes, rather than merely produce artifacts [348].

Meanwhile, empirical studies have begun testing how interfaces and interactive interventions can promote deeper engagement, including my own work detailed in this thesis. In the context of AI-assisted decision-making, Bucinca et al. found that requiring users to justify AI-assisted decisions reduces overreliance, albeit at a cost to user satisfaction [47], and Gajos et al. showed that removing direct recommendations leads to greater learning from AI explanations [123]. In programming and data analysis, Kazemitabaar et al. demonstrated that task decomposition and stepwise user prompting can improve control, reflection, and learning outcomes [171, 172]. Gordon et al. extended a similar approach to spreadsheet work, where co-audit tools help users verify AI-generated results [134]. Other recent systems explore metacognitive cues, Socratic prompts, critical-thinking interventions, and AI-generated critiques in search, learning, and data analysis workflows [96, 97, 99, 107, 157, 334]. Together, this work suggests that cognitive support is increasingly important for the interaction design of AI-assisted work.

I began investigating these questions in 2021, prior to the widespread availability of GenAI foundation models, when few empirical studies had examined human–AI interaction in open-ended, creative domains. Through these early empirical studies and prototyping with designers (see Chapter 3 and 4), I identified challenges and opportunities for supporting GenAI workflows through reflection, intent formulation, and deeper engagement—grounded in (meta)cognitive support strategies. My work contributes to this growing body of research by adapting and extending metacognitive

support strategies to the distinct demands of GenAI-assisted design and creation workflows. Throughout this thesis, I use the term "thoughtful" to describe interactions that scaffold critical thinking, reflection, and metacognitive awareness—supporting users in thinking more deeply about specific facets of a problem and broader issues across alternative framings or constraints.

The following section outlines key cognitive strategies that support thoughtful problem-solving and design—such as self-explanation, thought externalization, and task decomposition—highlighting how these approaches can foster metacognitive engagement and enable reflective, problem-oriented, and exploratory thinking in design contexts.

2.4 Strategies to Support Thoughtful Design and Problem Solving

Effective problem-solving and design often hinge not only on domain knowledge but also on the ability to strategically manage one's own thinking processes. Research across cognitive science, educational psychology, and design studies has highlighted the importance of (meta)cognitive processes—variously referred to as metacognition, self-regulated learning, critical thinking or reflection[1]—in enabling individuals to decompose complex tasks, monitor progress, and adapt strategies in response to evolving goals or constraints. *Metacognition* [115] refers to the awareness and regulation of one's own thinking processes, including knowledge about when and how to use particular cognitive strategies. In the cognitive and learning sciences, metacognitive support has been shown to play a crucial role in enhancing problem-solving abilities by enabling individuals to reflect upon and actively regulate their own cognitive processes [141, 194]. *Self-regulated learning (SRL)* is closely tied to metacognition and involves understanding and supporting learners' ability to manage and direct their learning processes through metacognitive skills like planning, monitoring, and evaluating their actions, typically occurring in distinct cyclical phases [29, 149, 273, 365].

Design and HCI research also increasingly emphasizes the epistemic complexity of design activities while highlighting the central role of metacognitive monitoring and control processes for design activities. For example, Ball and Christensen [28,

[1]For example, John Dewey is often credited with originating the modern concept of critical thinking and initially framed it as reflective thinking, emphasizing the active, persistent, and careful consideration of belief and evidence [85].

p. 49] explicitly draw parallels between metacognitive processes and prior design theories, such as Schön's notion of the role of *"reflection in and on action"* in design practice [93, 320, 322]. Similarly, Price and Lloyd [290] argue that addressing ill-defined design problems necessitates not only domain knowledge but also meta-awareness of one's biases and cognitive strategies. Empirically, studies on design processes have found evidence for the importance of metacognition for learning and mastering critical design-related skills [169, 196, 284, 287].

Based on this theoretical foundation, prior research has identified various strategies to support cognitive processes to enhance problem-solving and designerly thinking, such as self-explanation, thought externalization, and task decomposition.

Self-explanation and Reflective Questioning

Self-explanation is a common strategy for prompting individuals to articulate their reasoning and underlying assumptions to themselves, which supports them in clarifying and organizing their own understanding [364]. This process, often in combination with think-aloud-style verbalizations of thoughts, promotes the integration of new information with prior knowledge, fostering critical thinking and cognitive engagement in a task [141, 146].

A common technique to prompt self-explanation and reflection is by **asking questions** to foster deeper cognitive engagement during problem-solving and design activities. In educational contexts, deep-level reasoning questions (e.g., questions probing underlying principles or causal relationships) or inquiry-based prompts (e.g., prompts encouraging student-led questioning and investigation) have been shown to enhance learning outcomes by stimulating critical thinking and deeper exploration of complex concepts [34, 74, 86, 135, 350].

Here, a specific strategy is the *Socratic Method*, which employs guided open-ended questioning to stimulate critical thinking and reflection [102]. This approach has been used effectively across various fields, such as in healthcare education to develop critical thinking skills among students [151], programming to aid novice debuggers in identifying and resolving code issues [8, 11, 182, 381], supporting academic career development [276], and in creativity research to foster co-creativity between humans [337]. But also in design practice and education, reflective questioning plays a central role in guiding designers' thinking in exploring and refining ideas by challenging their assumptions as they work through complex, evolving problems [106].

Research analyzing design team communication has shown the crucial role of question asking for problem-solving [19], idea generation [69], design reviews [56], and design studio education [158]. Other work has explored ways of supporting designers through targeted questioning to enhance the design process by helping them articulate product requirements [375], stimulate idea generation [309], and highlight awareness of bias in designerly thinking [290]. Ozgur Eris developed a taxonomy of questions asked during design teamwork [106], building on prior taxonomies by Lehnert [205] and Graesser [135]. His taxonomy outlines three classes of questions for design: (1) *Low-level questions* for clarification, (2) *Deep reasoning questions* for causal explanations, and (3) *Generative design questions* for exploring alternative solutions. Research showed that student teams asking more Deep Reasoning and Generative Design questions achieved more innovative design outcomes [106].

Thought Externalization

Verbalizing one's thoughts is one form of **thought externalization**, where internal cognitive processes are made explicit through outward expression. In addition to verbalization, other forms include sketches, diagrams, or written notes that similarly serve to externalize thinking and support reflection [248, 357]. In design and problem-solving contexts, externalization plays a central role in enabling reflection, offloading working memory, and generating novel insights through interaction with material representations [87, 248]. Prior HCI and design research has highlighted how tools and interfaces that support externalization can facilitate ideation, communication, and metacognitive engagement in individual and collaborative design work [45, 255, 305, 362].

Task Decomposition

Another effective cognitive strategy is **task decomposition**, which involves breaking down complex problems into smaller, more manageable sub-problems or subtasks [257]. By reducing cognitive load, this approach enables individuals to allocate attention and resources more effectively, often resulting in more structured and successful problem-solving [72]. In design-related activities, task decomposition supports iterative structuring and reframing of goals and constraints, facilitating both problem understanding and solution generation [23, 175, 188, 195].

2.5 Summary

This chapter reviewed foundational theories of design and creation and examined how generative AI technologies reshape these processes—introducing new affordances and cognitive challenges. In response to these challenges, we surveyed literature advocating for human-AI interactions that actively scaffold more reflective and deliberate forms of engagement. We then turned to cognitive support strategies to foster thoughtful design practices. In particular, we focused on strategies such as self-explanation, thought externalization, and task decomposition, which have been shown to enhance metacognitive engagement and support reflective, problem-solving-oriented approaches to design. In the next chapter, I describe how I empirically identified several cognitive support strategies to help professional designers overcome challenges when working in GenAI-based design workflows.

Chapter 3

Investigating Strategies for Supporting Designer-AI Co-Creation

---

This chapter was adapted from my published paper:

> Frederic Gmeiner, Humphrey Yang, Lining Yao, Kenneth Holstein, and Nikolas Martelaro. 2023. "Exploring Challenges and Opportunities to Support Designers in Learning to Co-create with AI-based Manufacturing Design Tools." *In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany. ACM, New York, NY, USA.*

3.1 Overview

As outlined in the previous chapter, while GenAI holds significant promise for augmenting design and other creative processes, it also poses key challenges that often limit its usefulness to people. At the time when I conducted this study, few empirical investigations had explored how professionals in open-ended task domains (such as design) engage with GenAI systems in practice. I was particularly interested in exploring how professionals actually (try to) learn to co-create with AI tools, especially in complex domains like engineering and industrial design. To investigate these issues, I conducted a series of studies observing how trained mechanical and architectural designers without prior GenAI experience approached manufacturing design tasks using two computational AI tools. Drawing on theories of human-human collaboration and team learning [30, 183, 379], I examined how designers interacted with these systems through the lens of collaboration. I asked:

RQ3.1a *What challenges do designers face when learning to co-create with computational AI tools?*

RQ3.1b *How do designers overcome these challenges?*

Findings from this first study revealed that designers valued AI assistance but faced substantial challenges in understanding the tools' capabilities and outputs. Those who were able to produce feasible and satisfying designs learned to co-create with the tool by systematically testing the boundaries of its capabilities early on, by self-explaining AI behaviors they observed, and by sketching and reflecting on design issues.

These insights led to a second study, which asked:

RQ3.2 *What are effective strategies to support designers in learning to co-create with computational AI tools?*

I observed how experienced users guided newcomers, revealing support strategies such as prompting reflection, suggesting alternative approaches, and using multimodal communication, such as screen annotations and mouse gestures.

Finally, to inform future tool design, I asked:

RQ3.3 *What are designers' needs and expectations for human–AI co-creation?*

Across both studies, participants expressed a need for more conversational, context-aware tools that better support exploratory workflows and enable richer communication. This chapter details the two studies and synthesizes their findings to derive design opportunities for developing more supportive and adaptive AI-assisted design tools.

3.2 Background

3.2.1 Learning Complex Software

Prior HCI research has looked to evaluate and improve the learnability of complex software systems. Past studies explore how people of different ages learn a feature-rich notetaking tool [224], how professional engineers learn 3D design software [176], or how casual designers learn professional motion graphics software [164]. Often, people learn by searching web forums or asking knowledgeable colleagues for help [176]. Research on interfaces to support people in learning complex software has proposed dynamic feedforward tool tips [198], guided tutorial systems [110], and widgets that support self-directed trial and error learning [234].

While prior work has explored lenses such as self-directed learning for working with complex software [61], it remains an open question of how best to support self-directed learning for co-creation with AI systems that take an active role in the design process. For example, prior studies in Human-AI collaboration show that the black box nature of AI systems introduces new challenges where users grapple with non-transparent and non-intuitive system behavior, hindering coordination and communication when completing "collaborative" tasks [54]. To address these issues, various strategies like explainable AI or intelligibility are aimed at helping users refine their mental models of AI systems [246, 343].

However, even as AI models become more intuitive for users, we might expect that there will always remain a need for learning to work effectively with AI systems to, for example, develop *shared mental models* [168, 183]. Consequently, as I discuss in the next section, supporting humans in learning to effectively co-create with AI requires bringing in additional theoretical lenses.

3.2.2 Human-Human Collaboration as a Lens for Studying Co-Creative Systems

To design effective human–AI collaboration, researchers have suggested drawing lessons from studying what makes human–human collaboration effective [30, 31, 54, 152, 183, 371, 376, 406]. While it remains an open question to what extent scaffolds for human-AI collaboration should mirror the designs of supports for human–human collaboration [371, 406], human-AI interaction researchers suggest that theories and findings from psychology, education, and the learning sciences are currently underutilized. For instance, Koch and Oulasvirta [183] note that *group cognition*—the study of how agents relate to other agents' decisions, abilities, beliefs, common goals, and understandings—provides powerful concepts for work on human–AI collaboration, yet is rarely referenced within this literature. Group cognition comprises phenomena such as *grounding in communication* [68] (creating mutual sense through verbal and non-verbal communication) and *theory of mind* [103] (the ability of agents to be aware of their own and the other's beliefs, intentions, knowledge, or perspectives). Similarly, Kaur et al. [168] argue that like human-human collaboration, effective collaborations between humans and AI may require *shared mental models* between people and the AI to enable mechanisms such as adaptive coordination of actions among team members [68, 249]. These may include shared representations of the task to be accomplished, of each other's abilities and limitations, or of each other's goals and strategies [82, 114, 318, 360]. A line

of work addressing these opportunities has begun to explore how humans might be supported in developing and maintaining more accurate mental models of an AI collaborator's capabilities and limitations [30, 31, 184]. However, compared to concepts of human-human collaboration, honing only users' mental models is not sufficient enough for effective collaboration, which requires *shared mental models* between the user and system [168].

To date, little work has explored how best to support humans in learning to collaborate with AI on authentic tasks, such as design tasks, despite growing recognition of the need for such supports [54, 199, 230, 406]. Design tasks represent compelling challenges for human–AI collaboration, given that design problems are often ill-defined and require teams to navigate and negotiate both the problem and solution space [89] through an iterative process of generating ideas, building prototypes, and testing [159].

In this study, I investigate human-AI collaboration for emerging manufacturing design tasks—an area where successful task performance sometimes requires human–AI collaboration, yet where effective collaboration may be challenging to achieve without strong supports [199, 406].

3.2.3 Team Learning

While phenomena such as *grounding in communication, theory of mind* and *shared mental models* provide useful concepts to explain which cognitive and social phenomena enable collaboration among a group of agents [168, 183], these theories do not explain how groups of individual agents *learn* to effectively collaborate. To address this gap, *team learning* emerged to study what actions and conditions contribute to how human groups learn to effectively collaborate together [379]. For example, team learning studies suggest that the development of effective shared mental models is supported through an *active process of negotiation* between team members, involving *"constructive" forms of conflict*, *argumentation*, and *resolution* [148, 165, 360]. However, to date, team learning has been under-utilized as a lens for studying human-AI co-creation. In this work, I draw upon concepts from team learning, such as *active processes of communication, joint information processing*, and *coordination of actions* [240], to study what actions and support strategies can help designers learn to co-create with AI-based design tools.

3.3 Study 1: Think-Aloud Design Sessions

To investigate the current key challenges of professionals working in GenAI-based creation workflows, I conducted a series of think-aloud studies [363] with trained designers new to working with AI, where they worked with an AI design tool to complete a realistic advanced manufacturing design challenge. Think-aloud studies have people verbalize their thoughts while performing a task so that researchers can understand their cognitive processes (e.g., forming mental models [63], learning [398]). In my study, I use the think-aloud method to see how AI novices, who encounter a real learning challenge and are less biased than experts, learn to co-create with the AI tools. Participants first completed a 30-minute moderated think-aloud session where a member of the research team observed them working and listened to what they said they were thinking and doing while working. Half of the participants had a mechanical engineering background and designed a light and strong mounting bracket for a ship engine while considering the optimal manufacturing method and material combination using Autodesk's Fusion 360 Generative Design (based on topology optimization that generates multiple options) [235]. The other half of the participants with a background in architecture or industrial design designed a bike bottle holder made with shape-changing materials—a complex design task that is challenging to complete without computational support. Designers working on the bottle holder task worked with SimuLearn [388], a machine learning-based research tool built on top of Rhino3d that helps designers create structures from shape-changing materials.

After completing the design task, participants submitted their designs and joined a semi-structured interview to reflect on their experience of working with the design tools. Across the study, I collect the following data:

- Video and audio recordings and machine-generated transcripts of the open-ended think-aloud design sessions
- Audio recordings and machine-generated transcripts of the post-task interviews
- 3D designs created during the think-aloud sessions

3.3.1 Participants

I recruited 14 designers (4 female / 10 male, aged 19 to 64 *(M = 32.5, SD = 16.6))* with backgrounds in Architecture, Industrial Design, or Mechanical Engineering (Table 3.1).

TABLE 3.1: Overview of study 1 participants. P-F11* and P-S11* are the same participant who had experience in using both Fusion 360 and SimuLearn. Explanation for gaps in participant IDs: Some participants had dropped out after the first design session, or participants were assigned to Study 2 (see Section 3.6).

ID	Group	Age Years	Gender	Domain	Occupation	CAD Exp. Years	Indus. Exp. Years
P-F01	F360	27	M	CivilE	Student,MA	>5	2 – 5
P-F02	F360	27	M	MechE	Student,PhD	>5	2 – 5
P-F03	F360	25	M	MechE	Student,PhD	>5	2 – 5
P-F04	F360	26	M	MechE	Student,MA	>5	1 – 2
P-F05	F360	19	M	Arch	Student,BA	2 – 4	0
P-F11*	F360	64	M	MechE	Contractor	>10	>30
P-F12	F360	59	M	MechE	Designer	>10	>30
P-S01	SL	21	F	Arch	Student,BA	2 – 4	0
P-S03	SL	23	F	CompDes	Student,MA	2 – 4	0
P-S04	SL	21	M	Arch	Student,BA	>5	1 – 2
P-S06	SL	23	F	Arch	Student,MA	>5	<1
P-S07	SL	23	M	Arch	Student,MA	>5	0
P-S10	SL	33	F	IndusDes	Researcher	>5	6 – 10
P-S11*	SL	64	M	MechE	Contractor	>10	>30

Most participants were recruited from our institution's student body, but I also recruited three professional designers via an online forum for designers who work with Fusion 360 [298]. Participants had a minimum of two years of experience using CAD (Fusion 360 or Rhino 3D) but no experience working with the studied AI design tools, determined via a screening questionnaire. Participants included mostly undergraduate and Ph.D. students and three engineers with > 30 years of industry experience. I recruited participants familiar with either Fusion 360 or Rhino3d so that they could focus on learning to work with the AI co-creation features rather than on learning the CAD tool's user interface. Before the study, all participants signed a consent form approved by our institution's IRB *(STUDY2021_00000202)*. Participants were paid 20 USD per hour.

3.3.2 Study Context: AI-Based Design Tools and Tasks

To gather generalizable insights into designers' challenges, needs, and expectations around designer-AI co-creation, I observed how designers tried to learn to co-create with *two different computational AI tools* for digital manufacturing tasks. Participants were given a non-trivial, realistic design task to work on during the study. I selected a mechanical engineering design task concerning multi-dimensional optimization and an industrial design task exploring the use case of shape-changing materials. Both tasks are too complex to accomplish without AI and also have functional AI tools already developed. I collaborated with engineers and advanced manufacturing experts to identify and pilot the tasks to ensure they were adequately complex but not overwhelming for the target population. The tasks required participants to generate design solutions within a few hours over multiple sessions.

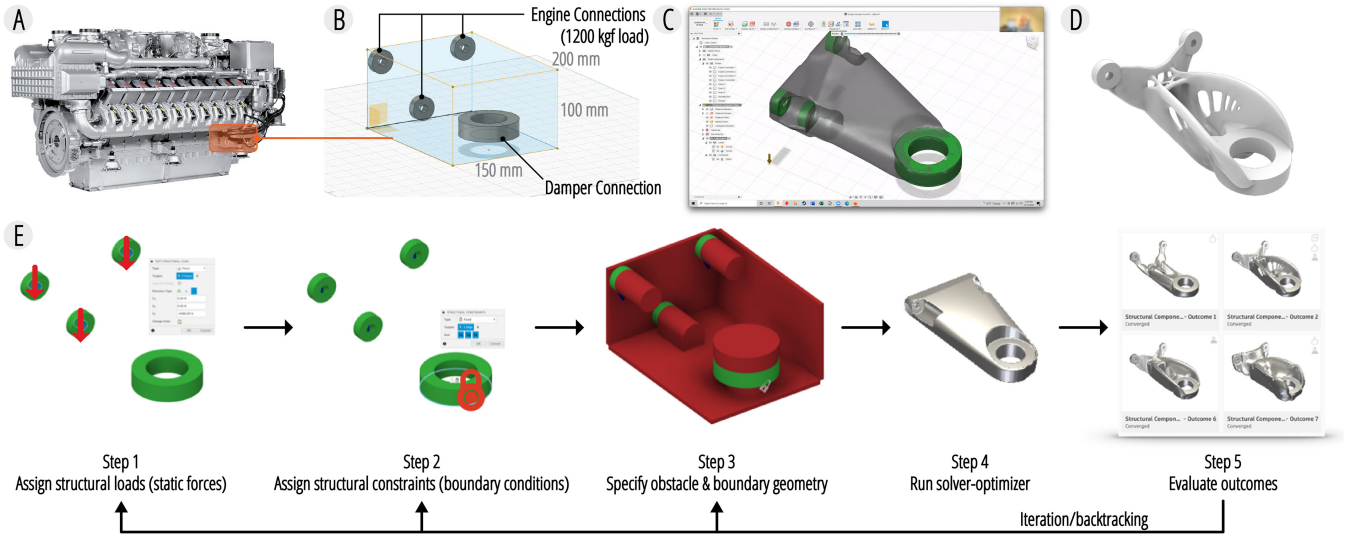


FIGURE 3.1: The Fusion 360 design task and workflow. The task involves (A) designing an engine bracket that connects the engine to a damper. (B) A starter file containing connection holes and bounding dimensions is provided to the users to initiate the design in (C) Fusion 360. The user is prompted to create (D) a viable design while minimizing weight. (E) The workflow involves five steps, and based on the AI system's solutions, the user may iterate the design by adjusting the design constraints and criteria to produce new solutions.

Mechanical design support tool

Mechanical designers worked with the "Generative Design" feature of Autodesk Fusion 360 [20], which helps designers to create lightweight and strong parts through

topology optimization and genetic algorithms. In this task (Figure 3.1A-D), the designer is asked to design a material-efficient and structurally-sound engine mounting bracket by considering the optimal manufacturing and material combination from a large pool of possibilities. While designing mounting brackets is common for mechanical engineers, optimizing designs for different manufacturing methods and materials is difficult without simulation and AI support. Traditionally, engineers would first build a part and then gradually remove or add material based on structural analysis to derive a weight-optimized part. Exploring different manufacturing options would be necessary for every material and manufacturing constellation—which is time-consuming and tedious. In contrast, Generative Design can automatically generate many different design options based on specified high-level requirements, which the designer can explore and choose from.

Participants were provided a starter file containing the geometric constraints and then had to specify the *structural loads* a part has to hold and its *structural constraints*, the GenAI solver's *obstacle geometry* (part areas that must remain free of material, such as clearances for bolt holes) (Figure 3.1 E). Participants then optionally requested a preview simulation before running the solver and then evaluated the AI-generated solutions to identify viable designs. If none of the outcomes were deemed satisfactory, the user might choose to iterate the design by adjusting the input criteria.

Industrial design support tool

Industrial and architectural designers worked with SimuLearn [388], a research system built on Rhino3D that uses ML-driven simulation and optimization to enable designers to rapidly create objects out of shape-changing materials. This manufacturing process creates 3D-printed flat grids out of PLA plastic that can transform into a volumetric shape when heated. The transformation and the resulting shape can be controlled by tuning the grid geometry and the portion of the active transformation element (i.e., actuator ratio) within the beams. This technique is envisioned to reduce production waste and shipping costs (e.g., flat packaging, reduced support material).

While new materials are being developed to manufacture shape-changing structures, designing shape-changing components poses unique challenges that designers are ill-equipped to handle. Unlike 2D and 3D design, designing with shape-changing materials involves a non-intuitive mismatch between the final target (3D shapes) and the design input (often 2D). Effectively designing for shape-changing materials requires

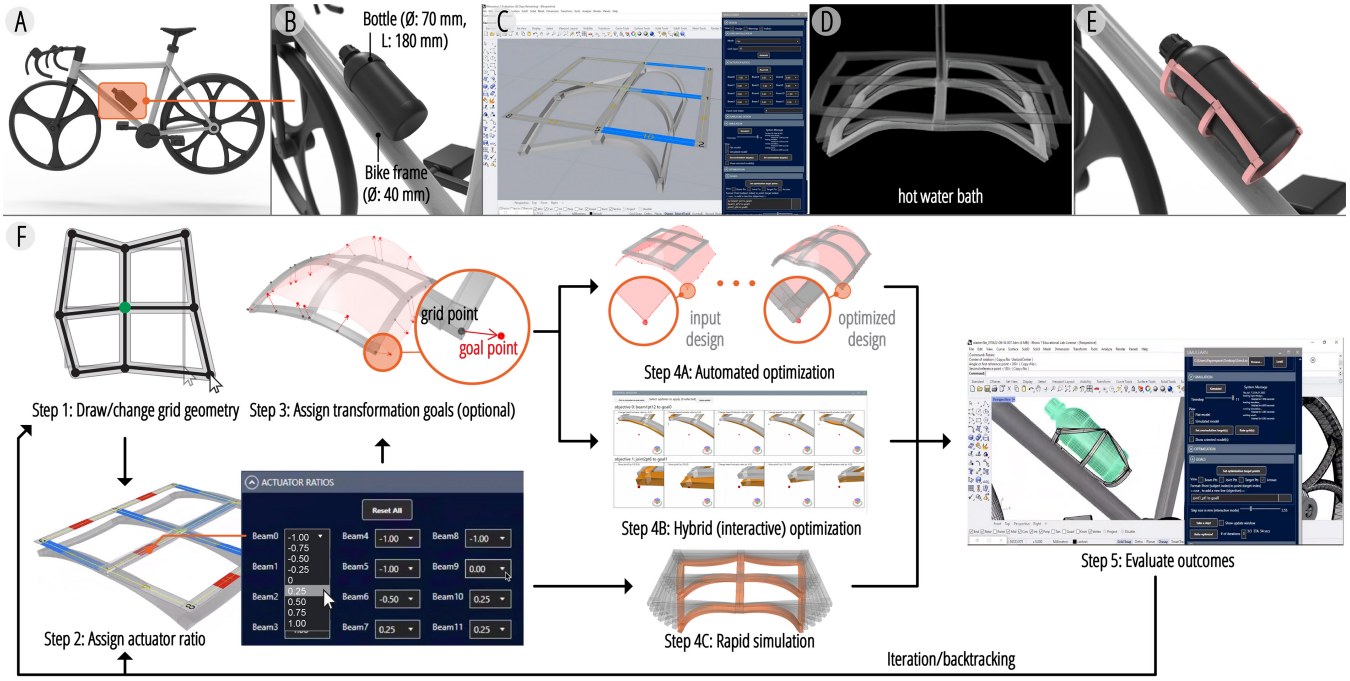


FIGURE 3.2: The SimuLearn design task. (A) The user is prompted to design a bike bottle holder using the provided (B) starter file and (C) the SimuLearn tool to create (D) morphing grid structures. (E) The transformed grid should assemble into the holder and fit around the bottle and bike frame. (F) The SimuLearn tool provides three types of design workflows. After initializing a morphing grid design, the user can choose between AI-assisted optimization functions or rapid simulations to iterate the design.

an understanding of (often complex) spatiotemporal, self-assembling material behaviors that may push against the limits of what humans can mentally simulate. At the low level, designing such materials requires modifying volumes voxel-by-voxel, which is infeasible for complex structures if done manually. AI-driven tools allow designers to create complex artifacts that would otherwise be impossible to create by hand. SimuLearn aids the design process by providing real-time simulation and optimization to iterate designs toward the desired morphing behavior.

In this task (Figure 3.2A-E), participants design a bike bottle holder using morphing grids. A starter file containing the bottle and bike frame geometry was provided to contextualize the design. This task was more open-ended than the mechanical engineering task since each designer may assemble the morphed grids in different ways to create the holder. To use the tool (Figure 3.2F), the designer models the grid geometry and assigns bending actuator ratios. Next, participants simulate the design, observe the predicted transformation, and iterate the design by changing the grid model and actuator assignment. Alternatively, participants may opt to use functions to optimize

the grids toward a targeted transformed shape. The optimization process can be either autonomous or interactive (i.e., the tool suggests edits for the user to choose from). To effectively work with the tool, users need to learn to work with the different levels of AI assistance to produce a satisfactory design iteratively.

Generalizablity of chosen design tools

I specifically study these systems for two reasons. First, while both tools support advanced manufacturing tasks, they represent computational systems with distinct purposes and interaction paradigms. Fusion 360's Generative Design module assists engineering designers with the generally familiar task of creating light and structural solid parts. The AI system helps designers to navigate a large design space and explore opportunities while adhering to specified requirements and constraints. SimuLearn, on the other hand, supports designers in working with an emerging material and manufacturing process unfamiliar to most designers. SimuLearn's AI tool provides rapid simulations of the shape-changing material and offers different levels of design assistance—from manual, over interactive, to autonomous optimization/iteration.

Second, each tool represents a different interaction style and synchronicity. In Fusion 360, users follow a structured sequence of steps to set up the parameters and acquire solutions. It may take a few hours to generate new solutions, and the designer may export the generated models at any time or iterate the design by adjusting the parameters and rerunning the solver. By contrast, SimuLearn's solver runs two to three magnitudes faster (5-180 seconds), and users interact with the system without a predefined workflow. Participants may also freely switch between the three levels of AI support at any point.

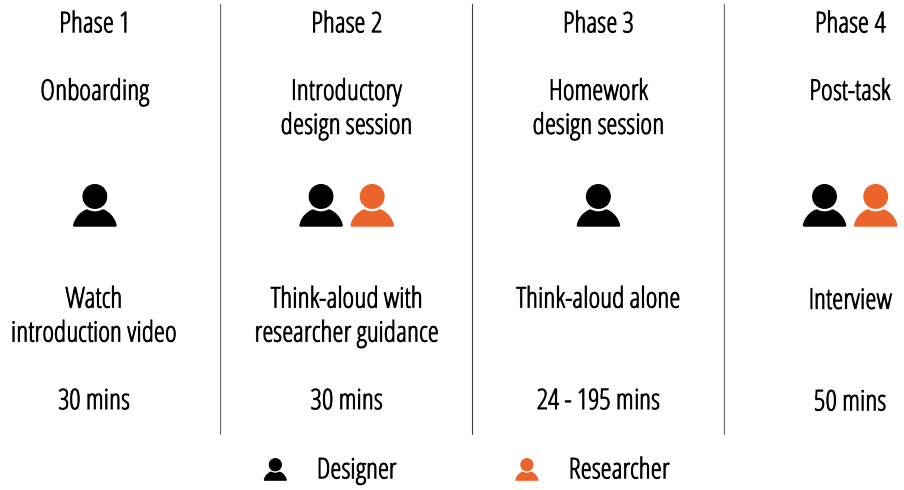


FIGURE 3.3: Overview of Study 1 think-aloud design sessions procedure. Participants were first introduced to the design tool and task, then worked while thinking aloud across multiple sessions. They completed an interview after turning in their design.

3.4 Study Design

3.4.1 Study Procedure

The study was structured into four phases (see Figure 3.3):

1) On-Boarding: Before the first session, participants received instructions by email on how to access the design tools running on a dedicated remote machine. They also watched a 25-minute video tutorial demonstrating the tools' core functionalities with a step-by-step example.

2) Intro Design Session: At the beginning of the first session, a research team member introduced the participants to the study, design brief, task, and starter file. Sessions were conducted over video conference (Zoom) with audio and video recording. Participants then worked while sharing their screens and thinking aloud. Participants were allowed to use all available support resources, such as internal help files, external video tutorials, or user forums. The researcher quietly observed the participants setting the parameters of the computational design system and took notes. Due to the complex and open-ended nature of the tasks, the research team interrupted the task after 30

minutes and asked participants to continue working on their own as a compensated "homework assignment" without any time limit.

3) Homework Sessions: Following the initial session, participants continued working independently for as long as needed to iterate and produce the final design submission. Participants used the same remote machine and joined a personalized video conference meeting with automatic recording to document their work while thinking aloud. I also deployed a simple web application to prompt the users to verbalize their thoughts. The application analyzed the speech input from the microphone and reminded the user to "keep talking" after twenty seconds of silence. Once the designer felt satisfied with the result, they sent me the 3D files.

4) Post-Task Interview: Within two days of submitting their result, participants completed a one-hour semi-structured interview with a research team member. Participants were asked to reflect on their experience working with the AI tool. The interviews were conducted remotely over video conferencing. The interviewer took notes, and the interview audio and video were recorded. The interview protocol contained 36 questions clustered into three topics: *collaboration with the tool*, *design process*, and *learning process* (see the Appendix for interview protocol). The topics and questions clustered under *collaboration with the tool* were inspired by measures from team learning literature on assessing collaboration quality and effectiveness of human teams [49, 165, 240, 360]. These include perceived team roles and coordination, communication between user and tool, conflict resolution, timing, and (shared) mental models.

3.4.2 Analysis

To gain insight into research questions **RQ3.1a** *What challenges do designers face when learning to co-create with computational AI tools?* and **RQ3.1b** *How do designers overcome these challenges?* I (1) evaluated the design outcomes and analyzed more than 40 hours of think-aloud videos and 17 hours of interview recordings using (2) video interaction analysis of think-aloud videos, and (3) reflexive thematic analysis of think-aloud sessions (videos, transcripts) and interview transcripts.

Evaluation of design outcomes

I evaluated the effectiveness of Designer-AI collaboration by measuring the time required to complete the task and designer satisfaction with their results as rated on a three-point Likert scale (satisfied, neutral, unsatisfied) during the post-task interview. I also measured product feasibility for the mechanical engineering task by checking the

designed engine brackets against the requirements in the design brief. The structural soundness was validated using finite element analysis (FEA), and the used material was checked by measuring part volume. I also checked the models for shape requirements (i.e., clear bolt holes, body within the bounding box). Since the bottle holder was a more free-form and aesthetic design task, I only checked if the user submitted their design and primarily relied on their self-reported satisfaction with the outcome.

Video interaction analysis

I used *video interaction analysis* [33] of the think-aloud recordings to understand participants' learning process while working with the AI features. To understand how well participants learned over time to use the AI features effectively, I tracked their interactions with the AI features relevant to the design task and documented whether the actions would produce satisfactory outcomes. For Fusion 360, I tracked how participants specified structural loads, mechanical constraints, and the obstacle geometry feature to control the bracket's bolt clearance and overall size. For SimuLearn, I tracked how participants used different AI-assisted features (hybrid and automated optimization) throughout the think-aloud sessions.

Reflexive thematic analysis

To understand participant's challenges, needs, and expectations when learning to co-create with the AI system, I performed a *reflexive thematic analysis* [44] of the interview data (transcripts) and the think-aloud sessions (video, transcripts). I followed an iterative inductive coding process and generated themes through affinity diagramming. I used ATLAS.ti to analyze transcripts, audio, and video.

In the initial coding, the think-aloud and interview transcript data were equally distributed among two researchers who generated preliminary codes utilizing both a *semantic* (what people said) and *latent* (my interpretations of the data) coding strategy. Next, the research team collectively identified initial codes and themes. I generated themes in a bottom-up manner. However, I looked at the data with a mindset of collaboration between the designer and the tool—inspired by previous studies on human-human collaboration, co-creation, and team learning [49, 240]. I also tried my best to identify and separate usability issues from the codes and themes to avoid confoundment.

The two researchers then coded the think-aloud recordings to document where designers specified system parameters or evaluated design outcomes. These moments

allowed me to find many of the problems that designers faced. I also coded non-verbal expressions like mouse gesturing or screen annotations that showed how designers attempted to communicate.

Finally, I created *summary videos* highlighting specific situations related to co-creation with the tool (e.g., designers confused by AI-generated outcomes). The video clips were annotated with a time code, participant ID, and a contextual description of the situation to share and discuss with the entire research team (for an example, please see the video figure in the supplementary material). The research team collectively analyzed the think-aloud summary videos in a half-day session and discussed the themes. I completed the qualitative analysis by iteratively reviewing and revising codes and themes until I identified a stable network of coherent and rich themes.

TABLE 3.2: Evaluation of design outcomes for the engine bracket design task (left) and bottle holder design task (right). The designer's satisfaction with the outcome is rated with green=satisfied, yellow=neutral, red=unsatisfied. For the engine bracket task, meeting structural and shape requirement checks are rated as X=fail, check mark=pass.

ID	Durat. Min.	Num. Iter.	Des. Satisf.	Struct. Req.	Shape Req.
P-F01	61	3	☹	X	✓
P-F02	85	2	🙂	X	X
P-F03	54	2	☹	X	X
P-F04	105	9	☹	X	X
P-F05	140	6	🙂	✓	✓
P-F11	122	6	😐	X	X
P-F12	160	5	😐	X	X

Engine Bracket Task

ID	Durat. Min.	Des. Satisf.
P-S01	97	☹
P-S03	190	😐
P-S04	225	☹
P-S06	166	😐
P-S07	63	☹
P-S10	189	🙂
P-S11	151	☹

Bottle Holder Task

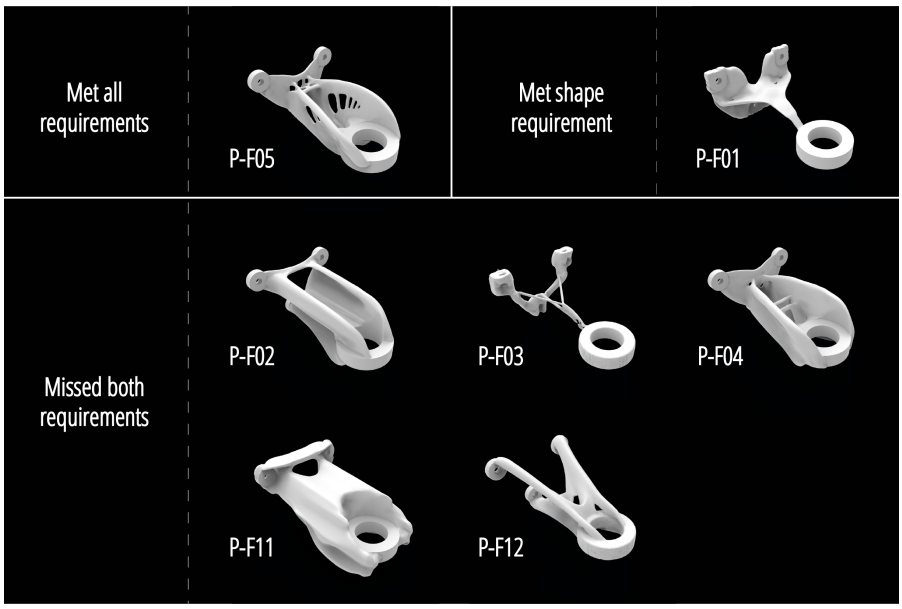


FIGURE 3.4: Overview of engine bracket designs created by participants. P-F05 met both the shape and structural requirements. P-F01 only met the shape requirements (the bracket has holes for bolts with sufficient clearance and respects the specified dimensions but is structurally too weak). All other brackets missed both requirements and were either too heavy, weak, larger than the specified dimensions, or did not have enough bolt clearance. For simplicity, I only show one design option per participant. Please see the Appendix for all submitted design options.

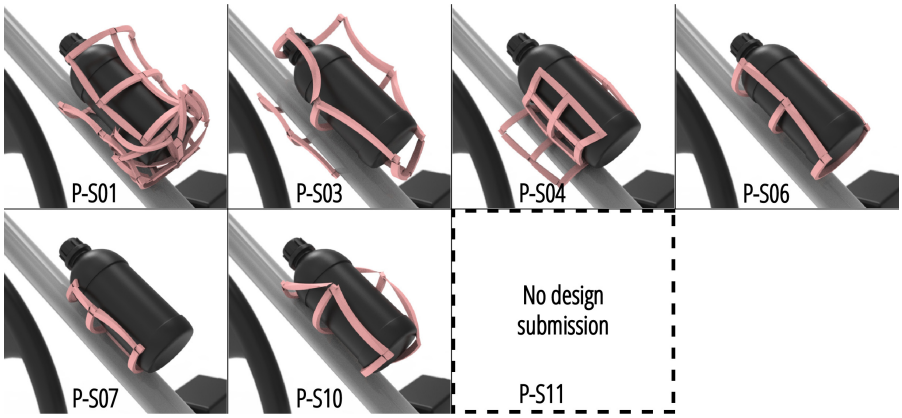


FIGURE 3.5: Overview of bottle holder designs created by participants. Participants came up with different designs using the shape-changing grid material in various ways. Participant P-S01 was not able to control the shape as intended and improvised by stacking grids together. While some designs (P-S06, P-S07) would not be able to hold a bottle, some designers (P-S04, P-S10) utilized the shape-changing grid structure to create promising bottle holder designs. Participant P-S11 stopped working on the task out of frustration and did not submit a design.

3.5 Results

Overall, participants expressed seeing potential value in the AI systems to support their design process—especially that the tools would enable them (at least in theory) to create and explore more complex designs in a shorter time than without AI. However, most faced unresolvable challenges in learning to effectively co-create with the tools. In the following subsections, I provide a brief overview of participants' performance on the design tasks and then explore what challenges they faced in co-creating with them (RQ3.1a). I then highlight what some successful learning strategies looked like (RQ3.1b).

3.5.1 Design Tasks Outcome Summary

In the engine bracket task, all participants were familiar with designing similar mechanical components by considering forces and constraints. Generally, such a task is a standard exercise in engineering education, and it was comparable to the example provided by Autodesk in the introduction video that the participants watched. Participants required between 54 and 160 minutes *(M=104, SD=39.7)* to complete the task (see Table 3.2 left). No participant mentioned in the post-interview that the task itself was too difficult for them. Yet, only participant P-F05 was able to produce a self-satisfactory design that met both shape and structural requirements (see Figure 3.4 and Appendix for additional designs). P-F02 was also satisfied with their design but opted to manually refine the generated geometry that did not meet the requirements (i.e., using excessive materials and blocked bolt holes). I was surprised to find that few engineering participants produced satisfying results, even though they were familiar with the type of design task. Designers struggled to perform this otherwise familiar design task when they attempted to do so with AI assistance.

For the industrial or architectural designers, designing a bottle holder in 3D was not perceived as difficult. However, working with shape-changing material structures was new and everyone expressed in the post-interview that working with the shape-changing material was "unintuitive" and "challenging." Participants worked on the task between 63 and 255 minutes in total *(M=154, SD=56.6).* All but one participant submitted a bottle holder design (Figure 3.5). This participant stopped working on the project after 151 minutes because he felt he could not control the material well, even with the AI. In the end, almost all designers were either dissatisfied with their final design or had a neutral opinion (see Table 3.2).

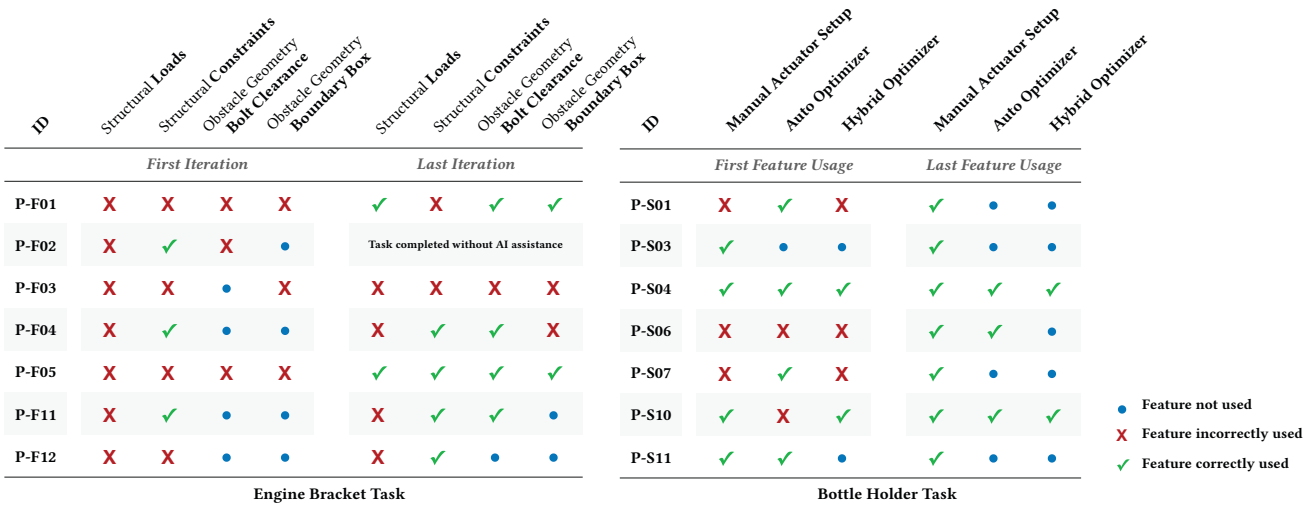


TABLE 3.3: Schematic overview of each participant's learning process of design task-relevant features of the AI system. Left: Engine Bracket Task. Right: Bottle Holder Task. Correct use of input parameters or AI tools is shown with check marks. Increasing numbers of check marks from the first iteration to the last iteration suggest participants learning to work with the AI system.

Table 3.3 shows that only a few designers understood how to set relevant parameters for the design task to produce satisfying results on the first attempt in both tasks. In Fusion 360, most designers learned over time to successfully specify the structural constraints and obstacle geometry for bolt clearances. However, many designers failed to correctly specify structural loads and the boundary box. As a result, some submitted designs were too heavy, weak, or larger than the specified dimensions. In particular, one designer (P-F02) decided to manually refine the design after the first iteration because they felt more in control this way. For SimuLearn, all participants quickly learned how to control the grid shape using manual adjustments and rapid simulations (Table 3.3 right). However, many avoided using the other two AI-assisted features after an unsuccessful first attempt and continued to work manually (see Section 3.5.2 for further explanation).

3.5.2 Challenges Designers Faced When Learning to Co-Create with AI (RQ3.1a)

I clustered challenges designers faced when learning to co-create into three themes: *Understanding and fixing AI outputs*, *working "collaboratively" with the tools*, and *communicating their design goals to the AI.*

Challenges in understanding and adjusting AI outputs

Designers were often confused about the generated results and had difficulty understanding the cause and remedy of "weird" outcomes. This often occurred when generated solutions contained minor aesthetic flaws such as surface bumps, holes, or slightly twisted geometry. Designers wondered about the AI's underlying design rationale and were unable to determine whether such design features were intended or caused by algorithmic glitches.

Furthermore, designers were unsure how to correct the generated designs (e.g., identifying the parameters that led to the problem). Some designers were hesitant to manually refine the generated geometry because they felt uncomfortable changing the optimized structure:

> *"...I also realized that if I was making any change in the mesh maybe I was changing the stress that that area will have. So I didn't feel very comfortable changing stuff because I knew that was optimized for my loads and my material."* P-F04

Designers were also unable to make sense of apparent structural issues in the generated designs, like when the generated parts were unreasonably thin or thick, provided little clearance, or produced confusing grid transformations. As verbalized by this designer while evaluating a generated engine bracket:

> *"it just doesn't make sense that, like this region here, which is obviously pretty thick... well I guess it's not. I just don't think that it can hold up the weight. It says the factor of safety is five. That just seems ridiculous to me."* P-F01 Think-aloud

At other moments, designers were uncertain whether the AI or the user was responsible for fixing the problem. In some cases, designers accepted imperfect results and attributed the flaws to the system. On one occasion, a designer verbalized their

concerns when evaluating a generated engine bracket with insufficient bolt hole clearance:

> *"It's a goofy-looking bracket. I would never design it like that, but this thing thinks it can do it like that [...] It just seems [that there needs] to be bolt heads and stuff, so I just don't see that bolt fitting in this area. But this is what it gave me, so I don't have a problem with that."* P-F13 Think-aloud

Challenges in working "collaboratively"

Designers frequently expressed challenges in sharing control with AI-based design tools, noting that it often felt like these tools were dominating the design process. In those situations, designers either gave up and accepted unsatisfying results, improvised 'hacky' strategies to work around the AI or abandoned the AI assistance altogether and proceeded to work manually. For instance, one designer using SimuLearn expressed frustration about having insufficient control over the design process and decided to accept imperfect results:

> "*I feel like the collaborative process [...] it seemed a little difficult to control. I felt that SimuLearn had more control over it than I did.*" P-S01 Interview

Similarly, another participant pointed out the lack of participation opportunities and conversation throughout the design process:

> *"I would say no, that it's not co-creative. The whole program creates the thing but based on my limits. [...] I don't feel like I interact in that creation of the shape. I just worked at the beginning and then I let the program do the rest. I missed that once there is one option, I cannot change anything, I cannot interact with the solutions that the program is giving me."* P-F04 Interview

In other cases, designers developed their own 'hacky' workarounds to attain feasible results. In Fusion 360, this happened when designers tried to use simple boxes as obstacle geometries to limit the material growth within the specified perimeter. However, the algorithm often found ways to "squeeze" in additional, unwanted material through the gaps between obstacle geometries. A Fusion 360 user expressed they felt like tricking the system when they were unsuccessful at blocking material generation:

> *"I think the software did kind of dominate the design process a little bit in that I was making things to satisfy the software instead of it kind of adapting to*

> *my needs [...] I think for something that's like supposed to be so smart and easy to use I wish it gave me more options instead of [me] trying to find these like little tricks. So I don't want to use little tricks. I don't want to use a hidden kind of 'I'll trick the software into thinking that it's correct.'"* P-F05 Interview

Finally, designers often avoided AI assistance when they could not learn to co-create with it effectively. This issue was particularly frequent when designers used SimuLearn, given that this tool explicitly allows designers to switch between different levels of AI assistance. When comparing which SimuLearn features designers used to complete the tasks, all designers learned over time to manually adjust, simulate, and iterate the design (see Table 3.3 right). However, only two designers (S04, S10) were able to work with the AI-assistance features (i.e., auto and hybrid optimizer) on the first try. All others expressed that they could not make sense of the AI output, even when they used the feature correctly. These participants subsequently avoided using the AI features and continued to work manually with rapid simulation.

Challenges in communicating design goals to the AI

Designers often had difficulty communicating design goals to the AI system. For example, designers were unsure about the use and implication of certain parameters. Furthermore, many designers recognized their knowledge gaps of parameters that defined manufacturing processes or materials. E.g., a designer was unfamiliar with the specificity of a manufacturing method and kept using default parameter values:

> *"I'll have to see if they have five-axis milling... minimum tool diameter 10 millimeters... uh I don't know, I'm not really a tooling guy... tool shoulder length... [laughs] minimum tool diameter... all right I guess I don't know... I just hit okay on that."* P-F12 Think-aloud

Designers often relied on the AI system's default settings or made assumptions about their effect when they were unsure about the parameters' meanings. Instead of seeking clarification from helpful resources, they often tried to determine a parameter's effect on the final result independently. However, it was oftentimes hard or impossible to notice and trace back parameter influences from the final results.

In the think-aloud sessions, I also observed several occasions where designers had different interpretations of the parameters. For example, in the Fusion 360 task, half of the users made mistakes when converting the loads into the correct unit expected by the

system. Similarly, when applying loads to multiple targets (i.e., bolt holes), the system applied the same load to each of the targets instead of equally distributing the load across the targets, which the designers had anticipated [1]. This mismatch led to higher load assignments and unnecessarily strong and bulky bracket designs. Interestingly, most designers verbalized their uncertainty about the load distribution when specifying parameters, as exemplified by this think-aloud comment:

> *"All the loads... I remember being a little wonky... so... I said three... let's see...what would be the case here? I don't know if all of these three forces are the same... that's the issue. I don't know if this is applying to each one in particular... like if it's 12,000 here, here, and here... or if it's split evenly? I hope it's being split evenly... that's what I'm assuming."* P-F03 Think-aloud

Although participants were aware that the system might interpret the load assignments differently, only a few were actually able to figure out and correct the mistake.

3.5.3 Learning Strategies Among Successful Designers (RQ3.1b)

Here I present findings related to *how designers overcame the previously reported learning challenges* **(RQ3.1b)**. I observed that all designers (after watching the introduction tutorial video) tried to learn to work with the tools through an iterative trial-and-error process. I also observed that participants sporadically consulted different support resources, including software tooltips and help files, and external resources like video tutorials, online user forums, and in some cases asking colleagues for help. Designers sought help from these support resources primarily after encountering interface or usability issues, which they often resolved. However, despite the available support resources, most designers struggled to learn to co-create with the tools effectively. Nonetheless, some designers employed successful strategies that helped them in learning to work better with the AI systems:

Systematically exploring AI's limitations and capabilities

I observed that, early on in their interactions, two designers (P-F11, P-S10) deliberately and systematically experimented with the AI tools to develop a better intuition of the AI's behavior, capabilities, and limitations. These designers conducted tests to understand what effects different parameter values would have on the final result and

[1] This is a known issue in Fusion 360 that many users have discussed in the user forum. See [21].

documented the value-result correspondence to create a mental model. For example, PS-10 realized that their initial design sketches were not feasible with SimuLearn and the shape-changing materials, thus decided to systematically test different extreme grid shapes to hone their mental model of the AI's behavior:

> *"Even though I tried sketching some stuff, I think it just didn't work. So I thought it's better if I just go into the tool and see if I will be able to do this. I tried stuff like folding one corner upwards and one corner downwards or stuff like that. I took lots of screenshots and those really helped me to understand like 'if I do this, then it's gonna behave like that' so I think initially it was a lot of trying to form a mental model and like what's the capability of this tool."* P-S10 Interview

Sketching, explaining and reflecting on design issues

Another strategy that helped designers overcome flawed outcomes was to actively abstract and explain the problem. In the think-aloud sessions, I observed when facing similar fundamental challenges like over-constraining the engine bracket (such that the loads had no effect), some designers were able to understand and overcome the issue by sketching out the bracket's free-body diagram and explaining the acting forces and constraints to themselves. I observed similar strategies in SimuLearn, where participants understood the significant influence of gravity during the transformation process by explaining the process to themselves:

> *"Whoa, I was not expecting that at all... uh... is that just because of gravity? And there's something crazy going on here... there's a lot of gravity... oh is it because I made the thing so big? Yikes, that is not at all what I expected."* P-S11 Think-aloud

TABLE 3.4: Overview of Study 2: Learning with a Peer Guide participants. Six additional participants (3xFusion 360, 3xSimuLearn) were paired with peer guides who had worked successfully with the AI design tools in Study 1.

ID	Group	Age Years	Gender	Domain	Occupation	CAD Exp. Years	Ind. Exp. Years	Guided by Participant
P-F07	F360	27	M	MechE	Student/PhD	>5	1–2	P-F05
P-F08	F360	27	M	MechE	Student/PhD	>5	1–2	P-F05
P-F13	F360	19	M	MechE	Student/BA	2–4	0	P-F11
P-S05	SL	26	F	Arch	Student/MA	>5	2–5	P-S04
P-S08	SL	20	F	Arch	Student/BA	2–4	0	P-S04
P-S09	SL	23	F	Arch	Student/BA	2–4	2–5	P-S04

3.6 Study 2: Learning with a Peer Guide (RQ3.2)

To gain insights into *how designers can be better supported in learning to co-create with computational AI tools* **(RQ3.2)**, I conducted additional think-aloud sessions where designers were paired with experienced peers to support them during the task (i.e., guided sessions). The guide had prior experience using the AI tools and provided support as needed to help participants to more effectively co-create with AI-driven features. Motivated by human-human collaboration, I aimed to derive insights into when and how to effectively support users in learning to co-create with AI by observing the support strategies, pedagogical moves, and communication patterns of the human guide.

3.6.1 Method

I recruited six additional participants (Table 3.4) following the same criteria as described in Section 3.3.1. The guides were recruited from the pool of participants who had completed Study 1 and demonstrated a thorough understanding of the domain, tool, and task (P-F05, P-F11, P-S04). I asked these guides to support the designers in learning to co-create with the tool. No script was provided to the guides because I intended to find possible support strategies from their natural interactions.

All guided sessions followed the same procedure as that of the unguided sessions, except that the homework sessions were limited to 50 minutes and the guides were present to help the designers. The designers and the guides communicated with each

other through audio, screen sharing, and screen annotations. A researcher quietly observed and took notes. After the design session, I conducted separate 15-minute semi-structured interviews with the guide and designer. All sessions and interviews were recorded (video and audio) and automatically transcribed.

I conducted a reflexive thematic analysis to identify situations and themes on how peer guides supported designers to overcome challenges in learning to work with AI tools. I specifically focused on aspects of collaboration and knowledge transfer such as communication, joint information processing, and coordination of actions [240]. I coded the video, think-aloud transcripts, and interview transcripts then generated themes with a focus on communication and learning by analyzing the guide's actions and support strategies that helped designers overcome challenges previously observed in Study 1.

3.6.2 Results – Guided Sessions

The guides supported the designer in using the tools, understanding the AI's behaviors, capabilities, and limitations, and sometimes suggesting and discussing alternative design goals or strategies. Guides primarily reacted to designers' verbalization and actions when they asked for help, expressed uncertainty, or when guides observed common mistakes. To get an impression of conversational dynamics, please see the video figure in the supplementary material. I portray five of the most common peer support strategies that helped designers to learn better to co-create with AI systems:

Guide providing step-by-step walk-through instructions

Guides often provided designers with step-by-step instructions for setting specific parameters. Such instructions were provided in response to designers' actions, such as showing confusion or struggle, but sometimes designers also specifically requested such assistance.

Guide reacting to designers' expressions of uncertainty

I observed that guides were especially sensitive to moments when designers signaled uncertainty or when designers verbalized knowledge gaps with hedging expressions such as *"I don't know"*, *"maybe"*, or *"I assume."* In these situations, the guides often intervened and offered support or suggested alternative design strategies. Here, the guide suggested creating an obstacle geometry in Fusion 360 to prevent material build-up at

the bottom of the part in response to the designer who was wondering how to keep the part within the specified dimensions.

> DESIGNER: *"...but because the bolt is not here, I don't know where it would be..."*
> GUIDE: *"I mean maybe on the bottom face, right? That's what's resting on the body of the ship effectively."*
> DESIGNER: *Yeah, let's go ahead and choose that bottom face."*
> (Designer P-F13, Guide P-F11)

Guide prompting designer reflection on generated designs.

I often observed situations where the guides prompted feedback from designers each time SimuLearn's simulation or Fusion 360's solver had finished. In those moments, peer guides often asked the designer *"Is that what you envisioned?"* or *"Is that what you wanted?"* or even *"Yeah, there you go! Is that how you want it to bend?"*. These prompts triggered designers to reflect on the generated designs, which helped the guides better understand how to provide support.

Guide suggesting alternative means and goals.

Beyond supporting tool operation and technical troubleshooting, the guides frequently suggested alternative means and goals to the designer, as in this dialogue:

> DESIGNER: *"It's going for the green part, but it's not able to figure out like a perfect way to get there without like touching this obstacle geometry."*
> GUIDE: *"Right, well I mean just make your obstacle geometry really long, say, all the way back past the connection to the ship, and then you're saying 'no material is allowed to go here' and that would make sure that you can always get a bolt in. See what I'm saying?"*
> DESIGNER: *"Yeah, I do"*
> (Designer P-F13, Guide P-F11)

Here the guide suggested an alternative way to achieve the designer's goal by enlarging the existing obstacle geometry. This strategy helped designers to better communicate design goals to the AI system and develop an intuition for harnessing the AI's capabilities.

Guide and designer making use of screen annotations and mouse gesturing to discuss design strategy.

An essential part of building understanding between the designer and the peer guide was through nonverbal communication, such as screen annotations, sketches, or mouse gestures. Guides frequently used the screen annotation feature built into Zoom to highlight elements they spoke about by circling or drawing arrows. I also observed that all designers naturally used the mouse cursor to emphasize design features through circling or pointing gestures when explaining something to the guide. Both behaviors are illustrated in this situation where the guide and the designer discussed a strategy to achieve a specific bottle holder shape:

> GUIDE: *"So in this case, maybe I suggest that you move these two points in particular more towards the center."*
> `[Guide draws arrows from points towards the center of the grid]`
> DESIGNER: *"And this one seems to have dropped downwards. Even this point here... The beam seems to be going downward."*
> `[Designer points with mouse at different beams]`
> GUIDE: *"oh this beam right here?"* `[Guide draws an arrow pointing at beam]`
> DESIGNER: *"Yeah."*
> (Designer: P-S08, Guide: P-S04)

Many designers and guides also annotated generated designs and sketched to clarify or illustrate their ideas. In summary, different forms of nonverbal communication helped designers and peer guides develop better shared mental models of the task and collectively overcome design issues.

3.6.3 Designers' Needs and Expectations for Co-Creating with AI-Based Design Tools (RQ3.3)

Based on my observations and interviews from both the unguided and guided sessions, I highlight four themes that capture the *needs and expectations that designers expressed around co-creating with AI-based design tools* **(RQ3.3)**.

Designers expect the AI system to have more contextual awareness about the design problem at hand.

In both systems, participants missed the kind of contextual awareness that a human collaborator might have about a design task, such as a part's function or how the part interfaces with other elements in the environment. Such a lack of contextual awareness was one of the main reasons people thought working with the tool was not collaborative. This lack of context also meant that the tool could not support the designers more proactively like a human partner, as described by this participant:

> *"Certainly it would have saved me some time if at the beginning the software would have said 'oh I see that these are your connection points. Can you actually get a bolt in there?' [...] Things like that would have felt really much more collaborative and helpful."* P-F11 Interview

Others expected the tool to offer more intelligent manufacturing and material suggestions or help them anticipate real-world design issues, as this participant expressed:

> *"I would like to see a design tool that would show me simulations of the water bottle in action, like 'oh is there enough friction' or 'will it actually stay in place while a cyclist is on the bike' and then provide suggestions of how to alleviate those problems."* P-S04 Interview

Designers desire a more conversational form of interaction with the tool.

Most participants complained about the lack of reciprocal interaction between them and the tool. Participants compared designing with the AI systems to *"programming"* or *"working with a skilled teammate who is not listening to you."* Participants wished for a more conversational interaction with the tools, as desired by this designer:

> *"More like a tool that I can have a conversation with while I'm always sure that everything that I'm making is fulfilling the expectation of the piece and the loads and materials and everything."* P-F04 Interview

Designers require support in thinking through design problems

Across all guided sessions, designers appreciated that the peer guide helped them learn to operate the tool but also to *think through design problems*—a feature they would eventually also expect from a co-creative AI tool. As summarized by this designer who reflects on working with their peer guide:

> *"I think working together with the peer guide was actually really helpful because he had a lot of insight. I felt like a couple of things that I didn't really think about just from a fundamental engineering standpoint of how the thing can actually be made. I think that was really beneficial."* P-F13 Interview

Designers expect an AI tool to provide project-relevant work examples

Designers frequently suggested that a co-creative tool should suggest similar task- and project-related examples created by other users. Such examples could help designers learn the system's capabilities and provide creative inspiration, as suggested by this SimuLearn user:

> *"Let's say I'm modeling some form and in real-time, the tool is searching some sort of database to show me some possibilities that other people have previously done, just by the similarity of the shape, and then I'm like 'yeah this shape can go there and go there.' So that's actually a creative input that can help—a bit more like you're designing with someone. It doesn't necessarily have to generate, it can pull up from other people and tell you 'here is how some other designers work with this.'"* P-S10 Interview

TABLE 3.5: Overview of design opportunities and example applications in relation to group cognition and team learning.

Support opportunity	Example Application	Relation to Models of Human-Human Collaboration
Scaffolding inductive learning of AI's capabilities and limitations		
Offer guided interactive mini experiments for exploring AI behavior *Seen in 3.5.3 and 3.6.2*	*System offers designers explorative mini experiments to develop intuition of the AI's responses to specific (extreme) parameter constellations.*	Refining mental model of AI's behavior and capabilities through experiential, hands-on learning; Learning to predict the tool's actions in response to own actions.
Present similar work produced by AI design tool *Seen in 3.6.3*	*System proactively suggests task-related project examples of designers creations with AI system as learning resource and source of inspiration.*	Refining mental model of AI's capabilities and limitations by learning from examples; Aligning designer's expectations with tool's capabilities; Getting inspired by co-creative AI's suggestions.
Prompting designers' planning and reflection		
Support designers in thinking through design problems *Seen in 3.6.2 and 3.6.3*	*System asks designers questions to help reframe or further explore problem and solution space, like "Have you considered other alternatives?", "Could the bottle holder also be placed at a different location on the bike?" System could also suggest alternative strategies and goals to designer.*	Helping designers to stay cognitively engaged in co-creative design process with AI.
Prompt designers' feedback and reflection *Seen in 3.5.3 and 3.6.2*	*Prompting designers' feedback when evaluating generated designs, like "Is this design what you envisioned?" Based on the feedback, system could offer support guidance on how to correct generated designs.*	Helping designers to identify their own mental state and communicate goal to AI system; Information pooling.
Improving co-creative AI tools' contextual awareness of designers' tasks and objectives		
Equip AI with increased contextual awareness of designer, task, and project *Seen in 3.6.3*	*System probes designers to share information about their background, skills and project, like high level design goals and manufacturing considerations.*	Fostering shared mental models of task and objectives by asking for explicit feedback or implicit social inference.
Respond to designers' uncertainties and knowledge gaps *Seen in 3.6.2*	*Enable designers to communicate uncertainties about parameters and own knowledge gaps to system by "flagging" parameters. System could track designers' "level of certainty."*	Foster shared mental models of each other's knowledge and goals. Increasing the feeling of designers' ownership of the mutual activity with AI system.
Multi-modal interactive communication for human–AI co-creation		
Allow a more interactive and conversational interaction between designer and AI system *Seen in 3.6.3*	*Utilizing conversational agentive interfaces for designer-AI co-creation*	Facilitating grounding in communication through turn-taking, acknowledgments, back-channeling, error correction; Mutual confirmation of the reception and interpretation of objectives.
Enable non-verbal communication between designer and AI system *Seen in 3.6.2*	*Designers provide feedback to AI system by sketching design goals or annotating generated designs for refinement (like in a design critique)*	Facilitating grounding in communication by selecting a domain and task-appropriate communication medium.

3.7 Discussion

Co-creative AI design tools have the potential to amplify the abilities of engineering and industrial designers. However, I found that designers face major challenges in (learning to) effectively co-create through understanding and adjusting system outputs, communicating design goals, and working "collaboratively" with the AI. Designers who overcame challenges did so by systematically exploring AI's limitations and capabilities, and by explaining and reflecting on their design issues. The observed support strategies of peer guides comprised step-by-step walk-through instructions, reacting to designers' expressions of uncertainty, prompting designer reflection on generated designs, suggesting alternative means and goals, and making use of screen annotations and mouse gestures to discuss design strategies. Overall, designers expected the AI system to have more contextual awareness about the design problem at hand, desired a more conversational form of interaction with the tool, asked for more support in thinking through design problems, and expected project-relevant work examples from the tool.

I discuss my findings in the context of prior research studying how users learn to work with complex software tools and relate my findings to concepts and empirical findings from research on human collaboration. I highlight design opportunities (Table 3.5) to better support designer-AI co-creation by scaffolding designers in actively exploring the boundaries of AI capabilities and limitations, prompting designers to actively reflect on design problems and observed AI behaviors, enhancing AI systems' contextual awareness of designers' tasks and objectives, and supporting more conversational forms of multi-modal communication between designers and AI systems.

3.7.1 New Learning Challenges for Human–AI Co-Creation

Based on my findings, I believe that the challenges designers faced when learning to co-create with the tools go beyond learning the tools' interfaces. Firstly, all participants were experienced designers familiar with the CAD software's interface and had watched a step-by-step instructional video demonstrating the operation of the AI features. Furthermore, designers used different support resources, including tooltips, help files, video tutorials, online user forums, and asking colleagues for help to overcome interface issues they faced. However, even with the help of these resources, most designers could not produce feasible and satisfying outcomes with AI assistance. While this result may have been due, in part, to the limitations of the AI tools themselves, it is clear that this was not the full story. Indeed, a few of the participants were able to

overcome challenges and learned to co-create effectively with the AI features. Hence I believe that the challenges participants faced when working with the AI systems (such as communicating design goals or understanding AI outputs) were partly *new learning challenges* due to the tool's more active role in the design process. These learning challenges go beyond learning the tool's interface. For example, a major challenge I identified relates to how designers specify all required parameters upfront instead of modeling and testing a part step-by-step. This workflow requires designers to think through the design problem in advance, which is challenging and different from the usual iterative design process. Such findings suggest that these new learning challenges require new support strategies for example, by prompting designers' reflection in response to expressions of uncertainty or suggesting alternative design goals that align with the AI's capabilities.

3.7.2 Toward Models of Human Collaboration as Lenses for Studying and Designing Co-Creative Systems

Participants had trouble learning to predict how the AI might behave in response to the specified parameters. They struggled to make sense of the AI system's reasoning and struggled to correct unwanted design issues. Prior literature on group cognition suggests that to achieve effective collaboration group members should be able to interpret each other's reasoning and predict roughly how their partner might behave in response to their own actions [183]. Similarly, from a team learning perspective, the findings suggest that designers who systematically explored the AI's limitations and capabilities early on were better at predicting the tool's actions in response to their own and produced more satisfactory results. This result is in line with studies on human-AI collaboration in decision-making, suggesting that users learn to better predict the machine's behavior through inductive mechanisms (i.e., via concrete examples and hands-on testing) than via general, declarative information about internal processes [59]. While explainable AI research focuses primarily on directly communicating information about the AI system to the user, recent research has suggested that more engaging and longer forms of learning and deliberate practice might improve human-AI collaboration [170]. However, in addition to supporting honing the user's mental model of the AI's capabilities and limitations, it is equally important for the AI system to have an understanding of the user's capabilities, limitations, and task context to enable more effective human-AI collaboration. Hence, this would require the AI

system to have better contextual awareness of the user and the current task at hand. I further discuss the resulting design opportunities in Section 3.7.3.

Most designers felt the tools were uncollaborative and had more control over the design process than they would have preferred. As a consequence, they accepted imperfect results, developed improvised workarounds, or avoided AI assistance altogether. Previous studies on group cognition and team learning suggest that group members feel more ownership of the mutual activity when the group learns to coordinate cognitive capabilities among participants—united by their interpretations of each other's mental states [250]. Groups can reach this level of collaboration by an active process of communication, joint information processing, and coordination of actions [240].

My findings from the guided sessions show that the guides' active communicative support strategies, such as reacting to designers' uncertainties and providing step-by-step instructions, helped designers learn to work confidently with the AI. Furthermore, when peer guides prompted designers' feedback and reflection on generated designs, the designers were required to articulate their intentions explicitly. As a result, both the guides and designers were able to discuss and better coordinate further actions to improve the outcome.

3.7.3 Design Opportunities and Future Work

The following section highlights several design opportunities I identified in the findings to support designers in learning to co-create with computational AI tools.

Scaffolding inductive learning of AI's capabilities and limitations

As discussed, some designers learned the AI system's capabilities and limitations by testing and documenting the effectual correspondence between various parameters and the generated result (see Section 3.5.3). In the guided sessions, peer guides also frequently provided step-by-step instructions for setting up parameters or walked designers through the sequence of steps (see Section 3.6.2). These inductive learning strategies helped designers to better predict the AI's behavior and understand its capabilities. Previous work on novice-AI music co-creation has also found that users systematically tested AI limitations to hone their mental model of the system's behavior [221]. Going further, to better support users in this learning activity, future co-creative systems may offer designers a set of hands-on mini-guided 'experiments' to better understand the system's responses to specific (extreme) parameter inputs [185]. Systems may also offer designers opportunities to view sets of examples of input-output pairs

to help designers develop useful mental models of an AI tool's generative capabilities and limitations (cf. [252]). A co-creative tool may also proactively recommend similar tasks and project-related examples created by other human-AI teams (see Section 3.6.3) to help designers learn the system's capabilities and provide creative inspiration.

Prompting designers' planning and reflection

Participants who were more successful at co-creating with the AI tools did so by abstracting and explaining their problems—either to themselves during think-aloud sessions or to the peer guides (see Sections 3.5.3 and 3.6.2). Literature from the learning sciences shows that self-explanation positively affects understanding and problem-solving [364]. In addition, participants from the guided sessions appreciated the guides' prompt for reflection on AI behaviors and their suggestions for alternative design goals or strategies (see Sections 3.6.2 and 3.6.3). Such actions helped designers think through the tasks and plan actions with AI tools. Conversely, designers who did not reflect on the design problems were unable to learn to understand the system's behavior, capabilities, and limitations well enough and failed to produce satisfactory outcomes. This observation was more prevalent in Fusion 360 than in SimuLearn since Fusion 360's long simulation time doesn't support the kind of rapid interactive adjustments as SimuLearn did, thus requiring the user to strategize their actions in advance. Hence, supporting users in thinking through the design problem for specifying parameters ahead would be especially beneficial for AI tools with longer processing time.

One additional explanation for why participants failed to produce satisfactory outcomes might be that AI systems can lead designers to over-rely on their support, creating an *"illusion of success"* that reduces their effort in solving the design problem [404], something I saw when designers accepted results even when they appeared unfeasible (see Section 3.5.2). To compensate for this tendency, a co-creative system may help designers reframe the problem or further explore the solution space by suggesting alternative goals or asking *generative design questions* [105] like *"What could other alternatives look like?"*. Moreover, actively reflecting on the design process is an essential part of professional design practices [320]. Much like the guides, a co-creative system could prompt feedback and active reflection on observed AI behaviors or generated designs by asking deep reasoning questions about the results, such as *"Is this generated design what you envisioned?"* (cf. [60]). Based on the feedback, the system may then offer support and help to coordinate further actions. This strategy would be complementary to the inductive learning support described in the previous section.

Improving co-creative AI tools' contextual awareness of designers' tasks and objectives

My results show that designers felt a lack of the tool's awareness about the design context and therefore missed the kind of proactive support a human partner might provide (see Section 3.6.3). While building contextual awareness into AI systems has long been a tradition in HCI research, it also presents many technical challenges. However, in the context of co-creative design tools, promising directions are being explored. For example, the system could derive its user model through explicit and implicit mechanisms to develop a shared mental model of the context by asking the user for information about the specific design task (i.e., parsing a written design brief for context and goals [347]) or infer design goals from user behaviors [200].

From a group cognition and team learning perspective, contextual awareness also includes an understanding of the other team member's existing or missing knowledge about the design task. Based on my findings (see Section 3.6.2), a co-creative tool might learn about the designers' knowledge by responding to verbalized knowledge gaps such as *"I don't know"* or responding to hedging expressions such as *"maybe"* or *"I assume."* This observation is in line with research showing student learning is positively affected by human tutors' responses to their expressions of uncertainty [117]. Based on this phenomenon, literature has explored how intelligent tutoring systems can detect and respond to student hedging [117, 286]. A co-creative system may also allow designers to communicate uncertainties about parameters and their own knowledge gaps. For example, designers may flag an "???" checkbox next to a parameter's input field to signal uncertainty. The AI system could then track designers' "level of certainty" for each parameter and provide reactive help or re-surface those parameters later in the design process to identify possible reasons for unexpected outcomes.

Multi-modal interactive communication for human–AI co-creation

My findings show that designers felt communicating with the AI was like giving instructions without receiving much feedback (see Section 3.6.3). Results from the guided sessions show that peer guides used conversational strategies like confirming the reception of each other's goals or asking for clarifications when they were unsure about the other's intentions. As conversation is widely seen as a vital mode of designing [320] and empirical work suggests that much of design work lies within conversations between collaborators and stakeholders [201], co-creative systems should consider using more back-and-forth conversation as an interaction interface. Furthermore, studies on

team learning show that the forming of effective, shared mental models is strengthened through an active process of iterative negotiation between team members, involving "constructive" forms of conflict, argumentation, and resolution [148, 165, 360]. Such a strategy may also prove useful for negotiating design goals.

My findings from the guided sessions also suggest that non-verbal communication may support design partners in developing better shared mental models of design goals (see Section 3.6.2). Designers and guides discussed goals and strategies by pointing at features with the mouse cursor or sketching with the screen annotation feature, similar to a *'spatial-action language'* described by Schön [320] which explains gesturing and drawing along with verbal expressions as typical forms of communication in traditional design critique sessions. Such non-verbal interaction is still an underutilized medium in human-AI co-creation. Allowing designers to use sketching, annotation, or gesturing atop generated results may help them communicate design goals to an AI system.

3.7.4 Human–AI Co-Creation Beyond Manufacturing

Although I identify opportunities to support learning to co-create with AI systems in the context of manufacturing, many of my findings could also apply to other human-AI co-creation domains such as image, music, or text generation. Given recent advancements in generative image AI models (such as DALL-E [295] or Stable Diffusion [307]) with fast release cycles of new tools and capabilities, supporting creative professionals in learning to effectively co-create with such tools might become increasingly important. Furthermore, many prompt-based AI models like DALL-E expect users to express their goals through text prompts, which is an unfamiliar modality for most of today's visual designers. Consequently, my findings suggest that AI tool users across many domains could be supported in learning to better co-create with AI systems by scaffolding inductive learning of the AI's capabilities, prompting users' planning and reflection, improving the tools' contextual awareness of tasks and objectives, and facilitating multimodal interactive communication between tool and user. Future work might further explore interfaces for supporting learning to co-create with computational AI tools in other domains beyond manufacturing.

3.8 Limitations

This study has several limitations. First, the participants only represent a subset of engineering, architectural, and industrial designers. Although all participants had relevant training in their design fields and worked with 3D CAD software, most had minimal industry experience. I also included three professionals with substantial industry experience to compensate for this imbalance, however, even these participants struggled. Further, the self-selected participants in my study were presumably interested and open to the idea of co-creating with an AI system. Thus, some of my findings may be reflective of this openness to co-creative work. Second, both AI tools are still early in development, and I noticed user experience issues that could benefit from improved UIs. To my best extent, I isolated UI issues from more fundamental challenges in learning to co-create with AI systems. Although the tools are relatively new, I believe my findings provide value in exploring new opportunities for co-creative design systems. Finally, despite my effort to make sure the design tasks were realistic, designers knew that they were in a research study and that the designs would not be manufactured. In a professional context, participants might have spent longer learning the tool to produce feasible designs.

Chapter 4

Exploring the Potential of Metacognitive Support Agents for Human-AI Co-Creation

---

This chapter was adapted from my paper:

> Frederic Gmeiner, Kaitao Luo, Ye Wang, Kenneth Holstein, and Nikolas Martelaro. 2025. "Exploring the Potential of Metacognitive Support Agents for Human-AI Co-Creation." *In Designing Interactive Systems Conference (DIS '25), June 05—June 09, 2025, Funchal, Madeira. ACM, New York, NY, USA.*

4.1 Overview

This chapter describes the exploration of novel interaction techniques to enhance designers' cognitive engagement and outcomes when working with GenAI tools. Specifically, I investigated how different metacognitive support strategies, including reflective questioning, encouraging planning and sketching, and expert-driven support, influenced design performance and problem-solving processes in an empirical study with mechanical designers. The findings provided evidence that metacognitive support led to higher-quality designs, particularly when strategies like mental simulations and sketching aided intent formulation and problem exploration. However, the study also revealed tradeoffs, such as the risk of overreliance on agent support and the limited effectiveness of reflective questioning when designers had already formed incorrect assumptions. I conclude this chapter by discussing how integrating adaptive metacognitive support into AI-driven design workflows can help designers navigate complex

design tasks more effectively by providing timely guidance and promoting deeper cognitive engagement.

4.2 Background

4.2.1 Multimodal and Collaborative CAD Systems

Current computer-aided design (CAD) tools primarily rely on WIMP (Windows, Icons, Menus, Pointer) interfaces, using pointer movements and keystrokes as input [259] to modify geometry in visual-spatial interfaces. However, multimodal inputs for CAD work, such as gestures or speech in combination with WIMP [174], can offer advantages, as experimentally demonstrated by Ren et al. [299]. This combination—leveraging the split-attention [22] and the modality effect [57]—allows users to process visual and auditory information simultaneously, which can reduce cognitive load and enhance performance in complex tasks. Similarly, user research studies often utilize the concurrent think-aloud protocol [197] to understand participants' thoughts and actions by encouraging them to speak about their thoughts as they perform a task. Concurrent verbalization can offer rich insights into users' knowledge and intents while only slightly increasing cognitive load during complex tasks [361] such as annotating existing CAD models through speech [285]. Inspired by the think-aloud protocol, Krosnick et al. [193] propose the interaction paradigm of *think-aloud computing* where computer users are encouraged to speak while working on a CAD design task to transcribe and capture rich knowledge with relatively low effort in real-time. Other empirical research shows how conversation and real-time support during CAD sessions can improve problem-solving. For example, revealing unique communication patterns that make multi-disciplinary engineering design work more effective [308] or systems for supporting CAD users by connecting them with human CAD experts in real-time [64, 167] or automatically provide context-sensitive learning resources [236].

Think-Aloud as an Intent-Elicitation Strategy for AI Agents. This prior work motivates voice as a promising interaction channel for metacognitive support in CAD: designers' visual attention is already heavily occupied by the graphical model, simulation results, and parameter interface, while spoken interaction can let them articulate goals, uncertainties, and reasoning without leaving the design workspace. In this thesis context, the think-aloud setup is therefore not only a study method but also part of the interaction premise: verbalization can expose designers' evolving intentions and

problem understanding so that support agents can respond to the design process in real time.

In this study, I aim to explore voice-based agent support interfaces and think-aloud user interactions for augmenting GenAI-driven CAD workflows by enabling low-effort continuous speech-based user intent and context-elicitation.

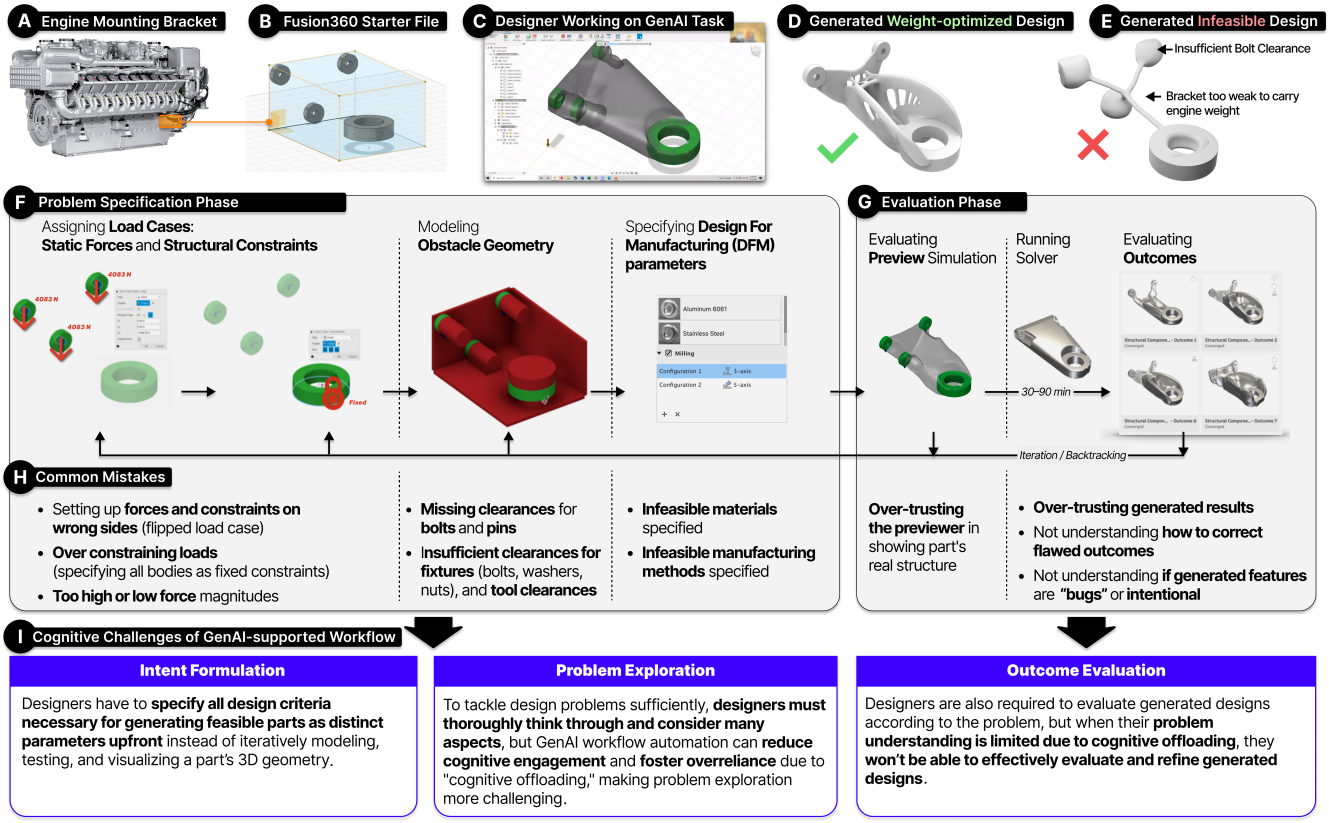


FIGURE 4.1: Overview of the Fusion 360 design task (A-E), workflow (F-G), common user mistakes (H), and cognitive challenges (I). Note that this task matches the "bracket design" task from the previous Chapter 3. The task involves (A) designing an engine bracket that connects the engine to a damper. (B) A starter file containing connection holes and bounding dimensions is provided to the users to initiate the design in (C) Fusion 360. The user is prompted to create (D) a viable design while minimizing weight and avoiding (E) infeasible features. (F–G) The workflow involves six steps, and based on the AI system's solutions, the user may iterate the design by adjusting the design constraints and criteria to produce new solutions.

4.2.2 Cognitive Challenges of AI-Assisted Mechanical Design Tasks in Fusion 360 Generative Design

In this chapter, I aim to develop support agents that help designers overcome the cognitive challenges they face when working with AI-assisted design tools. As a case study for designing and evaluating the support agents, I replicate the mechanical design task from the previous chapter's study (*designing an engine mounting bracket* in Fusion 360 Generative Design, see Section 3.3.2). As detailed in the previous chapter, I found that mechanical designers working with AI for the first time often failed to create feasible mechanical parts despite being familiar with the design tasks and CAD tools in general. Concretely, I observed that while most designers learned to specify some of these input parameters successfully over time, many failed to correctly specify structural loads and obstacle geometry for sufficient part clearances. As a result, most of the submitted designs were unfeasible because they were either too heavy or weak, larger than the allowable dimensions or had insufficient clearances around bolt holes, preventing the bracket from being mounted.

The most common errors that designers faced in this task are listed in Figure 4.1 H, grouped into four categories related to (1) insufficiently specified loads, (2) obstacle geometry, (3) materials and manufacturing options (DFM), and (4) outcome evaluation. These clusters of mistakes relate to key cognitive GenAI workflow challenges of **intent formulation**, **problem exploration**, and **outcome evaluation** (see Figure 4.1 I).

In the following section, I describe how I probed different support strategies aimed at helping designers overcome these challenges.

4.3 Constructing Support Agent Design Probes

My motivation was to study and compare different metacognitive support strategies in the context of GenAI-supported design tasks. As shown in the previous chapter, existing support resources and strategies (such as help menus, online forums, or video tutorials) seem to be ineffective in helping designers overcome the cognitive challenges involved in working with GenAI. Therefore, I speculated that metacognitive support strategies such as asking reflective questions to prompt self-explanation and planning and sketching activities might be more effective.

Inspired by previous work and findings on metacognitive support, I therefore asked:

- *What if a support agent simply asked questions? Could this in itself be enough to promote productive reflection-in-action and improve human-AI co-creation?*
- *What if an agent prompted designers to plan and sketch while supporting these activities through suggestions?*

TABLE 4.1: Overview of agent probes' support strategies and behaviors.

	SocratAIs	HephAIstus	Expert Freeform
Main Strategies	**Asking Questions**	**Planning, Sketching, with Suggestions**	**Freeform: Determined by Invited Experts**
Support Principles	Asking questions to **prompt self-explanation and reflection** [45, 96].	Providing **planning and sketching support** [7, 42] while supporting these activities by **suggesting design strategies** and **workflows** [25, 52].	Determined by invited expert acting as wizard
Response to User Queries	Responding with questions only	Answering factual user questions	Determined by wizard
Modalities	Voice	Voice, link sharing, screen annotations	Voice, screen annotations
Timing and Frequency of Messages	Determined by wizard	Determined by wizard	Determined by wizard
Wizard Enacted by	Researcher (first author)	Researcher (first author)	Generative Design experts from Autodesk
Example	`"What might be reasons why the GenAI system generated the shapes this way?"`	`"Can you walk me through your load cases and constraints by sketching a free-body diagram? I shared a link to a board for you to sketch on in the chat."`	

To investigate the impact of these metacognitive support strategies prior to developing functional AI-based agent systems, I engaged in exploratory prototyping and constructed different agents as design probes [39] (see Table 4.1), which were enacted by humans through a *"Wizard of Oz"* approach [76]. The wizards followed guidelines to adhere to each agent probe's general support strategy while retaining flexibility over when to send messages and how to phrase them in response to the in-the-moment context of each participant's session (see Section 4.4.4). Overall, I hypothesized that each of these support strategies could be effective in supporting designers on their own but also likely in combination. However, as a start, I wanted to investigate how certain strategies would work in isolation to better understand their impact, benefits, and tradeoffs on designers' metacognition and design process.

Further, to move beyond my own assumptions, I also asked:

- *How would Generative Design experts support designers new to working with GenAI in this task?*

The following sections describe my three agent probes *SocratAIs*, *HephAIstus*, and Expert-Freeform in more detail.

SocratAIs – Support by Asking Questions

Inspired by previous research on prompting self-explanation [141, 364], *SocratAIs* proactively asks users questions as they complete a design task to **prompt self-explanation and reflection on their design decisions**. Specifically, the agent follows a **Socratic questioning approach** to support designers' metacognition by constructing questions relevant to the phase of a design task that a designer is currently working on, such as specifying the part's loads, obstacle geometry, manufacturing considerations, or evaluating outcomes. These questions are aimed at supporting the designer in overcoming potential design challenges in a given task phase (see Appendix for agent guide). In line with a Socratic questioning approach, *SocratAIs* **only responds to user requests with further questions** and refuses to provide direct answers.

HephAIstus – Support through Planning, Sketching, with Suggestions

HephAIstus *(referencing Hephaestus, the Greek god of craftsmanship)* provides metacognitive support in the form of **planning and sketching** and by supporting these activities with suggestions around design strategies and tool operation. Inspired by prior research on the benefits of externalization activities in design [24, 130], the agent offers deliberate planning and sketching activities parallel to the CAD workspace to help users think through the design problem more strategically and visually. For planning, the agent suggests the user engage in a **project planning activity** by sharing a pre-generated text document outlining critical project-relevant aspects with the user (see Appendix for example document). This strategy aims to encourage users to think through the design task more deeply before switching to the CAD interface.

For the sketching activity, the agent suggests that the designer **sketch out load case-relevant forces and constraints as a *free-body diagram***[1] by sharing a link to a 2D drawing canvas containing the side and top view of the bracket as a starting point (see Appendix for an example). To support the planning and sketching activities, the agent proactively offers suggestions for the design task and software operation, inspired by work on supporting software learning and work processes [64, 167]. This

[1]Free body diagrams are common mechanical visual representations to illustrate the forces acting on a single object in a given situation, helping to simplify complex problems by isolating an object and showing only the forces directly affecting it.

entails providing alternative design options, highlighting overlooked software features, notifying about unintentional execution errors, or recommending tools and techniques to improve the overall design process.

In contrast to SocratAIs' question-asking approach, *HephAIstus* responds to user queries with direct answers, similar to modern chat assistants such as ChatGPT. Lastly, the agent can visually highlight areas on the user's screen to direct their attention to what the agent is talking about.

Expert-Freeform – Support from a Generative Design Expert

Lastly, to explore how human experts attempt to support designers' metacognition in this task, I invited experts in mechanical and generative design from Autodesk (the maker of Fusion 360) to serve as wizards, allowing them to provide their own interpretation of metacognitive support in a freeform manner. For these sessions, I recruited wizards from Autodesk's employee pool via internal mailing lists and snowball sampling (see Appendix Table B.2). Participants ranged between 27 and 47 years of age, with mechanical design experience between 3 and 15+ years. All experts had high self-rated proficiency in Fusion 360 Generative Design, were closely involved with its development, and had substantial experience training others to use the tool. The expert wizards were instructed to support the other designer in working with Generative Design and the design task by controlling the voice agent. I refrained from explicitly telling them to follow a specific support strategy; they were asked to provide real-time support in their preferred way, so long as they did not directly instruct the other designer on what to do.

Common Agent Capabilities

Besides the differing support strategies, each support agent probe:

- possesses (non-exhaustive) knowledge of additive manufacturing and generative design tasks [2];
- has access to the users' screen and think-aloud speech in real-time;
- can identify inconsistencies between the requirements stated in the design brief and the GenAI parameters specified by the designer by comparing the design

[2] Agents were facilitated by two members of the research team or domain experts from Autodesk. All facilitators had extensive experience with mechanical engineering tasks, Fusion 360, and Generative Design (see section 4.4.4 for further details).

brief and screen activities (e.g., detecting over/under-constrained load cases, infeasible material combinations, or wrong force setup)[3];

- can send voice messages to the user and (in HephAIstus and Expert-Freeform cases) annotate the user's screen and share links via chat.

4.4 Study Design

To elicit the impacts, potential benefits, and drawbacks of the support agent probes, I conducted a formative study with trained mechanical designers new to working with generative AI. Each designer was supported by a different agent probe (facilitated by a human operator in the background) while working with the Autodesk Generative Design tool to design a ship engine mount. In this study, I used the same mechanical design task as in the previous study described in Chapter 3 (*designing an engine mounting bracket using Autodesk Fusion 360 Generative Design*).

I used an exploratory "Wizard of Oz" (WoZ) prototyping approach [401], where a human operator controlled the voice agent probes in the background to simulate different support strategies. Instead of only following strict predefined rules, the wizards had certain degrees of freedom in enacting the agent probes' support strategies to explore broader design possibilities and implications in response to emerging situations during user sessions [401] (see Section 4.4.4). Designers were asked to think aloud during the task to elicit their cognitive processes (e.g., mental models [63], learning [398]) and knowledge and intents so that they could be used by the (WoZ-controlled) voice-based support agents. Participants worked between 31 and 99 minutes, then submitted their designs and completed a semi-structured interview to reflect on their experience working with the support agent.

I collected the following data:

- Video, screen, and audio recordings with machine-generated transcripts of the agent-supported think-aloud design sessions
- Audio recordings and machine-generated transcripts of the post-task interviews
- 3D designs created during the think-aloud sessions
- Log files with timestamps of all human-facilitated agent messages

[3]Such requirements could be explicit in nature (e.g., the force the bracket needs to hold) or implicit features, such as bolt clearances, which were not explicitly mentioned in the design brief

4.4.1 Participants

I recruited 20 designers (aged 20 to 42 *(M = 26.1, SD = 5.9))* with mechanical engineering backgrounds from engineering departments of North American universities and through the Upwork freelance hiring platform[4] (see Appendix Table B.1). Participants had between one and ten years of Mechanical Design experience and between zero to ten years of industry experience, as determined via a screening questionnaire. All participants had at least two years of experience using CAD and Autodesk Fusion 360 but no experience working with the Generative Design extension. I recruited participants familiar with Fusion 360 so that they could focus on learning to work with the AI-driven Generative Design feature rather than learning the CAD tool's user interface. Participants included a mix of undergraduates, graduate students, and professional engineers. Before the study, all participants signed a consent form approved by our institution's IRB. Participants were compensated 20 USD per hour.

4.4.2 Design Task and System

Participants were instructed to design a light and strong engine mounting bracket with Autodesk Fusion 360's [20] "Generative Design" feature (see Section 4.2.2 and Figure 4.1 for a detailed description).

4.4.3 Procedure

The study was structured into four phases split into two sessions:

1) Onboarding (30 minutes): After an introduction to the study, participants received a hands-on tutorial demonstrating Fusion 360 Generative Design's core functionalities through a step-by-step example design task.

2) Design Task - Part 1 (up to 70 minutes): After onboarding, participants were introduced to the design brief, task, and starter file containing predefined geometric constraints. They were also told that a virtual AI agent would support them during the design task (except for the members of the No Support group). Sessions were conducted over video conference (Zoom) with audio, screen, and video recording.

Participants worked while sharing their screens and thinking aloud, with research team members following the video call remotely and operating the support agent. Participants were allowed to use any available support resources, such as internal Autodesk help files, external video tutorials, or online user forums.

[4] http://www.upwork.com

Participants worked until they completed specifying the generative design inputs. They then started the Generative Design extension's solver, which completed the first session. Since the solver required 30 minutes of runtime, participants took a 30-minute break and then returned for the second session.

3) Design Task - Part 2 (up to 30 minutes): After the solver finished, participants evaluated the generated designs. If satisfied with the results, they could directly select three designs. Otherwise, they could re-adjust the design criteria and restart the solver, in which case they would return to evaluating the generated designs after the exit interview and select their final designs.

4) Exit Interview and Debriefing (20 min): After task completion, participants participated in a semi-structured remote interview with a research team member. Participants were asked to reflect on their experience working with the Generative Design extension, the think-aloud activity, and the agent support (see the Appendix for interview protocol). Additionally, after the Expert-Freeform-facilitated sessions, I interviewed the experts to gain further insight into their support strategies and challenges they perceived. The interview was audio and video recorded and the interviewer took notes. At the end of the interview, participants were debriefed about how humans had actually controlled the AI support agents.

4.4.4 Wizard of Oz Setup

Overall, I followed an exploratory Wizard of Oz prototyping approach [401] as a design space exploration where wizards had some flexibility in enacting the agent probes. This allowed me to make meaningful comparisons between the support strategies (*asking questions* vs. *planning and sketching support* vs. *expert freeform*) while also giving wizards flexibility in how to enact the different agent probes in detail, such as the exact message timing and phrasing. Below, I detail the instructions given to wizards and the study setting.

Wizard Details

The *SocratAIs* and *HephAIstus* agents were facilitated by experienced research team members trained in mechanical engineering, Fusion 360, and Generative Design. In some sessions, a second research team member with experience in mechanical engineering and generative design was co-present, verbally supporting the wizard.

These wizards followed the following general guidelines:

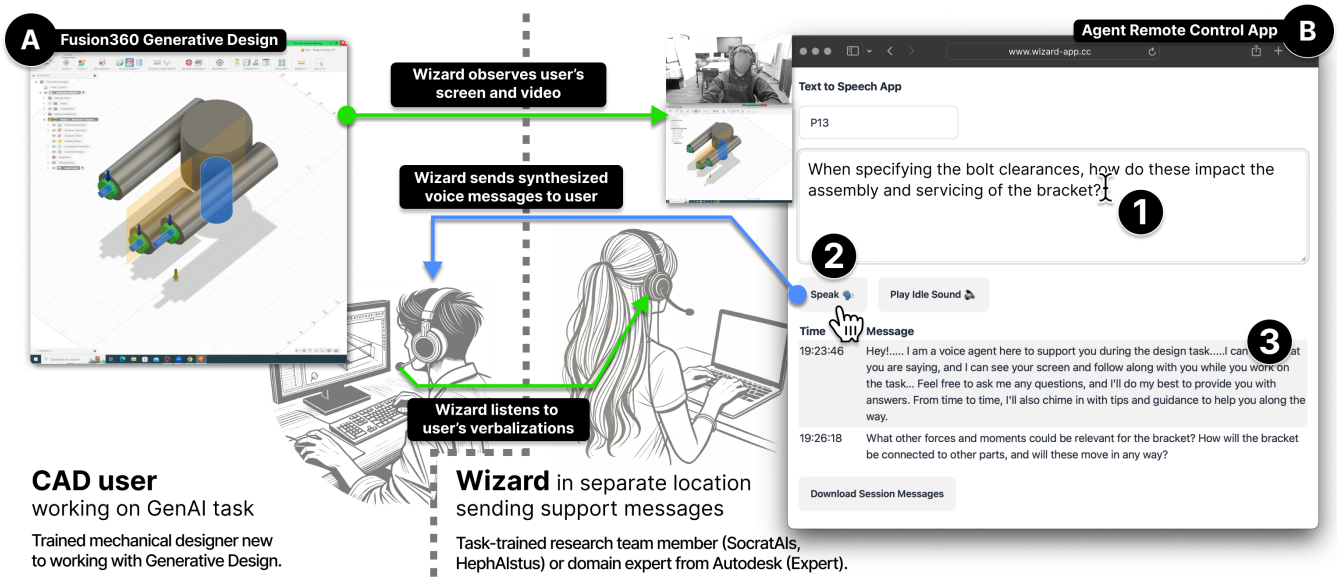


FIGURE 4.2: Process diagram of Wizard of Oz setup. The remotely located wizard (right) followed the designer's actions (left) by listening to their verbalizations and observing their screen and webcam stream. Using a web interface, the wizard could (1) type messages and send these as (2) synthesized voice messages to the user as agent messages. (3) All agent messages were logged with timestamps.

1. Follow the designer's verbalizations and screen actions and pay close attention to the task-specific design steps and challenges as outlined in Section 4.2.2.
2. Pay close attention to inconsistencies between the requirements stated in the design brief and the input parameters set by the designer[5].
3. Never directly tell the participant what to do, but rather provide supportive questions, hints, or suggestions depending on the enacted agent type.
4. Send messages whenever and as often as the wizard considered helpful to the designer, while paying special attention to moments when designers transitioned between design sub-tasks (such as from specifying obstacle geometry to specifying loads), showed hesitation, used hedging expressions, or paused to think silently.
5. Formulate messages in the way the wizard considered most helpful while adhering to the agent's support strategy (e.g., only asking questions).

For the Expert-Freeform agent wizards, I did not provide specific support-strategy guidelines; I only instructed them to provide support in their preferred way, so long as

[5]Such requirements could be explicit (e.g., the force the bracket needs to hold) or implicit features, such as bolt clearances, which were not explicitly mentioned in the design brief.

they did not directly instruct the supported designer on what to do (see Section 4.3).

Setting

For all sessions, participants and wizards were in separate locations during the design task, and communication between the wizards and participants was established via Zoom video conferencing software (see Figure 4.2). Although most sessions were co-located, with participants and wizards in separate but neighboring rooms of the research lab, eight sessions were conducted remotely. In the lab sessions, participants completed the task on a computer workstation running Fusion 360 with the Generative Design extension. Remote participants were provided access to a web-based computer[6] with the same setup for remote sessions.

Participants shared their screens via Zoom and wore an audio headset during the task to capture their verbalizations and ensure they could hear the agent's voice. The wizard joined the same video call using a generic name ('Agent') with a deactivated webcam to follow the participant's screen actions and verbalizations. In addition, the wizard could generate and send agent voice messages using a self-developed web control interface (see Figure 4.2 right). For the Expert-Freeform wizard sessions, in addition to the researcher and the designer, the task expert anonymously attended the conference call in the background and remotely controlled the voice agent via the web interface.

The agent control interface was developed in React.js and uses Google's text-to-speech API to generate the agent voice from the wizard-typed text (see Figure 4.2). The interface features a button to toggle the playback of an idle sound cue to sonically indicate an 'agent is processing' state to the participant. Additionally, the tool logs all generated messages with a timestamp and session ID exportable in JSON format. I used audio-routing software[7] on the wizard's computer to inject the generated agent speech audio into the video call. To mitigate possible gender bias effects, I deliberately selected a gender-ambiguous voice option for the agent, following suggested best practices from prior research [354].

4.4.5 Measures and Analysis

To gain insight into the research questions, **RQ4.1** *How do different support strategies impact the design process?* and **RQ4.2** *What are the perceived benefits and challenges of metacognitive support agents?*, I evaluated the design outcomes and analyzed ~19

[6] using Paperspace
[7] https://existential.audio/blackhole/

hours of think-aloud videos and ~6 hours of interview recordings using a combination of video interaction analysis and reflexive thematic analysis.

Design outcome evaluation

I evaluated the design outcome feasibility by checking the submitted engine brackets against the requirements in the design brief, rating across five criteria, each yielding one point:

1. The structural soundness was validated using finite element analysis (FEA).
2. The feasible load case setup was checked in their Fusion 360 project file.
3. The optimized mass was not extremely light or heavy.
4. The part had feasible fastener clearance (i.e., clear bolt holes)
5. The part's mass and volume fit within the acceptable bounding dimensions.

The GenAI solver generated around 20 designs, and to compensate for possible variability in the generated outcomes, I asked participants to choose three feasible parts from which I then selected the highest-scoring part as their final design.

Video interaction analysis

I used *video interaction analysis* [33] of the think-aloud recordings to understand how agent support impacted participants' design process. For coding the think-aloud video and transcript data, I applied the following coding procedure:

First, I tracked participants' interactions with the Generative Design features relevant to the design task and documented whether the actions would produce satisfactory outcomes. Specifically, I tracked how participants specified **(1) structural loads** (forces), **(2) mechanical constraints**, and the obstacle geometry feature to control the bracket's **(3) bolt** and **(4) dampener pin clearances**, and **(5) overall size**. Second, I tracked the interactions with the support agent probes: For each agent message, they coded if it had an observable impact on the participant considering a new aspect related to the design task, which needed to be apparent from the designer's verbalizations or actions *(coded with 'none,' 'weak,' or 'strong')*. For the design assistant agent probe (HephAIstus), I also tracked users' direct messages to the agent and if agent messages were generally observably helpful to the user *(yes/no)*. Third, I tracked when the agent sent the planning sheet or the free-body diagram sketching board and when the designers interacted with these. From this data, I then created time-series event plots for

each session with `R` and `ggplot2` to visually identify patterns. In addition, I created *summary videos* for each participant, highlighting all situations featuring agent messages or other interesting designer-agent interactions (for an example, please see the video figure in the supplementary material).

Reflexive thematic analysis

To understand participants' attitudes toward the agent support, I performed a *reflexive thematic analysis* [44] of the interview data (transcripts). I followed an iterative inductive coding process and generated themes through affinity diagramming. I used ATLAS.ti to analyze transcripts, audio, video, and Miro for affinity diagramming.

First, I coded the interview transcript data utilizing both a *semantic* (what people said) and *latent* (my interpretations of the data) coding strategy. Next, in an iterative process and in frequent consultation with other research team members, I identified initial codes and themes. Based on the time series plots from the video interaction analysis and the summary videos, I then associated the participant statements from the interviews with specific situations in the design sessions to cross-validate the impact of agent messages and identify additional qualitative themes and interaction patterns. I iteratively reviewed and revised codes and themes until I identified a stable network of coherent and rich themes.

TABLE 4.2: Table summarizing participants' outcome design scores across five criteria (checkmarks) and process statistics by support group.

Support Condition	**SocratAIs**						**HephAIstus**						**Expert-Freeform**						**No Support**					
Participant	**S1**	**S2**	**S3**	**S4**	**S5**		**H1**	**H2**	**H3**	**H4**	**H5**		**E1**	**E2**	**E3**	**E4**	**E5**		**B1**	**B2**	**B3**	**B4**	**B5**	
Passing Structural Analysis (FEA)		✓	✓	✓	✓			✓	✓	✓			✓	✓		✓	✓								
Correct Load Setup		✓		✓	✓				✓	✓			✓			✓	✓							
Mass Optimized		✓	✓	✓	✓				✓	✓			✓			✓	✓							
Feasible Fastener Clearances	✓	✓		✓	✓		✓	✓		✓	✓		✓	✓	✓	✓	✓			✓	✓			
Feasible Part Size	✓	✓	✓	✓	✓	*AVG*	✓	✓	✓	✓	✓	*AVG*	✓	✓				*AVG*	✓			✓	✓	*AVG*
Outcome Quality Score	2	5	3	5	5	**4.0**	2	3	4	5	2	**3.2**	5	3	1	4	4	**3.4**	1	1	1	1	1	**1**
# Agent Messages	38	27	21	17	15	**23.6**	38	64	21	40	32	**39**	27	29	21	36	52	**33**						
# User Messages							15	39	8	8	12	**14.4**	2	8	8	7	12	**7.4**						
# Triggered New Considerations	8	13	2	2	5	**6.0**	4	5	0	4	3	**3.2**	4	8	1	2	10	**5.0**						
Duration (min)	74	74	32	42	56	**55.5**	92	89	59	99	70	**81.7**	37	47	31	43	55	**42.5**	53	37	63	83	59	**59.0**
Normalized Message Frequency	0.5	0.4	0.7	0.4	0.3	**0.4**	0.2	0.2	0.2	0.2	0.2	**0.2**	0.6	0.3	0.5	0.5	0.6	**0.5**						

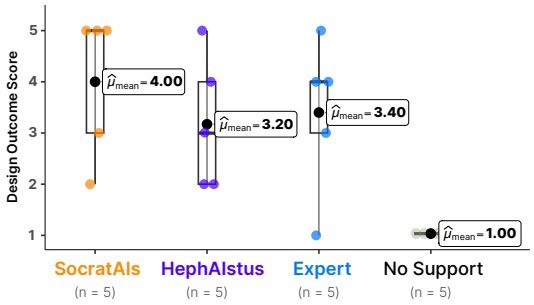


FIGURE 4.3: Plot showing the design outcome scores between agent-supported groups and no support.

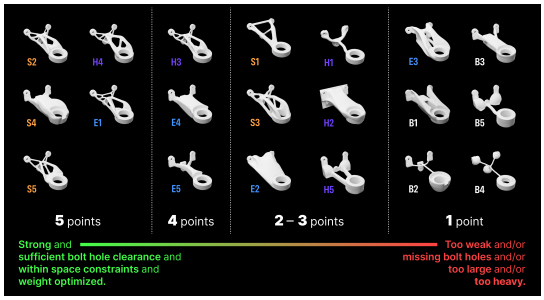


FIGURE 4.4: Overview of engine bracket designs created by participants grouped by quality score (1–5). White IDs indicate participants from the *No Support* group (all scored one point).

4.5 Findings

All participants completed the task without abandoning it. Overall, most agent-supported designers overcame more GenAI workflow challenges and produced more feasible designs than unsupported designers (Table 4.2). However, different agent strategies impacted the design process in different ways. Most designers saw benefits in agent support, but I also elicited various trade-offs and differing preferences for support interactions.

4.5.1 Impact of Different Support Strategies on Design Process (RQ4.1)

4.5.1.1 Design Outcomes

Overall, participants with support produced notably higher-quality parts (see Table 4.2 and Figure 4.3 and 4.4), with an average outcome score of $M = 3.5$ ($SD = 1.4$), compared to participants with no support who had an average outcomes score of $M = 1.0$ ($SD = 0.0$). All participants in the *No Support* group incorrectly specified the bracket's load case, and consequently, no final design passed the structural analysis (see designs with white labels in Figure 4.4). Additionally, brackets created in the *No Support* group had inaccurately specified obstacle geometry, resulting in infeasible fastener clearances or material exceeding the required package size.

In contrast, while the outcome quality varied within and across the agent-supported groups, the majority of supported designers created brackets that passed the structural analysis and had feasible fastener clearances while staying within the required space limitations. Between supported groups, the average design score varied slightly, with SocratAIs-supported users having the highest number of designs fulfilling the load and spatial requirements [8].

4.5.1.2 Frequency and Impact of Agent Messages (across conditions)

In terms of number of messages, agents sent between 15 and 64 messages per session, with the highest group average being $M = 39$ (SD=15.8) in the HephAIstus group, $M = 33$ (SD=11.9) in the Expert-Freeform group, and $M = 23.6$ (SD=9.3) in the SocratAIs group (see Table 4.2). However, these counts of the number of messages sent by HephAIstus and Expert-Freeform also include responses to user-initiated queries and therefore are naturally higher than for the SocratAIs group. To gain a normalized comparison of message frequency between conditions, I calculated the number of messages initiated by the agents divided by the session duration (agent-initiated messages/session duration), which revealed a similar message frequency per minute across agent groups of $M = 0.4$ (SD=0.15) for SocratAIs, $M = 0.2$ (SD=0.03) for HephAIstus, and $M = 0.5$ (SD=0.14) for Expert-Freeform.

[8] Note that given the low sample size per group and highly imbalanced data in the *No Support* group (participants without agent support produced low-quality results matching previous study results, see Chapter 3), I only report descriptive rather than inferential statistics for these data.

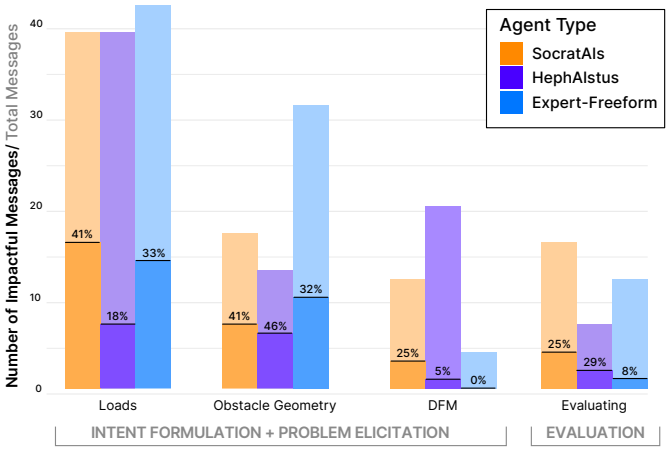


FIGURE 4.5: Plot illustrating the total number of messages per support topic category and agent type (unsaturated colored bars) and the percentage of observable impactful messages that triggered observable new considerations (saturated areas).

Regarding the messages' impact on the design process, the number of messages that triggered observable new considerations (i.e., impactful messages, as defined in 4.4.5) ranged from zero to 13, with means ranging between $M = 6.0$ (SD=4.6) SocratAIs, $M = 5.0$ (SD=3.9) Expert-Freeform, and $M = 3.2$ (SD=1.9) HephAIstus. Interestingly, although the HephAIstus group had fewer messages that triggered observable new considerations compared to the other two groups, their final design outcomes were comparable. This suggests that the planning and sketching activities prompted by the HephAIstus agent may have supported productive design reasoning, even when fewer individual messages were coded as impactful.

Topic-wise, across all groups, most agent messages concerned *intent formulation* and *problem exploration* in the *Loads*, *Obstacle Geometry* and *DFM* categories followed by messages supporting *Evaluation* (see Figure 4.5). Regarding the messages' impact on considering new design-relevant aspects, the highest number of impacts had messages supporting intent formulation and problem exploration: *Obstacle Geometry* (between 32% to 46% across groups) and *Loads* (41% SocratAIs and 33% Expert-Freeform).

In terms of contrast between the number of messages and impact on design considerations, agents varied drastically between the support aim topics: HephAIstus' *Loads* and *DFM* categories had only half or a third of the impact (18% and 5%) as SocratAIs while having a similar or more number of total messages. Similarly, the Expert-Freeform's messages supporting *Evaluating* had only half of the impact (8%)

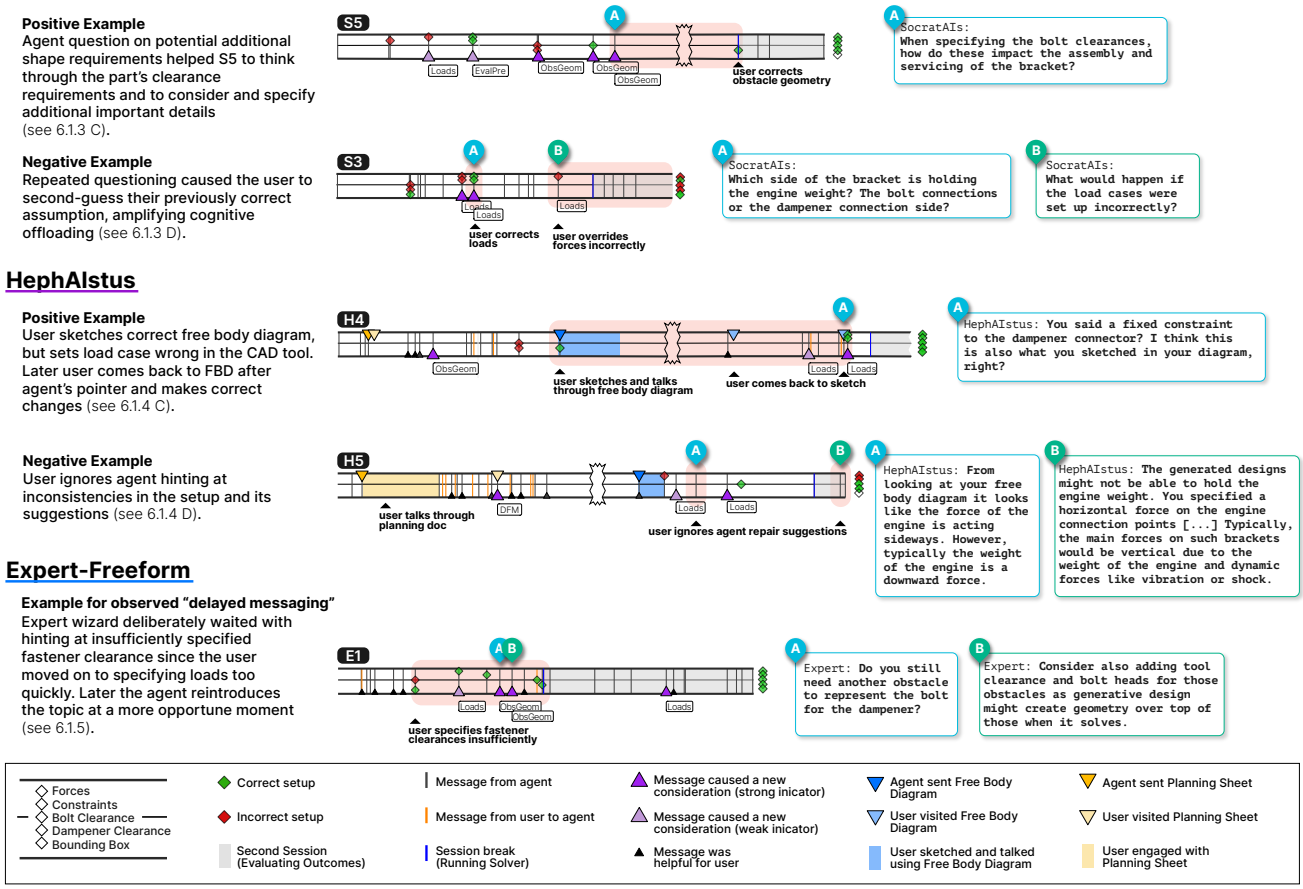


FIGURE 4.6: Timeline excerpts visualizing participant and agent interactions throughout the design task; timelines are divided into lanes, each showing (in)correct GenAI input specifications (diamond shapes) for (1) forces, (2) constraints, (3) bolt and (4) dampener clearances, (5) bounding box (from top to bottom); black and orange vertical lines represent exchanged agent and user messages with purple and black triangles indicating an observable impact on the design process.

as the other support agents while having a similar number of total messages. In contrast, for the *Obstacle Geometry* and *Evaluating* categories, messages in the HephAIstus group had the highest consideration impact while having the lowest number of messages compared to the other groups (46% and 29%).

4.5.1.3 Interaction Dynamics – SocratAIs (asking questions)

Overall, participants paused their think-aloud verbalizations when listening to agent messages (118/118 questions). However, depending on the user's situation, they responded differently: immediately responding to the agent's questions by giving an answer (61/118 questions); finishing their line of thought and sub-task before replying (28/118); pausing to think and reflect silently before verbally replying (13/118); or directly responded with simple acknowledgments after thinking for a while in silence, such as *"Yeah, you are right,"* or *"That's a good point,"* even if the message was not a direct suggestion but an open-ended question (11/118); or providing no response (5/118). Some participants (2/5) asked the agent a question at the beginning of the session, but they stopped asking the agent more questions afterward, recognizing that it would not provide an answer but reply to user requests only with questions.

SocratAIs had mixed impacts on helping designers overcome design challenges across participant sessions (see example timelines in Figure 4.6 and Appendix Figure B.1 for all sessions). In some sessions (2/5), agent messages had an observable strong impact on helping designers overcome design challenges (S2, S5). Meanwhile, in other sessions (2/5), I could only observe weak evidence of impact, where some agent messages had a positive impact, but overall, designers were unable to overcome most major challenges (S1, S3). In one session (S4), the designer created feasible outcomes without facing major challenges or showing observable agent impact, but SocratAIs helped them to consider additional design-task-related factors.

4.5.1.4 SocratAIs' Positive Effects

In most sessions (S2, S3, S4, S5), **reflective agent questions helped users with more precise intent formulation and problem exploration.** For example, an agent question probing reflection on potential additional shape requirements of the part helped S5 think through the part's clearance requirements and consider and specify additional important details, leading to a feasible bracket design:

SocratAIs: When specifying the bolt clearances, how do these impact the assembly and servicing of the bracket?

S5: [Looks at the preview simulation] So this part won't be serviceable [...] you'd need enough clearance for the socket. So I will go back to edit model..." [user adds more obstacle geometry].

(see also timeline S5 in Figure 4.6). Later, S5 reflected on the helpfulness of the agent's message: *"It was asking something about how the obstacle geometry affects the serviceability of the part. That was basically telling me that I needed to leave some clearance for tools and for maintenance. That was very helpful"* (S5).

I also noticed in several cases that **questions were especially effective** in helping designers formulate intent and specify the design problem more accurately **when prompting users to mentally simulate the real-world aspects of the bracket**. For example, S3 had mistakenly modeled the load case reversed to the real-world situation ("flipped" load case), causing the solver to build a structurally unstable bracket. An agent question prompting them to reflect on the part's function from a real-world perspective (`Which side of the bracket is holding the engine weight? The bolt connections or the dampener connection side?`) helped the user update their mental model and load case.

In two sessions (S2, S5), **reflective questions during the preview and outcome evaluation phases helped designers better evaluate and correct faulty designs.** For example, S2 generated brackets based on incorrectly specified loads (forces and constraints assigned to the wrong sides). While evaluating the design, the user noticed the structurally weak parts but was unsure about the cause. An agent question probing deeper thought about the GenAI solver's mechanism caused the designer to realize their flawed mental model and correct the load specification:

SocratAIs: What might be the reasons why the solver generated the shapes this way?

S2: [user thinks] Yeah, because it did not consider building material between these three [bolt connections] ... maybe because of the forces... [user checks forces again] It should be acting downward based on the weight of the engine [user corrects load case].

For confident users, questions helped in problem exploration by considering additional design factors.

In one session, S4 did not encounter major challenges, and while the agents also asked similar questions as in the other sessions, these also had no observable impact on supporting critical cognitive challenges. However, agent questions (such as `"Considering the engine's environment, what alternative or additional assumptions might we make about the load cases?"`) helped the user to consider additional relevant

factors, such as suitable materials for the bracket's maritime environment or additional forces resulting from ship movements. Later, S4 reflected in the interview: *"I think it was pretty useful [...] it prompted me about the assumptions about the loading. And I was like: 'oh yeah, this is on a ship, so it has to survive lateral loads and not just ground loads'." (S4)*

4.5.1.5 SocratAIs' Negative Effects

In one session (S3), **repeated questioning amplified cognitive offloading, leading to flawed results.** At first, SocratAIs' questions during the initial setup phase helped S3 correct their overconstrained load case. However, a later question (`What would happen if the load cases were set up incorrectly?`) made them doubt their initially correctly specified load case setting and change it for the worse (see S3 in Figure 4.6).

In another case, **questions were less impactful in correcting a user's solidified wrong assumptions**. S1 had incorrectly set up a load case with fixed constraints and forces assigned to the same geometry, which would cancel out the impact of the force on the bracket's structure in the solver. An agent question (`Can you walk me through your intention of assigning a force and a fixed constraint to the same geometry?`) caused S1 to provide an explanation of their reasoning, eliciting their incorrect assumptions. But, instead of realizing misconceptions, their explanations reinforced their assumptions, preventing them from correcting issues.

4.5.1.6 Interaction Dynamics – HephAIstus (Planning, Sketching, with Suggestions)

Besides proactive support, HephAIstus also responded to user requests like a voice-based chat assistant such as Alexa or Siri, and users addressed the agent between 8 to 39 (M=14.4) times per session (see Table 4.2 and orange vertical lines in Figures 4.6 and B.1). These user-initiated requests included asking the agent about manufacturing-related facts, such as material properties, or asking for help with load case-related tasks, such as calculating forces. Some users requested confirmation or feedback from the agent on the design process (*"Am I missing anything, agent?"*) or requested guidance (*"Ok, what's next?"*). Overall, these sessions were characterized by phases of active back-and-forth conversations between the user and agent, as apparent from the clusters of dense orange and black vertical lines in the event timelines in Figures 4.6 and

B.1. The proactive messages initiated by the agent included suggestions regarding Fusion 360 operation (such as specific tools within the Generative Design extension) and overcoming design challenges (such as reminding users to use the preview or pointing out mismatches between the design brief and their setup).

Overall, I found that **HephAIstus had mixed impacts on helping designers overcome design challenges across participant sessions.** In 1/5 sessions, the agent messages had an observable strong impact on helping designers overcome design challenges (H4), while in 3/5 sessions, weak impacts on overcoming design challenges were observable (H1, H2, H5). In 1/5 sessions, a designer created almost feasible outcomes without facing major challenges or observable agent impact (H3), but it was clearly observable that the agent helped the user operate the software more effectively.

4.5.1.7 HephAIstus' Positive Effects

The agent-provided sketching board helped some designers in loads-related problem specification and intent formulation. When the agent prompted designers to explain the bracket's load case by sketching a free-body diagram (FBD) and sharing a link to a prepared drawing board, all users followed the link and used the board to sketch out diagrams while verbalizing their thoughts (see blue triangles and highlighted passages in Figure 4.6 and B.1 and example board in Appendix). In several cases, the sketched free-body diagram served as a conversational anchor and reference point between the user and the agent. For example, while H4 had first sketched an FBD with feasible load cases, they then incorrectly specified the load case in Fusion 360. Later, the agent pointed out the inconsistency between the sketched FBD and the load case setup in the CAD tool, which led the designer to correct the input specification (see H4 in Figure 4.6).

All designers explored the agent's project planning sheet, but few revisited it during the session. While 2/5 users quickly went through the document at the beginning, 3/5 users (H2, H3, H5) spent between 3–10 minutes in the document, utilizing its provided structure to talk through, reflect on and plan the design process step by step before starting to work in Fusion 360 (see yellow triangles and highlighted passages in Figure 4.6 and B.1). For example, the planning sheet supported H5 in reflecting on and exploring suitable material options while using the document to add notes about different material characteristics (see H5 in Figure 4.6). However, only two users revisited the document later in the session (H4, H5).

I also observed that in many cases, **proactive agent suggestions reminded designers about overlooked steps, unintentional execution errors (slips), or software features.** For example, the agent reminded H2 to run a preview simulation to better assess bolt clearances before starting the solver (`You might run a preview simulation at a later point to evaluate the clearances before starting the solver`). This message led H2 to run a preview and realize insufficient bolt clearance.

4.5.1.8 HephAIstus' Negative Effects

While 52% of agent messages had an observable impact on helping the user work on the task and operating the software (101/196 messages, see black triangles in Figure 4.6), only a fraction (0.06%) directly helped overcome cognitive design challenges by considering new design-task relevant aspects (16/196 messages, purple triangles in Figure B.1). I also observed that **directly pointing out inconsistencies in the users' setup only helped some users correct existing issues.** For example, in two cases (H1, H5), designers repeatedly failed to correct ill-defined load case setups despite the agent directly pointing these out and providing concrete suggestions for correcting them (see H5 in Figure 4.6). In those situations, participants decided not to follow the agent's suggestions and instead followed their own (partially incorrect) intuition.

4.5.1.9 Expert-Freeform Support Strategies

From analyzing the session videos and post-task interviews with the expert facilitators, I identified several support strategies that helped designers similar to my support agent probes.

I observed that **experts frequently supported designerly thinking and metacognition while also highlighting overlooked design issues to guide users**. Similar to HephAIstus, experts proactively highlighted potential issues, such as missing or misrepresented GenAI parameters, to ensure critical considerations were addressed early.

Additionally, I observed that **some experts used a question-asking strategy** to support users in intent formulation, problem exploration, and outcome evaluation similar to SocratAIs.

Some **experts also deliberately delayed messages** when the user moved on to a different sub-task too quickly and waited to reintroduce the topic later at a more opportune moment (see E1 in Figure 4.6).

Lastly, I observed that **experts also frequently supported users in navigating CAD software features**, helping them by offering guidance on Generative Design

functions, recommending workflow optimizations, helping with calculations such as load distribution, and providing help to locate tools and options as needed.

4.5.2 Perceived Benefits and Challenges of Metacognitive Support Agents (RQ4.2)

Overall, participants appreciated the support from the agents. S2 stated that *"it's doing a good job [...] by assisting you throughout the whole design work"* and H1 noted a perceived efficiency gain: *"I feel like without [the agent], [...] it would have definitely taken a longer amount of time."* Besides positive aspects, participants highlighted trade-offs and challenges, which I present in the following sections, organized around the different support strategies and general agent interactions.

4.5.2.1 User Feedback on SocratAIs (Question-Asking)

Participants consistently found the agents' questions valuable, particularly those that prompted reflection on key design aspects, such as load cases and clearances. These questions helped refine GenAI input specifications by encouraging them to reconsider functional details and correct initial assumptions, as one participant noted, *"The questions [...] helped me reflect and go back over my train of thought and see, 'Am I missing something? Does this look like I'm doing what I'm supposed to do?'"* (S5).

Participants also reported that questions encouraged them to slow down and critically evaluate their thinking, much like a professor would in a one-on-one setting, as S1 explained: *"when [..] you're just sitting down designing by yourself, you don't often run through those things. So having someone to stop you and say, 'Why do you think that works?' is a good check every now and then"* (S1).

Some users found agent questions redundant but preferred them over missing important steps. For example, one participant noted that while *"25% of the questions actually helped,"* the rest pulled their attention away from the current task (S1). Others mentioned that the agents sometimes asked questions about actions they were already performing, which felt unnecessary as they were already thinking through those steps. However, participants acknowledged that redundancy was preferable to missing something important and found the frequency of questions appropriate.

Some found the questions more useful for problem exploration and intent formulation: *"It did a good job [during] the initial part of the setting up of the design"* (S3). **Others saw more benefit in supporting outcome evaluation**, particularly

when analyzing and comparing designs: *"[Questions during outcome evaluation phase were more] valuable because when you're comparing this many designs, it's good to be reminded of what's most important to compare and prioritize"* (S2).

4.5.2.2 User Feedback on HephAIstus (Planning, Sketching, with Suggestions)

The agent-provided project planning sheet was generally perceived as helpful, giving users a structured way to approach their tasks and a document to guide their process, as this participant stated: *"To show you an actual work plan from the design to the actual fabrication and production of the piece is good. [It helps you to] separate [the design process] into different steps and how we're going to work from here"* (H2).

Designers found that sketching helped them visually think through the design problem: *"Sketching a free body diagram was definitely helpful. I mean, it was just good to see before I had set it up in Fusion sort of my plan for where the loads and constraints were gonna go"* (H3). Several participants suggested that the sketching feature could be improved by making it more interactive—for example, by providing real-time calculations, augmenting sketches with force vectors, and offering a library of example diagrams—to support thinking through a part's design requirements.

Many participants also valued the agent's proactive suggestions. For example, users found it helpful when the agent pointed out possible inconsistencies in their load specifications. As H1 described: *"I wasn't sure why [the solver] was generating so thin [parts] and having the agent explain to me, 'hey, it's probably because of the constraints that you set, you have canceling loads, you should not do that.' That was good feedback to modify the constraints"* (H1).

Designers also liked when HephAIstus helped them catch slips and correct mismatches in real-time. For example, one user appreciated a hint that pointed out a mismatch between the force requirements in the design brief and the load setup: *"I thought it was extremely helpful. I [mis]read the instructions with the weight capacity [...] I'm glad I received a prompt to make sure that the weight distribution was accurate because it knew there was a difference between the weight the bracket was supposed to hold and the engine's total weight. I was impressed that the agent provided me that prompt to check and make that design change"* (H1).

Additionally, **participants also found the agent more efficient than searching online or scrubbing through video tutorials for answers.** Being able to ask the agent questions directly about specific software functions or design issues saved time

and allowed them to stay focused on the task without interrupting their workflow: *"I didn't have to stop what I was doing [...] to go to Google and find information"* (H3).

The fact that the agent would also annotate the screen and highlight relevant interface elements was widely appreciated for reducing the need to search for tool functions manually, as one participant noted: *"when I had doubts about specific functions in Fusion, I asked the agent and it was very helpful in that. And the fact that it would highlight the word to click, that was very useful"* (H2).

4.5.2.3 Reflections and Suggestions of Expert-Freeform Wizards

In the post-task wizard interview (after supporting a designer), the task experts reflected on their support strategies and challenges. Some **experts emphasized the difference between operating Fusion 360 and thinking through design problems, underscoring that while both of these tasks are equally important, they often require two distinct "mindsets":** *"I was thinking of it from a 'how do I use Fusion' standpoint. [But it's] kind of like two parts of the brain: One is like, 'I know where the buttons are, I know the workflow,' and there is like, really creative problem-solving"* (Expert 4).

Furthermore, some also suggested that **agent-initiated "design reviews" during outcome evaluation could help users in critical evaluation of GenAI outcomes**. Similarly, others suggested to **introducing deliberate "checkpoints" or reflection phases between design task steps** (e.g., when the user transitions from specifying loads to obstacle geometry), which could allow for better scaffolding of (metacognitive) support throughout the process.

Several **experts also saw the potential to pre-structure workflows by offering early guidance on problem exploration and setting up generative design inputs.** Some highlighted the value of "preemptive" planning activities, similar to HephAIstus' planning sheet—suggesting that preparing a clear design plan before switching to the GenAI tool could improve the overall design process.

Some experts deliberately focused on supporting users in outcome evaluation and also suggested designers to run a solver preview early to obtain visual feedback, helping users quickly assess if their setup was correct. Another observed recurring strategy was that during outcome evaluation, Expert-Freeform agents suggested looking back to realize flawed load specifications, as this expert described in the interview: *"If we get really hefty results like blocky stuff, then as the agent, I can say,*

'Do these parts seem over-designed? Let's look back at our load cases!' And then we can recognize, 'okay, we applied that load to the full load to every entity'" (Expert 1).

Lastly, experts also suggested instead of directly speaking agent messages, to **annotate screen elements to signal available feedback from the agent**, for example, by circling critical parts, to provide users opportunities to initiate a conversation with the agent when desired by clicking on these highlighted regions.

4.5.2.4 Feedback on General Interactions With Agent Probes

Participants generally appreciated the voice modality of the agent, finding it faster and more efficient than typing, as H2 described, *"talking is more time efficient because in chat I'll first have to explain the issue and it will take longer and be less clear"* (H2).

Others highlighted the benefit of voice-based interactions for complex tasks that required creative or visual exploration: *"In a software like this, I could definitely see use in [voice]. Text can be convenient, [but] you might miss it. Having a voice is helpful [for] anything that requires an exploratory or creative process"* (H1). However, some users noted that **voice interaction could be impractical in shared or public workspaces**, such as offices or labs, and would prefer an additional text-based alternative in such environments.

Users also shared differing opinions about the overall benefits of the question-asking and support suggestion strategies. Designers in the SocratAIs group generally valued open-ended questions rather than providing direct answers, as it encouraged them to think critically about their design decisions and allowed for flexibility in their approach, as S5 shared: *"I don't think it should have pulled me straight up the answer. I think it was better to tell me, 'Hey, you should think about this,' because not every part is going to be maintained the same way or serviced the same way [...] By asking you a more open-ended question, it pointed you in a direction, but it also left open the possibility to ignore it"* (S5). However, a few users also suggested that **more direct educational scaffolding with explanations could be more helpful for less experienced users**, while other more experienced users could prefer shorter prompts to ensure that their workflow stayed on track. Some participants in the HephAIstus group also emphasized that to fully rely on the agent, they would need to trust its understanding of complex design contexts and ensure that its recommendations are accurate, especially for critical engineering tasks.

Table 4.3: Overview of design considerations and key learnings.

	Design Considerations / Key Learnings	**Seen in**
*Opportunities for agent-based **metacognitive support***		
A1	Cueing users with **thought-provoking open-ended questions** can help with *intent formulation*, *problem exploration*, and *outcome evaluation* in GenAI-assisted design tasks.	4.5.1.2, 4.5.1.3, 4.5.1.4, 4.5.1.9
A2	Prompting **mental simulations through questions and sketching** can assist designers in thinking through design problems and more accurately formulating intents and specifying GenAI model inputs (supporting *intent formulation* and *problem exploration*).	4.5.1.4, 4.5.1.7
A3	Offering users **agent-driven "design review sessions" during part evaluation** and introducing **dedicated "reflection checkpoints"** to scaffold metacognitive support throughout GenAI-based design processes better.	4.5.1.4, 4.5.2.3
A4	Giving users **control over the type of metacognitive support** depending on their needs and experience level.	4.5.2.4
A5	Providing designers **custom-generated user-editable design checklists** to support planning and scaffolding of design decisions.	4.5.1.7, 4.5.2.2
*Opportunities for agent-based **CAD support***		
B1	Offering **suggestions for design decisions and tool operation** in combination with **metacognitive support** to help **improve users' tool fluency** and **overcome cognitive GenAI workflow challenges**.	4.5.1.9, 4.5.2.3
B2	Enabling users to **verbally request support from agents** can help to maintain focus and reduce context-switching	4.5.1.6, 4.5.2.2
B3	In addition to **voice agent feedback**, utilizing visual **screen annotations and text** can lower mental effort.	4.5.2.2
B4	Agents that follow user behavior over time offer the potential for **proactively providing reminders, hinting at inconsistencies, and suggestions for metacognitive support, tool operation, and design task considerations**.	4.5.2.2, 4.5.1.9
B5	**Visually signaling available agent feedback** for users to optionally engage in can reduce task interruptions.	4.5.2.3

4.6 Discussion

In the following sections, I discuss my findings and their implications for metacognitive design support systems and agent-based CAD support while highlighting key learnings, design considerations, and open questions for future GenAI design support systems (see Table 4.3).

4.6.1 Toward Metacognitive Design Support Systems

My findings indicate that agent-facilitated metacognitive support can play a positive role in helping designers overcome the cognitive challenges of GenAI workflows. I saw that designers receiving some form of support often had improved design outcomes compared to those without assistance[9].

In this study, I categorized support strategies into distinct agent probes to reveal nuanced benefits and tradeoffs but also saw that none of the agents served as a one-size-fits-all solution. This suggests that combining multiple strategies may ultimately prove more effective in practice, and future work should explore systems with blended approaches.

Regarding specific metacognitive support strategies, my findings indicate that **(A1) cueing users with thought-provoking open-ended questions can help with intent formulation, problem exploration, and outcome evaluation leading to improved AI-generated outcomes** (see 4.5.1.2— 4.5.1.4, 4.5.1.9). These findings align with prior evidence on the crucial role of questions within design processes [56, 106, 290]. Similar to other recent work [77, 276], my findings also emphasize AI agents' possible role as facilitators that can stimulate users' critical thinking, which challenges the common notion of GenAI systems as "oracles" that only provide definitive (but possibly inaccurate) solutions or answers. However, going further, a challenge will lie in determining when "asking" versus "telling" the user would be most appropriate. Further investigations could draw on principles from learning science suggesting metacognitive processing may only be effective if preceded by adequate knowledge or initial instruction [35, 141].

Also, I saw that asking questions alone can have limitations: In this study, questions were less effective at challenging ingrained incorrect assumptions, indicating that guidance beyond questioning may sometimes be required—especially when users hold deep

[9]These results in the unsupported group align with my previous findings using the same task with a similar population as described in Chapter 3, and statistical tests also showed no significant differences in population characteristics between supported and unsupported groups in this study.

misconceptions. Similarly, repeated questioning also presented a dual effect: while it often helped users to repair flawed inputs, it sometimes led to over-reliance, with designers thinking the AI might know something more than them or that the AI is right rather than engaging in deeper reflection. This risk of dependency aligns with other findings on in-action feedback during design tasks, where excessive guidance was observed to diminish self-reflection and critical evaluation [100, 404]. Future work should, therefore, explore *when* and *how* metacognitive support systems could provide assistance without increasing automation reliance.

From a technical perspective, recent advancements in natural language processing (NLP) have enabled automated generation of Socratic questions for teaching math [328] or debugging [8] and generating domain-specific educational questions by pre-training LLMs [48]. Building atop such technical foundations, future research could explore design task-specific question generation to prompt designers' self-reflection and critical thinking aligned to specific design domains. Furthermore, inspired by emerging process mining techniques focused on metacognition and self-regulated learning phases [1, 41, 405], future work could further optimize systems by tailoring prompts to designers' specific situational metacognitive needs, such as *intent formulation*, *problem exploration* or *outcome evaluation* phases.

The findings indicate that **(A2) prompting mental simulations through questions and sketching can assist designers to think through and more accurately formulate intents and specify GenAI model inputs** (see 4.5.1.4, 4.5.1.7). These findings align with previous research on design cognition, suggesting that mental simulation presents a vital metacognitive process in design activities [27, 28]. In addition, I saw that providing distinct support for thought externalization and visualization through sketching (a known cognitive amplification strategy in design [24]) helped designers more carefully think through input specifications for the GenAI solver, thus improving designers' intent formulation and problem specification. Overall, metacognitive agent support might potentially be helpful for many design processes, whether GenAI-supported or not. Based on these findings, future work should explore further metacognitive support mechanisms relevant to design within and outside of GenAI-assisted tasks, such as prompting mental simulations through questions or guiding users in gradually sketching and eliciting relevant input criteria.

While many questions may not have directly helped designers, I saw that the *right* reflective question at the *right* time can have a significant impact on the design process. However, anticipating and catching the right moment can be tricky, but some

situations seemed to be more opportune than others. For example, during the GenAI preview and outcome evaluation phases, reflective questions helped designers in assessing and correcting generated parts by linking back component's structural errors to insufficiently specified model inputs (see 4.5.1.4). Likewise, some experts and users also emphasized that support during evaluation phases would be especially useful (see 4.5.2.3 and 4.5.2.1). Building atop this learning, future systems could **(A3) offer users agent-driven "design review sessions" during part evaluation** (similar to reviews in design education or professional collaborations [131, 265]) **and introducing dedicated "reflection checkpoints" between GenAI setup steps** (e.g., when the user transitions from specifying loads to obstacle geometry) to better scaffold (metacognitive) support throughout GenAI-based design processes (see 4.5.2.3).

My findings also revealed differences in user preferences between more confident and inexperienced designers regarding questions versus suggestions (see 4.5.2.4), highlighting the desire for systems to give users **(A4) control over the type of metacognitive support depending on their needs and experience level.**

Lastly, future metacognitive support systems could provide **(A5) custom-generated user-editable design checklists to support planning and scaffolding of design decisions** (see 4.5.1.7, 4.5.2.2).

4.6.2 Opportunities For Agent-based CAD Support

In this study, HephAIstus—in addition to its metacognitive planning and sketching support—had a positive impact on helping users work on the design task and software operation. However, HephAIstus' suggestions helped designers less to overcome GenAI-related cognitive challenges than SocratAIs' questions. Especially for supporting *intent formulation* and *problem exploration* related to load cases, SocratAIs was twice as effective as HephAIstus (see Section 4.5.1.2 and Figure 4.5), indicating that for intent formulation and problem exploration, questions paired with planning and sketching support might be more effective than suggestions. Consequently, I conclude that **(B1) metacognitive support through reflective questions, planning, and sketching is equally crucial for effectively supporting designers in GenAI tasks as providing suggestions for design decision and tool operation** [10] (see 4.5.1.9, 4.5.2.3).

Furthermore, based on the insights derived from HephAIstus and Expert-Freeform, I see various opportunities for systems providing real-time support for design tasks

[10] This is also indicated by the comparable quality of outcomes across the different agent-supported groups.

and software operation, along with metacognitive support in CAD and GenAI design workflows. Notably, I see opportunities for **(B2) enabling users to directly request information and metacognitive support from agents verbally while working on a task**, which seemed to have helped users maintain focus while reducing context-switching (see 4.5.1.6, 4.5.2.2).

Similarly, users also appraised the agent's **(B3) visual tool guidance by directly highlighting relevant GUI elements** (see 4.5.2.2). Recent NLP advancements in speech processing and synthesis [239, 355], as well as the increasing ability of multi-modal AI models to visually understand and operate software GUIs [154, 260], provide promising foundations for future research to explore such multi-modal conversational support agents further.

Additionally, by capturing and responding to user behavior, verbalizations, and (screen) context over time, **(B4) agents can proactively provide reminders, hinting at inconsistencies and suggestions for metacognitive support, tool operation, and design task considerations** (see 4.5.2.2, 4.5.1.9). As multimodal LLM context windows increase, such contextual longitudinal support seems to become increasingly feasible.

Lastly, instead of support agents always directly verbalizing their messages, future support systems could instead **(B5) visually signal available agent feedback for users to engage in if and when desired**, which could reduce task interruptions (see 4.5.2.3).

4.7 Limitations

This study has several limitations. First, the study followed an exploratory prototyping approach [401] that enabled me to compare the different agents' metacognitive strategies while allowing flexibility in how support was delivered (e.g., message timing and phrasing). As a result, the design insights are partially shaped by the individual wizards (myself and four external experts), and repeating the study with different wizards may yield slightly different outcomes. Furthermore, to analyze the impact of agents on the design process, I used video interaction analysis to identify moments when participants visibly considered new, relevant aspects in response to agent messages. While this yielded valuable findings, future work could incorporate additional user-agent interaction dynamics to further surface complementary insights. Second, the participants represent only a subset of engineering designers. While all participants had relevant

training in design and experience with 3D CAD software, many had limited industry exposure. To address this imbalance, I included five professionals with more extensive industry experience. Most of these professional users were part of the Expert-Freeform-facilitated agents, which might have biased the results. However, I disregarded this potential bias since the observed behaviors were similar across all supported groups. Furthermore, as the participants in this study were self-selected, they were likely interested in or receptive to GenAI systems. This openness to AI-supported work may have influenced some of the findings. Third, although I aimed to ensure the design tasks felt realistic, participants knew they were part of a research study and that their designs wouldn't be produced. They might have invested more time learning the tool and thinking through the problem to create practical designs in a real-world setting.

Chapter 5

Prototyping Multimodal GenAI Real-Time Agents

This chapter was adapted from my published paper:

> Frederic Gmeiner, Kenneth Holstein, and Nikolas Martelaro. 2026. "Prototyping Multimodal GenAI Real-Time Agents with Counterfactual Replays and Hybrid Wizard-of-Oz." *In CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain. ACM, New York, NY, USA.*

5.1 Overview

In the previous chapter, I describe how I identified novel and effective interaction techniques to enhance designers' cognitive engagement and outcomes when working with GenAI tools. Informed by these encouraging findings, I built a functional prototype of *SocraBot*, a multimodal voice-based support agent aimed at cognitively assisting designers with GenAI-based mechanical and manufacturing design tasks. SocraBot follows along with a designer's verbalizations and screen actions while working on a task and proactively provides cognitive support messages in opportune moments, such as reflective questions or prompting sketching activities. However, while building SocraBot atop a multimodal Large Language Model (LLM), I realized that existing prototyping and evaluation methods were insufficient for anticipating the nuanced situational complexity and contextual immediacy required.

This chapter presents two novel AI prototyping methods—*Counterfactual Replays* and a *Hybrid Wizard-of-Oz* approach—that address the limitations of existing prototyping techniques for evaluating functional prototypes of real-time, multimodal AI agent

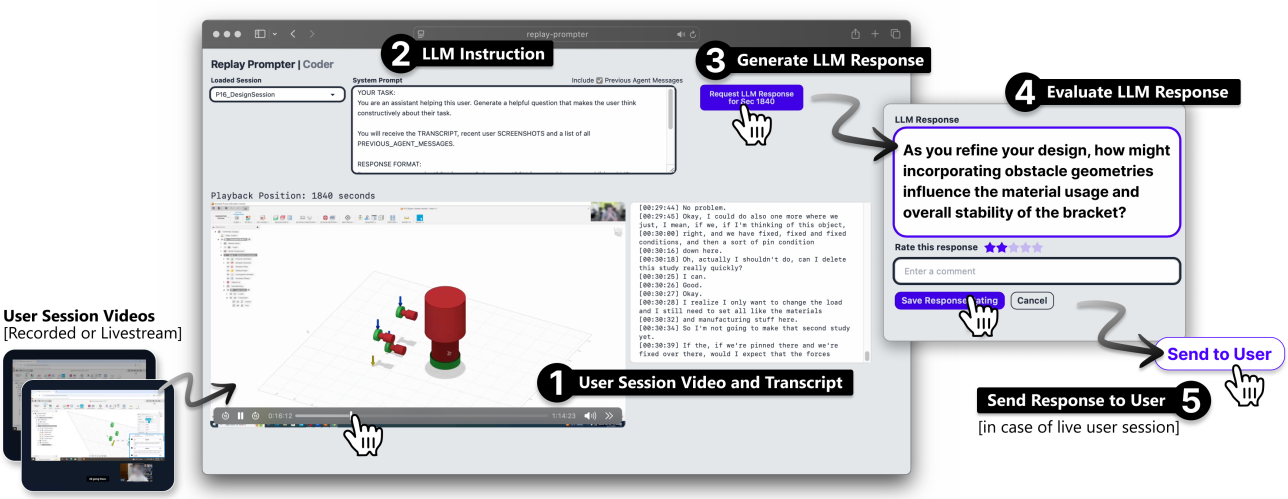


FIGURE 5.1: The Counterfactual Replay Prompt Evaluation Toolkit. The toolkit enables iteratively testing and refining prompt strategies for multimodal real-time agent systems by replaying user session videos. (1) Prompt designers can see recorded or live user session videos and transcripts; (2) optionally refine the model instructions and (3) generate candidate LLM responses based on the prompt and replayed video context. (4) Responses can be evaluated and optionally labeled (rating and comment). (5) In live hybrid Wizard-of-Oz sessions, approved responses can be sent directly to the user's interface. This workflow supports immersive, in-context evaluation of prompt strategies for real-time multimodal support agents.

systems. Following a Research through Design (RtD) approach, I describe and reflect on how I developed these methods to allow me to rapidly test and refine prompt strategies for implementing a real-time multimodal agent support system in an experiential and user-centered way. *Counterfactual Replays* enabled me to replay video recordings of user sessions and interactively test how different model system instructions influenced AI model outputs in various contexts. Similarly, *Hybrid Wizard-of-Oz* allowed real-time steering of GenAI system behavior during live user sessions. By pairing these methods with empirical user test sessions in an iterative design process, I gained a granular and more nuanced understanding of how to refine prompting strategies to improve SocraBot's user experience and support impact. Overall, these methods provide a practical and adaptive approach to prototyping complex multimodal agent behaviors, bridging the gap between automated benchmark testing and experiential in-situ user studies. I conclude by discussing the implications of these approaches for prototyping multimodal GenAI experiences.

5.2 Background

5.2.1 Prototyping Generative AI Applications

The increasing complexity of AI systems, such as ML foundation models, challenges traditional prototyping approaches. In principle, interaction design relies on iterative processes in which designers externalize concepts and ideas to explore alternatives and refine experiences before implementation [212]. In doing so, prototyping interactive user experiences typically follows a sequence of activities, starting with rough, low-fidelity sketches and progressing to functional implementations that closely mirror final products [51, 310]. However, working with ML-based AI systems often challenges these conventional approaches for several key reasons. First, designers often struggle to develop accurate mental models of ML model capabilities and limitations [92, 209], which can lead to a disconnect between designer expectations and actual system behavior. Second, implementing even basic AI functionality traditionally required specialized technical knowledge and substantial development resources—expertise and time that most design teams cannot readily access [390].

In response to these challenges, HCI researchers have coined the concept of "AI as a design material" [92, 389, 391]. Unlike conventional materials, AI exhibits probabilistic behavior (outputs can vary even for identical inputs), emergent properties (e.g., unexpected capabilities outside of trained task scenarios), and rapidly evolving capabilities (frequent model updates that may alter system behavior). As a result, familiar prototyping approaches do not always transfer cleanly. Designers instead need methods that allow them to work with uncertainty and ongoing change, rather than treating both as problems to be removed. Within this context, the rise of **pre-trained foundation models**, including Large Language Models (LLMs), marks a further shift in this landscape. Because designers can now produce relatively sophisticated AI behavior through natural-language instructions, or prompts, LLMs lower many of the technical barriers that once confined substantial AI prototyping to people with specialized expertise [46, 218]. This shift now extends to multimodal Vision Language Models (VLMs), such as GPT-4o [266] and Claude-3 [15], which can process text, images, and video within a single model. Designers can therefore prototype interactions across several modalities without having to assemble or master a separate AI system for each one.

However, on the one hand, prompt-based prototyping has simplified the creation of functional GenAI prototypes. But this ease has also created new design challenges around **prompt and context engineering**, which is the systematic process of crafting

and refining model input instructions and input data to achieve consistent behaviors across diverse contexts [84, 143, 218, 313]. To support this process, techniques such as prompt chaining [386] have emerged that allow breaking complex tasks and instructions into more manageable sequential steps, with outputs from initial prompts serving as inputs to subsequent prompts.

Within this context, a common method for testing and comparing the effectiveness of different prompt strategies is prompt evaluation, which allows systematic assessment of a model's behavior across task-relevant scenarios. In AI research, such evaluation is often conducted using **standardized benchmarks**, such as structured datasets and tasks designed to measure model performance consistently. Examples include GLUE for natural language understanding [369], ImageNet for computer vision [83], or more recently, MMLU for assessing general-purpose reasoning [150]. However, these benchmarks typically require large, carefully curated datasets and are mostly useful for comparing model performance across models or versions. Beyond such benchmarks, reinforcement learning with human feedback (RLHF) has become a widely used approach for aligning model behaviors to human expectations, with human evaluators providing preference data that guides iterative refinement [270].

However, for less established tasks, there are often no existing datasets or benchmarks available. To support prototyping in these novel task situations, several tools have emerged that **support interactive, ad hoc prompt evaluation and comparison**. For example, tools such as ChainForge [16] and EvalLM [177] offer graphical interfaces that allow researchers and practitioners to quickly compare how different prompts affect LLM outputs across novel scenarios. However, these tools are typically limited to text-based tasks and do not extend to multimodal contexts, thereby failing to support temporal and visual data streams.

In addition to these ad hoc evaluation tools, a complementary line of work has focused on **embedding LLMs into interactive "click prototypes."** For example, tools like ProtoAI [340] and PromptInfuser [282] allow to integrate AI functionality directly into prototyping environments such as Figma. These tools allow designers to simulate AI-driven interactions in interface mockups, facilitating easier exploration of how LLM-generated text might appear within a user interface. Yet the focus of these tools also remains largely on text generation, without supporting richer multimodal input or real-time agent behaviors.

In response, to support **multimodal GenAI prototyping**, more recent tools such as Jigsaw [213], InstructPipe [408], and ComfyUI [70] enable the rapid creation and

testing of multimodal pipelines via graphical flow-based interfaces. On the one hand, these environments enable chaining multiple models for handling text, images, and video, thereby significantly lowering the barrier to building multimodal prototypes. However, while effective for constructing and experimenting with pipelines, they do not provide a way to simulate and evaluate proactive, real-time interactions, such as a live support agent following a user's task and intervening at contextually appropriate moments. This gap highlights the need for new prototyping approaches tailored to multimodal, situated, and time-sensitive GenAI experiences.

Inspired by these existing prototyping approaches, this chapter presents novel methods specifically designed for prototyping multimodal real-time AI agent systems that are not supported by existing tools. Concretely, *Counterfactual Replay Prompting* and *Hybrid Wizard-of-Oz* address the unique challenges that emerge when designing GenAI systems that must integrate visual, verbal, and temporal dimensions while maintaining contextually appropriate proactive behaviors.

5.2.2 User-centered Evaluation of AI Applications

Despite recent advances, AI prototyping tools still lack robust integrated user testing, thereby failing to help designers consider the nuanced real-world contexts in which AI interactions unfold. This disconnect creates significant challenges for iterative design processes [142, 306]. While mainstream interface design tools like Figma [111] and Adobe XD [4] excel at creating static interaction flows, they are insufficient to capture the dynamic, context-dependent behaviors of AI systems in real-world use [108, 392]. The resulting prototypes may appear functional in controlled settings but often fail in authentic use.

Similarly, prompt evaluation or model-chaining tools allow rapid testing of prompt and model combinations but typically depend on anticipated use cases ("toy examples") and predefined test data [389], creating a substantial gap between simulated testing and real-world usage. Increasingly, most major AI providers offer evaluation frameworks—for example, OpenAI's Evals [268] or Google's Stax [133]—that allow developers to define benchmark-style evaluation sets to systematically compare model outputs. While effective for well-defined tasks with clear measures for success criteria, these frameworks are unsuitable for evaluating time-based, proactive scenarios where models must track user activity and intervene appropriately. Similarly, ad-hoc testing tools such as ChainForge [16] and EvalLM [177] support systematic text-only prompt evaluation but fall short for multimodal, context-sensitive interactions.

To bridge this gap without developing fully functional AI prototypes, researchers have often used Wizard-of-Oz methods, where humans enact AI behavior [76, 179]. While useful for early feedback, they rarely capture realistic AI limitations and behavior patterns [153]. Recognizing these limitations, researchers have developed *hybrid Wizard-of-Oz* approaches [368, 390] where human facilitators mediate between users and actual AI systems, maintaining testing flexibility while incorporating authentic AI capabilities and constraints. Other approaches, such as *Replay Enactments* [153]—an extension of User Enactments [263]—allow designers to experience different AI behaviors in simulated contexts through data replays [258]. Recent research has also proposed methods for prototyping mobile multimodal AI experiences in situ, enabling designers to directly create and test mobile AI prototypes on actual devices rapidly in real-world contexts [280].

This chapter's research builds upon methods such as data replays and hybrid Wizard-of-Oz techniques, adapting them specifically for emerging multimodal GenAI models to create an experiential, user-centered approach to prompt-based prototyping that grounds evaluation in authentic contextual interactions rather than abstract specifications, synthetic datasets, or made-up test cases generated by design teams or AI itself.

5.3 Methods

To investigate the methodological complexities that emerge when prototyping and evaluating multimodal VLM-powered real-time experiences, I adopted a Research through Design (RtD) approach [132, 410], which recognizes that design knowledge emerges through direct engagement with situated design challenges. Thus, I conducted my methodological explorations in the context of a specific project and goal: implementing *SocraBot*, a multimodal real-time assistant built on top of a prompt-based pre-trained VLM (GPT-4o).

Rather than starting from set hypotheses, I immersed myself in the concrete design challenges presented by this project, allowing insights to emerge organically through cycles of creation and critical reflection. This approach enabled me to generate procedural, conceptual, and pragmatic insights that can inform HCI researchers and practitioners who seek to prototype or study similar multimodal agentic systems, offering transferable design knowledge about methods for creating real-time, proactive, and situated AI support systems [126].

To support methodological transparency in documenting my design process and reasoning, I adopted Bayazit's three-stage framework [278]:

1. ***Knowledge elicitation in an unstructured and unanalyzed form.*** I used Git's version control commit history to document all code and LLM instruction (prompt) changes through granular commits (49 commits in total) for later qualitative data analysis [356]; During all prototyping sessions with users, I recorded the screen, webcam, voice, and system interaction data of participants alongside data from my "wizard" operator interface (screen recording, webcam, system log files).
2. ***Data analysis and interpretation.*** After the project ended, I performed a reflexive thematic analysis [44] of the collected data to identify key design decisions in response to emerging challenges and reflection-in-action that occurred during the design process. I followed an iterative inductive coding process, during which data segments were coded and recursively refined. Specifically, I used ATLAS.ti to analyze Git commits alongside the audio, video, and transcripts of the counterfactual replay and WoZ sessions. These codes were then grouped and synthesized into themes using affinity diagramming on a digital whiteboarding tool (Figjam).
3. ***Finding validation.*** Throughout the project, the research team held regular meetings to review ongoing design challenges, emerging insights, and themes. After completing data collection, I conducted the initial pass of the thematic analysis, after which the entire team revisited the themes collaboratively to cross-check interpretations and ensure agreement among team members.

5.3.1 Prototyping Scenario: Designing and Implementing SocraBot

My goal was to design and implement a proactive cognitive real-time support agent that can assist users in working on a mechanical design task in an AI-assisted 3D CAD workflow (*Designing an engine mounting bracket using Autodesk Fusion 360 Generative Design*). Building on findings from an earlier formative study on metacognitive support agents—which showed that agent-facilitated metacognitive scaffolding can help designers overcome cognitive challenges when working with GenAI tools [129]—I designed *SocraBot* as a situated instantiation of these ideas.

The aim of the *SocraBot* support agent is to follow along with the user's screen actions and verbalizations and to help the user better think through the design task by

(1) **asking reflective questions**, (2) **offering sketching and planning support**, and (3) **providing suggestions for design strategies and software operation**.

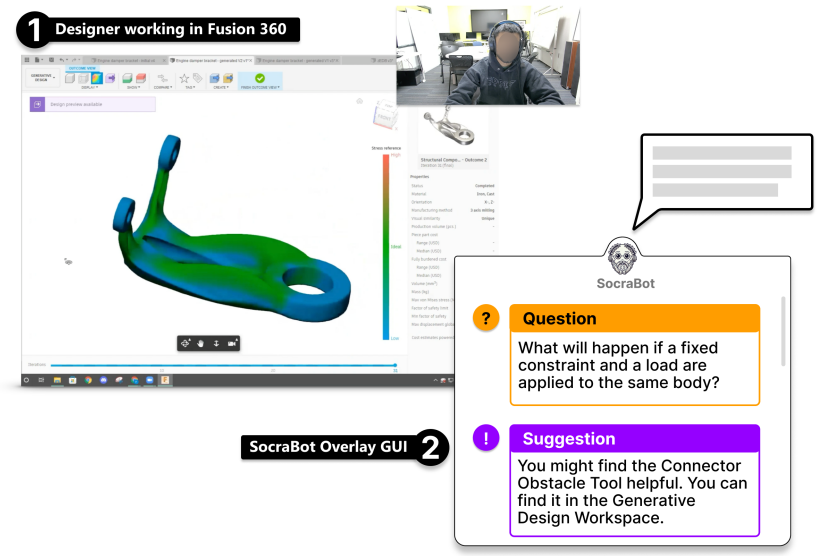


FIGURE 5.2: SocraBot's front-end user interface. (1) While a designer works on a mechanical design task in Autodesk Fusion 360 (2) SocraBot follows the user's verbalizations and screen actions and proactively sends support messages that appear in the overlay GUI window; new messages will also be automatically read as a synthesized voice message to the user.

UX Design and Implementation Goals

SocraBot's design goals are directly informed by the formative Wizard-of-Oz study from the previous Chapter 4, which found that asking (the right) reflective questions and supporting planning and sketching activities has a positive effect on helping designers create more feasible designs compared to non-supported users. Based on insights from this previous study, I derived the following design goals for SocraBot:

1. SocraBot should have **real-time access to the users' screen and speech** and follow designers' verbalizations and screen actions closely;
2. SocraBot should **possess (non-exhaustive) knowledge of additive manufacturing and generative design tasks**;
3. SocraBot should **proactively send messages to the user at opportune moments**. New messages will appear on the user's screen in an overlay GUI window (see Figure 5.2) and will also be automatically read as a synthesized voice message to the user;

4. SocraBot should **identify inconsistencies** between the requirements stated in the user's design brief and the GenAI parameters specified by the designer by comparing the design brief and screen activities (e.g., detecting over/under-constrained load cases, infeasible material combinations or wrong force setup)[1];
5. SocraBot should **pay close attention to the user's task-specific design steps and challenges**, including:
    - Specifying a part's **load cases** (e.g., forces and structural constraints).
    - Modeling appropriate **obstacle geometry** to guide the 3D solver's generation process (e.g., bolt and dampener clearances).
    - Defining **DFM parameters** (e.g., materials and manufacturing options).
    - Supporting users in **evaluating design previews** and **generated outcomes**.
6. SocraBot should **avoid directly instructing users what to do** and instead provide support through reflective questions, suggestions, and hints.

Implementing SocraBot by prompting a VLM

To implement SocraBot, I chose to use a multimodal Vision Language Model (VLM), which can handle text (for the user's continuous speech transcriptions) and image inputs (for the user's screen captures) simultaneously to generate support messages. I chose to use GPT-4o due to its long context window and high benchmark results at the time of the study [266]. Such multimodal VLMs process inputs through a combination of *system* and *user prompts*, where the *system prompt* specifies model behavior, while the *user prompt(s)* dynamically provide text and image inputs as *context*. The model integrates these inputs into a shared embedding space, allowing it to interpret and generate responses that account for both linguistic and visual information in a coherent manner.

Although I already had gained a clear sense of users' support needs from the previous formative study, translating these into a functioning real-time agent system meant defining its behavior through a carefully crafted system prompt and feeding it live speech transcripts and screen context. Achieving the desired situated support behaviors, however, required extensive refinement and testing, making SocraBot's design a

[1]Such requirements could be explicit in nature (e.g., the force the bracket needs to hold) or implicit features, such as part clearances, which were typically not explicitly mentioned in design briefs and considered to be expert domain knowledge.

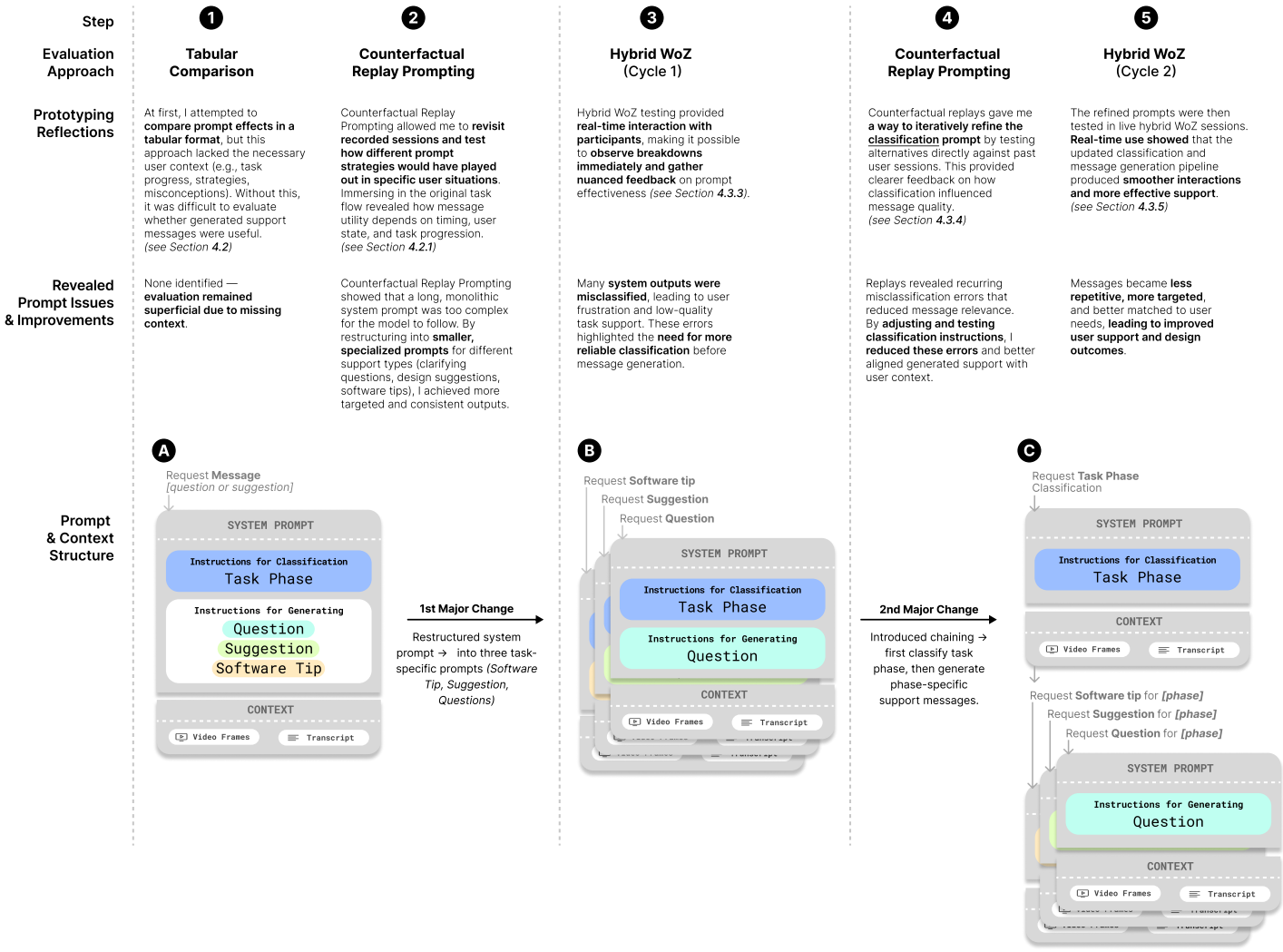


FIGURE 5.3: Process diagram illustrating the sequence of my prototyping steps (1—5), employed evaluation approaches, and reflections, alongside revealed prompt issues and improvements. The bottom row shows the resulting evolution of my prompt strategy over the course of the prototyping process (A—C).

fitting case for my prototyping involving iterative prompt modification and evaluation (see process diagram in Figure 5.3).

5.4 Findings

This section first provides an overview of my prototyping process and then details the prototyping challenges I encountered, along with the methodological solutions that

emerged in response (see Figure 5.3).

5.4.1 Overview

Based on insights from the earlier Wizard-of-Oz study on metacognitive real-time support agents in the context of manufacturing design tasks described in Chapter 4, I had gained a clear sense of users' support needs and how SocraBot could be designed to help them. From the data collected in that earlier study (videos and logs), I first created a collection of video recordings (15 unique user sessions) that served as my dataset for testing prompts and model outputs.

Next, to implement SocraBot, I was faced with the challenge of *"getting the design right"*[51]: Prototyping SocraBot's behavior in detail by prompting a multimodal pretrained VLM. My goal was to create a high-fidelity integrated working prototype [156]. But what were the right workflow and tool(s) for this prototyping process?

I soon realized that prototyping personal proactive real-time agent systems is particularly challenging due to the need to handle multimodal, real-time interactions with dynamic, continuous inputs such as live speech, text, and screen video feeds. Relying on prompt-based VLMs introduced yet another layer of complexity, since their non-deterministic nature required extensive trial-and-error iterations to achieve consistent results. Moreover, unlike many other AI tasks, there was no established benchmark test available to evaluate performance, and existing ad-hoc prototyping tools only supported text-based outputs, offering little guidance for multimodal, time-based scenarios.

At first, I tried a "benchmark"-style evaluation approach and experimented with prompting GPT using anonymized transcripts and screen-capture snapshots from my dataset, and then tabulated the generated outputs for review (see Table 5.1). While this semi-automatic evaluation allowed me to quickly compare prompt variations at scale, it remained difficult to judge their helpfulness within the context of an individual user session. Although many messages appeared well-formulated in isolation, I could not anticipate their practical effectiveness without the unfolding context of each user's design process, making it hard to assess their true value. As a result, I concluded that this approach did not sufficiently capture the temporal, situated nature of users' real-world design work and would not provide reliable feedback on how SocraBot's interventions might actually affect a designer in practice. This led me to ask *how can we rapidly evaluate multimodal prompt strategies using limited data in an immersive and experiential way?* **(RQ5.1)**

TABLE 5.1: Example data from the initial prompt evaluations for comparing messages generated by human wizards during formative user testing and the GPT-generated messages in a tabular format (given the same user session context up until the timestamp).

Session	Timestamp	Test Data (Human-generated)	LLM-generated
...	...	...	...
P03	00:08:41	How will the bracket be connected to other parts, and will these move in any way?	How will the difference in weight between the engine and the damper affect the design of each connection point, and how can you ensure the bracket can handle potential variations in load distribution?
P03	00:09:48	What are the structural constraints of your load cases?	Can you double-check the load application points and explain how the weight of the engine is transferred through the bracket to ensure the load paths are accurately represented in your design setup?
...	...	...	...

Furthermore, going beyond existing previously collected user data, I was curious to find out *how can we evaluate multimodal prompt strategies with live session data in an experiential way?* **(RQ5.2)**

The following sections describe my prototyping process, in which I addressed these questions in detail.

5.4.2 How Can We Rapidly Evaluate Multimodal Prompt Strategies Using *Limited* Data in an Experiential Way? (RQ5.1)

At the beginning of the prototyping phase (Figure 5.3, step 1), I wanted to gain a general feeling for how I could get GPT to generate support messages similar to the kind of messages that the human expert wizards had sent in the formative study. I started by comparing the human wizard messages from the previously recorded user sessions (from the study in Chapter 4) with new support messages generated by GPT. First, I created an **initial *system prompt*** which was composed of several sections with detailed instructions (Figure 5.3, A):

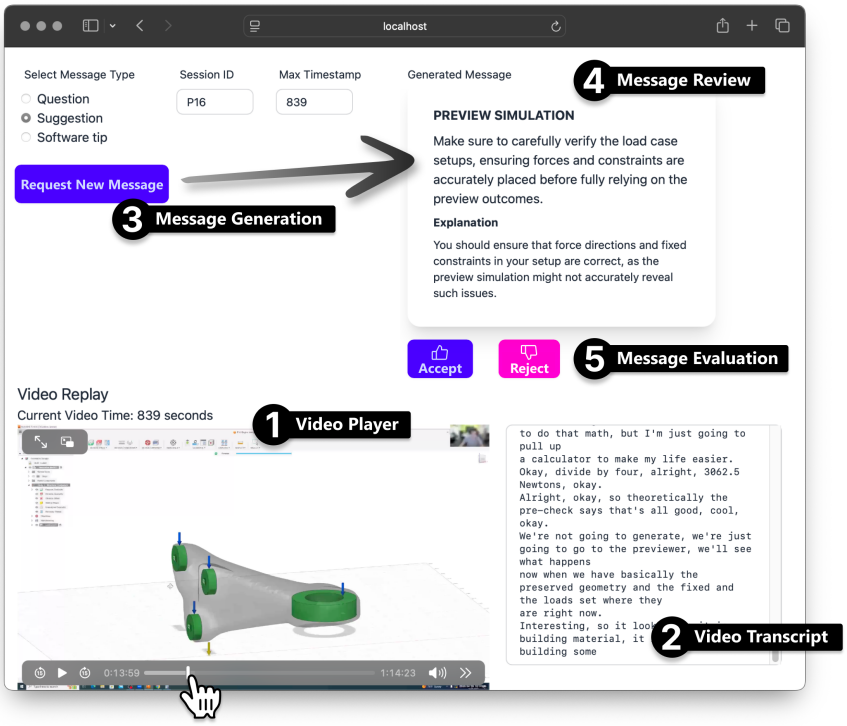


FIGURE 5.4: Interface of the Counterfactual Replay Prompting tool used during prototyping; using the video player (1), one can explore different situations of recorded user sessions along with the synchronized transcription (2), and then generate a model response for a specific situation (3); and review (4) and evaluate (5) the generated message. *Note: This interface differs from the open-source version.*

1. A description of the **general task** *(supporting a designer working in Fusion 360)*,
2. The user's specific **design brief**
3. **Common user challenges** in this task
4. Instructions for **recognizing different design task steps and phases** (e.g., what the user is currently concerned with)
5. Instructions for **generating** different types of **support messages**:
   (a) **Reflective questions**
   (b) **Suggestions** (design-related or software operation-related)

After constructing the system prompt, I then created a pipeline that would allow me to compare the resulting model responses with human support messages from moments in the recorded user sessions. The pipeline consisted of the following steps: First,

I stored all session verbalization transcripts and screen recordings in a database as individual utterances and video frames along with their timecode (Table 5.1). Next, to get the context of a session for a specific time, I created a script that would dynamically reassemble the session's utterances and video frames up until that time. Lastly, to generate new messages with an LLM and to compare these with the existing human messages, I constructed an LLM inference call for each human wizard message by providing the session's user context up until that message's timestamp (transcript and video frames) and stored the LLM's response in a table (Table 5.1).

Developing a More Immersive and Flexible Prompt Evaluation Approach

Given these encountered limitations, I was confronted with the question of how I could prototype GenAI prompting strategies for SocraBot using my *existing* data (user session recordings) in a more experiential and flexible way. My solution to this challenge was a new form of video-based prompt testing, namely *counterfactual replay prompting*. Inspired by methods like replay enactments [153], I built a tool that allowed me to play back videos of recorded user task sessions and to trigger VLM message generation at any moment in the video by providing the video's current context (Figure 5.4). Generated messages then directly appeared in the tool above the video. This allowed me to be more immersed in the user sessions and have a better understanding of the user situation when evaluating generated messages, which gave me more confidence and flexibility for testing how the LLM prompts—and thus SocraBot—would play out in different situations.

Technically, I realized the *video replay prompting* interface as a web application with an integrated video player and UI elements to trigger message generation. In its interface, I also included a UI element to specify custom parameters, for example, to flexibly generate different kinds of support messages (such as *questions* or *suggestions*). I could also rate the generated messages directly in the interface using accept (thumbs up) or reject (thumbs down) buttons, with these ratings stored in a database for later analysis. The front end communicated with a back end, which handled the VLM inference requests and also included the database to store evaluation ratings from the front end. For every requested message, the backend *dynamically created the model instruction's context* from the session's user transcript and video frames, as well as the session's support message history and requested message type (*reflective question* or *suggestion*). To change the agent's behavior, I could directly adjust the system prompts and iteratively refine them in the backend prompt text file.

My prompting strategy was intended to generate *reflective questions* and *suggestions* at appropriate times, and to *automatically infer* whether to generate a suggestion for a *design strategy* or for *software assistance*. Using my setup, I was able to test how well this prompting strategy achieved those goals by observing resulting system behavior in the context of real-time replays of various recorded user interactions. Employing the *Counterfactual Video Replay* tool, I immersed myself in various user sessions to test SocraBot's support behavior:

Using the interface, I stepped through the recordings, pausing at moments where a user encountered a challenge or made a design move, and requested questions or suggestions from SocraBot. I compared the generated output to what a human facilitator might have said and noted mismatches or gaps. When responses seemed off-target, I revised the backend system prompt—adjusting wording, structure, or emphasis—and immediately re-ran the same moment to see whether the change improved the output. I also used the interface's rating functionality to mark outputs as acceptable or not, creating a lightweight record of my judgments stored in the database. Through this cycle of watching, generating, editing, rating, and retesting, I rapidly explored how different prompt formulations shaped the kinds of support the system produced.

While the question generation generally performed well, I identified limitations in the suggestion functionality. Specifically, when requesting *suggestions*, the model consistently returned only design strategy advice but failed to produce any software-related tips. This revealed a significant gap in the system's ability to differentiate between types of support content. Given these observations, **I concluded that having a single system prompt might be too long and complex for GPT to follow**[2].

I therefore **decided to adjust the prompting strategy and restructure the system prompt into three shorter and more specialized prompts** for generating (1) *reflective questions*, (2) *design suggestions*, and (3) *software tips separately* (see Figure 5.3, B). In alignment with these prompt changes, I also updated the UI of the video replay front-end to allow me to specify one of the three support categories when triggering a new message generation.

After refining my approach, I re-tested the new prompt strategy using counterfactual video replay prompting. In these sessions, I evaluated the outputs by comparing them against my expectations for each category and used the built-in rating mechanism to record my judgments on their appropriateness and usefulness. Overall, I found that breaking down the initial single prompt into smaller, specialized instructions led

[2]This limitation is not unique to my work; difficulties in steering LLMs with long or complex prompts have been widely recognized (see [206, 217]).

to more controlled, targeted, and helpful support message generation—aligning much closer with the intended behavior.

5.4.3 How Can We Evaluate Multimodal Prompt Strategies *With Live Session Data* in an Experiential Way? (RQ5.2)

Using *Counterfactual Video Replay Prompting* allowed me to better understand how to refine the prompting strategy to make SocraBot more aligned with the intended support behavior throughout differing user contexts by utilizing previously recorded user sessions. However, I soon reached limitations inherent to this counterfactual approach: Users' actions remained fixed in recorded data, making it impossible to see how people would react to AI-generated messages. While I could test how well SocraBot would be able to create support messages catering to users' changing contexts, I could not test how these messages would impact new users. This raised the question about how the system would behave in new situations and how I could keep testing and refining the prompting strategy, incorporating live user data.

Moving to a Hybrid Wizard-of-Oz (Human+LLM) Approach

As a solution to these limitations, I came up with an adapted *Hybrid Wizard-of-Oz* method that would allow me to prototype prompts for SocraBot's behavior in a live setting by supporting users working remotely on a task in real-time. For multiple reasons, I chose a Wizard-of-Oz setup rather than full automation. First, automatically inferring the right timing for proactive interventions remains an unsolved challenge, making it difficult to evaluate how well prompts would play out in real time without human oversight. Second, keeping a human in the loop allowed the wizard to actively observe, learn, and adjust—both by deciding when to trigger support and by overriding system outputs when necessary. This setup ensured that I could test prompt strategies in realistic user contexts while still maintaining the flexibility to refine them in situ. Lastly, immersing myself in the user sessions and steering the VLM in real time gave me direct experiential insight into both the strengths and shortcomings of the model and prompting strategy, deepening my understanding of how different design choices shaped user experience.

To incorporate live data, I first expanded my pipeline by creating a user-facing front-end app for SocraBot (*SocraBot Message App*). This software[3] (installed on a user's

[3]based on Electron, React, Daily.co, and Deepgram speech-to-text

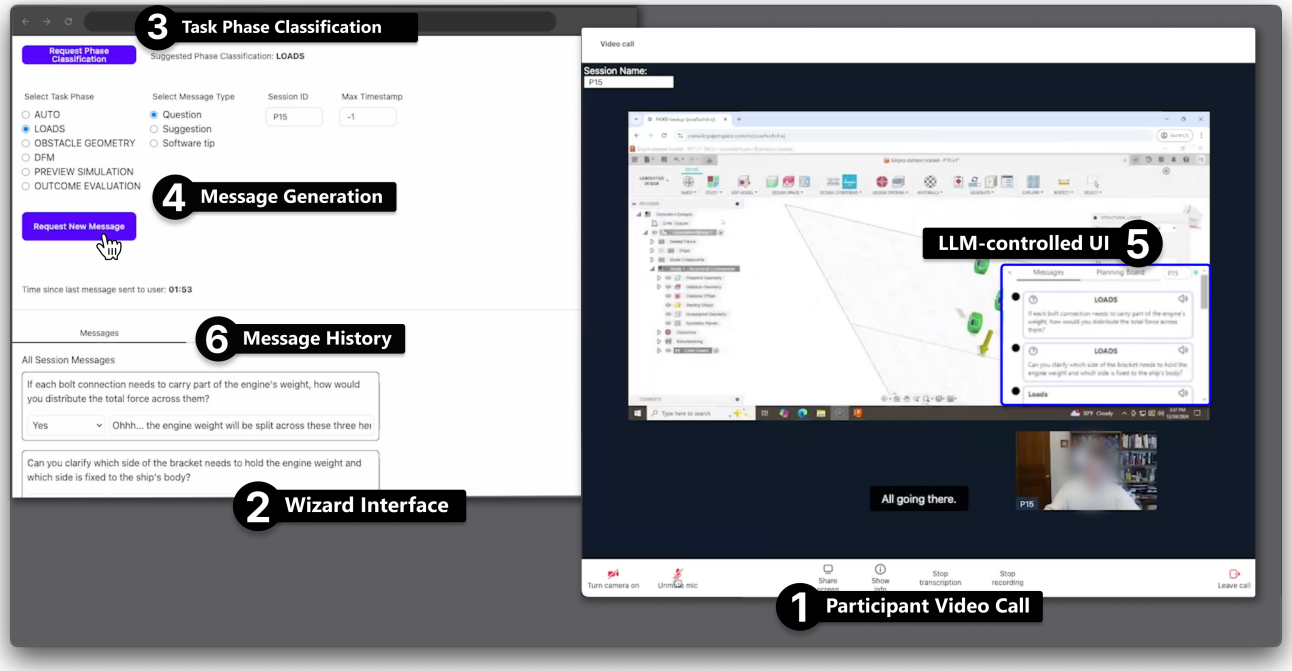


FIGURE 5.5: The Hybrid Wizard-of-Oz setup; (1) Wizards can follow along with user actions in the video call app; (2) using the wizard interface, they can trigger various model inference calls such as (3) task phase classification and (4) message generation; (5) wizards can optionally send generated messages to the user's SocraBot interface; (6) all generated messages can be reviewed and rated in the message history. *Note: This interface differs from the open-source version.*

system) can playback and display SocraBot's messages in a resizable, always-on-top window located on the bottom right of the user's screen (see Figure 5.2 and 5.5). The app also captures the user's screen video feed and verbalization transcript and transmits these in real-time to a remote web server and database for storage.

Hybrid Wizard-of-Oz User Study Setup

To test how well the existing prompting strategy supports *new* users working on a design task (new data samples), I conducted a user study utilizing the *Hybrid Wizard-of-Oz* method. Overall, the study design replicated the setup of the formative study described in Chapter 4, allowing for a direct comparison. To examine how the prototype's support messages influenced users' design processes, I also assessed the quality of their final design outcomes (see *Measures and Analysis* below). To establish a baseline, I compared these outcomes against data from the original study (which used an

identical between-subjects design).

Participants I recruited 10 designers (aged 20 to 43 *(M = 24.7, SD = 7.1))* with mechanical engineering backgrounds from engineering departments of North American universities and through the Upwork freelance hiring platform[4] (see Appendix Table C.1). Participants had between one and ten years of Mechanical Design experience and between zero to ten years of industry experience, as determined via a screening questionnaire. All participants had at least two years of experience using CAD and Autodesk Fusion 360 but no experience working with the Generative Design extension. I recruited participants familiar with Fusion 360 so that they could focus on learning to work with the AI-driven Generative Design feature rather than learning the CAD tool's user interface. Participants included a mix of undergraduates, graduate students, and professional engineers. Before the study, all participants signed a consent form approved by my institution's IRB. Participants were compensated 20 USD per hour.

Procedure Participants were randomly assigned to either the SocraBot Hybrid WoZ *First Cycle* or *Second Cycle* groups. Overall, the study sessions followed the same procedure as the formative study in the previous chapter (see 4.4 for further details).

Hybrid Wizard-of-Oz Setup In all sessions, participants and I (acting as the wizard operator) were located remotely and connected via video conferencing software. Participants accessed a cloud-based computer[5] preconfigured with Fusion 360 and the *SocraBot Message App* (the user-facing interface). While observing participants' screen activity and verbalizations in real time, I operated the hybrid Wizard-of-Oz interface to generate and deliver support messages.

Measures and Analysis To gain insight into SocraBot's impact on the design process, I evaluated the design outcomes and analyzed the recorded think-aloud session videos using video interaction analysis [33]. I collected the following data:

- Video, screen, and audio recordings with machine-generated transcripts of the agent-supported think-aloud design sessions (including the wizard's screen)
- 3D designs created during the design sessions
- Log files with timestamps of all agent messages

[4] http://www.upwork.com
[5] using Paperspace

TABLE 5.2: Summary of participants' outcome design scores across five criteria (checkmarks) and process statistics by support group (see Appendix C.2.2 for analysis method details). Note: the *"Without SocraBot"* group is identical to the unsupported group from the study in Chapter 4.

	Without SocraBot						SocraBot **Hybrid WoZ FIRST Cycle**						SocraBot **Hybrid WoZ SECOND Cycle**					
Participant	**N1**	**N2**	**N3**	**N4**	**N5**		**A1**	**A2**	**A3**	**A4**	**A5**		**B1**	**B2**	**B3**	**B4**	**B5**	
Passing Structural Analysis (FEA)										✓			✓	✓	✓	✓	✓	
Correct Load Setup										✓			✓		✓	✓	✓	
Mass Optimized										✓			✓	✓	✓	✓	✓	
Feasible Fastener Clearances		✓	✓				✓	✓	✓				✓	✓		✓	✓	
Feasible package size	✓			✓	✓	*AVG*	✓				✓	*AVG*	✓	✓	✓		✓	*AVG*
Outcome Design Score	1	1	1	1	1	**1**	2	1	1	3	1	**1.6**	5	4	4	4	5	**4.4**
# Messages Causing New Considerations (Impact)							2	4	1	2	1	**2.0**	1	4	2	5	3	**3.0**
# Agent Messages Sent							19	20	14	13	19	**17**	14	12	8	18	8	**12**
Duration (min)	53	37	63	83	59	**59.0**	63	58	52	64	58	**59.0**	60	73	65	48	52	**59.6**

I applied the following measures:

- ***Number of impactful messages:*** I used *video interaction analysis* [33] of the think-aloud recordings to understand how agent support impacted participants' design process (see Appendix C.2.2 for more details).
- ***Design outcome score:*** I evaluated the design outcome feasibility by checking the final engine brackets against the requirements in the design brief and rating them across five criteria: *(1) structural soundness*, *(2) feasible load case setup*, *(3) optimized mass*, *(4) feasible clearance*, and *(5) volume within maximum dimensions* (see Appendix C.2.2 for more details).

Overall, the measures and analysis were identical to the study in Chapter 4 (see 4.4).

Diagnosing Additional Prompt Limitations through Hybrid WoZ (First Cycle)

The first cycle of Hybrid Wizard-of-Oz (WoZ) sessions demonstrated the value of this method as a rapid way to assess my latest prompting strategy in a live context. By letting the wizard trigger or withhold model messages in real time, I was able to observe how the system responded to unfolding user behavior and to capture immediate evidence of positive impacts and shortcomings. Equally important, each denied message was annotated with a short explanation, creating a record of where and why the

model's outputs were misaligned. This hands-on process gave me a situated understanding of how difficult it was for the model to keep track of task context and where my prompt structure was breaking down.

Through this methodological lens, the first iteration revealed a recurring issue: the model often misclassified the user's current *task phase*, producing irrelevant or distracting support. Since relevance depended heavily on whether the user was specifying structural loads, defining obstacle geometry, or evaluating outcomes, these misclassifications undermined the usefulness of the generated messages. The hybrid WoZ setup made such errors visible in situ, highlighting the need to separate classification from message generation in my prompting pipeline (Figure 5.3, C).

At the level of user outcomes, SocraBot's support had only a modest impact. In supported sessions, impactful messages (i.e., those triggering observable new considerations as in 4.4) ranged from one to five per session ($M = 3.0, SD = 1.6$). Messages primarily helped people improve the part's obstacle geometry by considering additional factors such as bolt and tool clearances (Table 5.2). Design quality improved slightly for SocraBot-supported users compared to non-supported users ($M = 1.6$, $SD = 0.9$ vs. $M = 1.0$, $SD = 0.0$), but many participants in both groups still failed to specify the bracket's structural loads and obstacle geometry adequately, resulting in parts that did not meet all design criteria.

In summary, although the system's performance was weaker than hoped, the first Hybrid WoZ cycle provided critical methodological insights: the live wizarding exposed misclassifications that remained invisible in such detail in prior offline testing. These lessons directly informed the next iteration, where I began separating task-phase classification and message generation into distinct prompts.

Refining Prompt Structure by Separating Classification from Generation

Insights from the first Hybrid WoZ cycle underscored that misclassifications of user activity were a central bottleneck. Methodologically, this highlighted the need to disentangle different reasoning steps within the prompting pipeline. In response, I adopted a prompt chaining approach [386], restructuring the system into two stages: one prompt dedicated to *task-phase classification* and another for *message generation* (Figure 5.3, C). This separation allowed me to assess the model's classification accuracy independently of the usefulness of its generated messages, and gave the wizard further points of intervention.

In practice, the classification prompt first inferred whether the user was in the task phase of, for example, specifying constraints, defining geometry, or evaluating outcomes. The message-generation prompt then produced a support message tailored to that phase and to the chosen message type (question, suggestion, or software tip). The wizard could inspect each classification, override it when it did not fit the observed activity, and only then request a message. This setup retained human control while providing a clearer view into the model's reasoning process.

Before re-running Hybrid WoZ sessions, I used *Counterfactual Video Replay Prompting* to refine the classification instructions (Figure 5.3, step 4). Replays of prior recordings revealed recurring errors, which I addressed by expanding the classification prompt with more explicit guidance and mistake–correcting patterns. For example, commit `13b9d5a` [*Task phase classification: More specific explanation of challenges and how to recognize these*] (Appendix Table C.2) introduced detailed criteria for recognizing challenges in load specification, obstacle geometry, manufacturability, simulation, and outcome evaluation. These incremental changes, documented through Git commits, allowed me to track how prompt revisions improved the model's ability to classify phases correctly.

With this refined pipeline in place—classification first, message generation second, and wizard oversight at each step—I proceeded to a second cycle of Hybrid WoZ sessions to validate the new setup in real time.

Validating Improvements through a Second Hybrid WoZ Cycle

In the second Hybrid Wizard-of-Oz cycle, I further tested the revised prompting pipeline in additional live sessions. Methodologically, I found that, separating task-phase classification from message generation proved valuable: it surfaced the model's reasoning more transparently, allowed the wizard to intervene at each stage, and enabled me to evaluate classification accuracy and message usefulness independently. In practice, the wizard overrode roughly half of the classifications (by choosing a different task phase for message generation than the one suggested by the model), but this was a relatively lightweight intervention with outsized effects: once an incorrect task phase was corrected by the wizard, the downstream message was still generated by the model. This direct control prevented missteps from propagating, created a clear record of where the model continued to struggle, and highlighted how stabilizing classification greatly improved the usefulness of the AI-generated messages.

Compared to the first Hybrid WoZ cycle, the generated support messages were noticeably less repetitive and more targeted. By exposing classification results in the interface and allowing real-time overrides, the generated messages fit more closely with unfolding user activity. Overall, this combination of transparency and control not only improved the immediate alignment of support messages but also provided valuable insights for evaluating my classification and message generation prompts in isolation. This made it clear that message generation produced satisfactory results (increased positive impact on users) once the correct task phase was specified.

At the outcome level, designers in this cycle achieved significantly higher-quality results (see Appendix C.2.2 for analysis). All participants created brackets that passed structural analysis, and most designs also met clearance and space constraints. The average outcome score rose to $M = 4.4$ ($SD = 0.5$), compared to $M = 1.6$ ($SD = 0.9$) in the first cycle (Table 5.2). Support messages guided designers better in specifying relevant inputs for the generative solver, resulting in more robust designs.

In sum, the second Hybrid WoZ cycle showed that the refined pipeline offered both methodological and practical benefits. It provided a clear framework for diagnosing and fixing errors, which enabled more reliable delivery of useful support messages and resulted in measurable improvements in design outcomes. While task-phase classification still required further refinement, this stage of prototyping demonstrated that the overall system could reliably support users and provided me with a strong foundation for advancing SocraBot toward a more effective and usable real-time support agent.

5.4.4 Summary of Prototyping Methods and Insights

I encountered several prototyping challenges and addressed them through two complementary methods:
First, in prototyping a multimodal real-time support agent on top of a pretrained VLM, I faced the dual **challenge of having only limited user data for evaluation while also lacking established evaluation benchmarks.** I then initially wanted to **compare prompt effects in a tabular format** (inspired by benchmark-style approaches), **but was unable to assess the usefulness of model outputs** (proactive contextual support messages) due to missing the unfolding task context—such as what the user had already done and tried in the task session, what strategies they were pursuing, or what misconceptions they held.

To address this, ***Counterfactual Video Replay Prompting*** **allowed me to immerse myself in recorded user scenarios and flexibly test how different prompt**

strategies would play out across varying user contexts. This revealed that a single, lengthy system prompt was too complex for the model to follow (which is generally a known limitation of language models, and figuring out *when* and *how* to split up prompts is part of the iterative trial-and-error process). **I therefore restructured the system prompt into smaller, specialized prompts** for different support types *(reflective questions, design suggestions, and software tips)*, which produced more controlled and targeted support behavior.

However, counterfactual replay prompting **did not allow me to explore system behavior in entirely new user situations**, since the actions in the recordings were fixed. To address this limitation, I **introduced an adapted *Hybrid Wizard-of-Oz* method** that enabled evaluation of prompt strategies during live user sessions. Here, real-time interaction revealed frequent misclassifications of user activities by the model, which led to irrelevant or frustrating support messages. Based on these observations, I **split the prompting pipeline into two stages—task-phase classification and message generation—and added mechanisms for manual overrides**, giving the facilitator more control to steer and document the model's behavior during sessions.

I then cycled back to counterfactual replay prompting, using it to refine the classification prompt and test corrections against prior recordings. This helped me diagnose and further reduce recurring misclassification errors. A subsequent round of hybrid Wizard-of-Oz sessions validated these refinements in real time: messages became less repetitive, more targeted, and better aligned with user needs, which led to improved design outcomes. In this way, the two methods complemented one another—counterfactual replays supported rapid, immersive testing of prompt variations, while hybrid Wizard-of-Oz sessions provided experiential validation in live contexts. Together, they enabled me to iteratively shape the prompting strategy as a foundation for a more automated version of SocraBot.

5.5 The Counterfactual Replay Prompt Evaluation Toolkit

Beyond methodological insights, this research contributes an open-source toolkit[6] that supports one key element of my approach: counterfactual replay prompting. For this, I refactored the internal tools developed during the SocraBot prototyping process into a reusable, modular, and expandable prototyping application. The resulting self-contained web application enables HCI researchers and practitioners to rapidly test multimodal,

[6] `https://github.com/Augmented-Design-Capability-Studio/replay-prompt-eval`

prompt-based GenAI behaviors by replaying recorded user sessions and generating model outputs at arbitrary points in time (see Figure 5.1 and the video figure in the Supplementary Material). In addition to generating and evaluating model outputs for different system prompts, the application also includes a *coding view* that lets users label and annotate generated messages. The backend stores all human-generated labels (metadata such as message ratings and comments) in a JSON file, allowing for easy export for further qualitative or quantitative analysis in other software. To use the toolkit in live hybrid WoZ settings beyond counterfactual replay, users can extend the backend to connect it to their own applications (e.g., a video call application's live video and transcription stream). Further details can be found in the Appendix C.2.

5.6 Discussion

This chapter documents my firsthand experience of implementing SocraBot, a GenAI-based application built on top of a multimodal VLM, revealing the complex challenges that emerge during this prototyping process. In response to these challenges, I developed a novel prototyping and evaluation pipeline consisting of *Counterfactual Replay Prompting* and a *Hybrid Wizard-of-Oz* approach, which enabled me to improve the GenAI-based system by observing how different prompting strategies affected system behavior across varying situations—insights that traditional evaluation methods would have failed to surface at this early prototyping stage.

In the following sections, I discuss the implications of these novel approaches for prototyping multimodal real-time GenAI agent experiences beyond this specific SocraBot case. I examine how the introduced methods address complexity challenges in prompt engineering, complement existing prototyping methods, and support a more designerly approach to creating GenAI-based systems that meet user needs in complex, real-world contexts.

5.6.1 Taming Complexity with Iterative Prompt Decomposition Through Interactive Human-in-the-loop Steering and Evaluation

The reported process of implementing SocraBot highlights how evaluating GenAI model performance for novel multimodal tasks without established test data and benchmarks feels like navigating blindfolded. In these situations, practitioners typically resort to

iterative trial-and-error testing when working with prompt-based models [402]. Currently existing ad-hoc prompt evaluation tools largely focus on text generation [16, 177], falling short in handling the multimodal and time-based data required to evaluate proactive real-time systems like SocraBot. This limitation becomes more evident when working with emerging multimodal capabilities that combine visual, verbal, and contextual interactions. Overall, this work demonstrates how interactive and experiential evaluation methods such as *Counterfactual Replay Prompting* and *Hybrid Wizard-of-Oz* bridge the gap between blind trial-and-error and rigid automated benchmark testing.

These approaches fostered deeper problem understanding early in the development process by letting me observe how model outputs played out in realistic task contexts. Counterfactual replays, for example, revealed how suggestions that looked linguistically sound in a table often failed when judged against the unfolding user activity (Section 5.4.2). Hybrid Wizard-of-Oz sessions, in turn, exposed how misclassifications of user context (such as their current task phase) could cascade into ineffective support behaviors,[7] directly motivating my restructuring of the prompt architecture into smaller, specialized modules (Sections 5.4.3–5.4.3).

By **offering a gradual path toward automation**, these methods allow practitioners to begin with highly interactive and situated evaluation before transitioning to more automated testing as their understanding improves. Rather than replacing benchmark testing, the methods presented in this chapter serve as a crucial bridge for emerging models and task domains where test data is scarce and effective prompting strategies remain unclear. Such methods also provide experiential insights into model behavior that help shape which *kinds of data* should be collected for future, more systematic evaluation.

As I have illustrated throughout this chapter, these methods prove especially valuable for taming complexity during the early stages of developing multimodal real-time agent systems. They enabled me to decompose complex prompt structures into smaller units while still maintaining a holistic view of how these modules function together in a pipeline. For instance, replaying full sessions revealed a skewed distribution of message types. This insight, visible only when prompts were tested against unfolding user activity, guided me to split these categories into separate prompt modules and iteratively evaluate their balance within the pipeline (Section 5.4.3).

[7] Similar cascading errors have been observed in chain-of-thought and reasoning pipelines, where early missteps compound into larger failures [397, 407].

5.6.2 Supporting Designerly and User-centered Prompt Engineering Approaches

The way I employed these prototyping methods aimed to support a *holistic practice* of prompt engineering. Designers must attend not only to users—whose goals, skills, and practices vary widely—but also to stochastic ML models, whose behavior is inconsistent and requires careful probing. User-centered design, in this light, becomes a mediating practice between two sources of uncertainty: people and models. The designer's role is to continually negotiate this space, refining prompts so that model behavior aligns with evolving user workflows.

Immersing myself in concrete user situations—through video replay or during hybrid Wizard of Oz sessions—allowed me to develop a comprehensive, contextual understanding of how to structure and chain prompt instructions without requiring massive upfront data collection. Though time-intensive, the hybrid Wizard of Oz sessions provided crucial insights into model behavior across various situations (see Sections 5.4.3 and 5.4.3). This helped me develop a tacit understanding of how the model responded to different inputs, which directly informed my adjustments to the prompt structure.

A valuable insight emerged during hybrid Wizard of Oz testing when I could observe and override the model's misclassifications in real time (see 5.4.3). Seeing when the model made incorrect classifications in the first chain element (classifying the task phase) and then overriding these decisions for subsequent message generation provided an effective mechanism for assessing individual prompt effects and their interplay. Iteratively moving back and forth between counterfactual replays and hybrid WoZ cycles also proved powerful: replay sessions allowed me to capture more data, experiment rapidly with new prompt ideas, and refine them in a low-stakes setting, while hybrid WoZ sessions tested these refinements in live interaction. This cyclical process allowed me to bootstrap the agent while being embedded in the interaction itself, offering a unique form of experiential feedback that is rarely available through conventional evaluation.

Overall, the proposed value of these experiential prompting methods lies in their support for reflective practice in prompt engineering. Unlike traditional evaluation approaches focused on technical metrics, these methods emphasize the practitioner's developing relationship with the model, revealing how its behaviors respond to various inputs and contexts. This approach recognizes that effective prompt engineering involves both technical skill and "designerly" intuition developed through immersive observation and interaction. By supporting this reflective dialogue between designer

and system, these methods help practitioners develop the tacit knowledge needed to create more responsive, contextually aware GenAI experiences that genuinely address user needs rather than merely optimizing for technical performance metrics.

Such intuition-building prototyping practice is particularly important in the context of cognitive assistant design, where performance metrics are difficult to obtain: support unfolds over long-running tasks, and clear signals of effectiveness may only appear after extended interaction. Unlike question-answering systems, which provide measurable feedback in every exchange, assistants for complex design work offer fewer and slower signals, making experiential, in-situ evaluation a crucial part of the prototyping process.

5.6.3 Future Synergies with Existing Methods and Tools

The prototyping process and toolkit described in this chapter build upon existing methods such as data replays, hybrid Wizard of Oz, and prompt-evaluation frameworks. While surfacing insights about the specific prototyping process within one project and domain (design support agent), my process reflection also revealed exciting opportunities for integration with existing methods and other task domains.

For example, a promising avenue is to incorporate Counterfactual Video Replay Prompting into existing prompt evaluation tools. Current evaluation frameworks (e.g., [16, 177]) provide rich interfaces for text-based prompt assessment but lack support for multimodal, time-based data. By integrating video-based replay capabilities, these tools could be expanded to evaluate multimodal GenAI experiences. Similarly, model pipeline editors (e.g., [70, 213]) could benefit from incorporating replay-based evaluation methods, enabling practitioners to test how different model pipelines respond to authentic multimodal user scenarios rather than hypothetical examples. This integration would provide more nuanced insights into how prompt chains perform across varying contexts and user behaviors. The Hybrid Wizard-of-Oz approach could be extended beyond video-based interactions to encompass emerging spatial computing platforms such as virtual and augmented reality. As multimodal GenAI models increasingly power spatial immersive experiences, these methods could therefore help designers evaluate and refine prompt strategies for novel spatial interaction tasks where contextual understanding becomes even more complex.

Another promising opportunity lies in combining evaluation with data labeling.

While the methods presented in this chapter primarily focus on refining prompt strategies through observation and intervention, they could also be extended to capture human interventions in steering model behavior. These intervention points might serve as valuable training data for improving the model performance itself. By documenting when and why human overrides occur, we could build feedback loops that allow GenAI models to learn from these corrections over time, gradually reducing the need for human mediation. In addition, these processes could be combined with reflective prompt generation strategies, such as GEPA [6], which enable AI systems to iteratively refine their own prompts through natural language reflection. While such reflective optimization holds promise for reducing human effort in prompt engineering, the findings in this chapter highlight the continued importance of designer-led evaluation to ensure that evolving prompts remain aligned with real user needs and contextual constraints.

In conclusion, all these potential synergies illustrate how methods such as Counterfactual Replay Prompting and Hybrid Wizard of Oz can complement and enhance existing prototyping approaches, creating more comprehensive evaluation ecosystems that address the unique challenges of multimodal, context-sensitive, real-time GenAI applications across diverse interaction paradigms and task domains. The open-source toolkit I developed provides a practical foundation for such integrations: it can be extended with modules for integrating replay-based evaluation and human-in-the-loop live steering and labeling into other prompt evaluation pipelines, and adapted to domains beyond cognitive design assistants. In doing so, it offers a bridge between research prototypes and broader community use, supporting both academic experimentation and practitioner-driven tool development.

5.6.4 Limitations

This chapter's core contribution is a case-study demonstration of how the prototyping methods introduced here (*Counterfactual Replays* and *Hybrid WoZ*) enabled me to iteratively refine SocraBot's system behavior; a full evaluation of SocraBot's effectiveness as a support system lies outside the scope of this work and would require a dedicated study. While the methods described in this work offer promising advantages for prototyping multimodal GenAI experiences, I highlight the following limitations. First, this research centers on a single prototyping case in a specific domain, and further research is required to validate its generalizability across other GenAI applications. Broader validation across different contexts and by independent evaluators would further strengthen confidence in these methods—an opportunity I aim to support through

the open-source toolkit, which enables others to replicate, extend, and critique my workflows.

Second, these approaches face scalability challenges due to their reliance on human oversight, making large-scale simultaneous evaluations impractical. However, the methods' primary aim is not scale but deep contextual insight during early-stage prompt development.

Third, the Hybrid WoZ approach introduces potential wizard bias, as human judgments on *when* to intervene can influence outcomes and prompt iterations. Yet such intervention is not merely a limitation but also a vital part of the learning and refinement process in early-stage prototyping—consistent with the "fake it to make it" tradition in HCI, where human scaffolding helps probe what future automation should deliver [336]. The remaining challenge lies in automating intervention timing: inferring when proactive support should occur is still an open problem in HCI, and remains particularly difficult to standardize or automate.

Reproducibility is also a concern; human involvement (especially when overriding LLM outputs) makes it challenging to recreate consistent evaluation conditions, complicating the systematic comparison of prompt strategies across evaluators. These challenges can be mitigated through precautions such as clearly documenting decision criteria, standardizing logging practices, and, in team settings, conducting cross-rater discussions. Ultimately, these methods are intended as tools for experiential understanding, which ideally will lay the groundwork for more systematic and automated testing in later stages. Finally, Counterfactual Replay Prompting depends on pre-recorded sessions, which may not reflect the full range of user behavior, potentially leading to prompt strategies tuned to narrow interaction patterns rather than real-world variability.

Chapter 6

Intent Tagging: Enabling Thoughtful Co-Creation with LLMs

This chapter was adapted from my published paper:

> Frederic Gmeiner, Nicolai Marquardt, Michael Bentley, Hugo Romat, Michel Pahud, David Brown, Asta Roseway, Nikolas Martelaro, Kenneth Holstein, Ken Hinckley, and Nathalie Riche. 2025. "Intent Tagging: Exploring Micro-Prompting Interactions for Supporting Granular Human-GenAI Co-Creation Workflows." *In CHI Conference on Human Factors in Computing Systems (CHI '25), April 26–May 01, 2025, Yokohama, Japan. ACM, New York, NY, USA.*

6.1 Overview

In the previous chapters, I describe how I explored thoughtful co-creation interactions for supporting engineering designers' thinking and metacognition through voice-based agents in geometry-based generative design tools. This chapter describes the expansion of my exploration of metacognitive support interactions beyond voice-based agents and engineering design tasks across other task domains and interaction modalities. Motivated by further exploring metacognitive support interactions beyond voice-based agents and engineering design tasks, I developed a novel input strategy for controlling Large Language Models (LLMs) called *Intent Tagging*—an interaction technique for AI-assisted rich content creation, such as slide presentation authoring. *Intent Tagging* allows users to iteratively articulate and refine their intent through granular, adaptive graphical UI elements. To further explore the feasibility of this technique, I built *IntentTagger*, a web-based interactive system that lets users iteratively create and modify

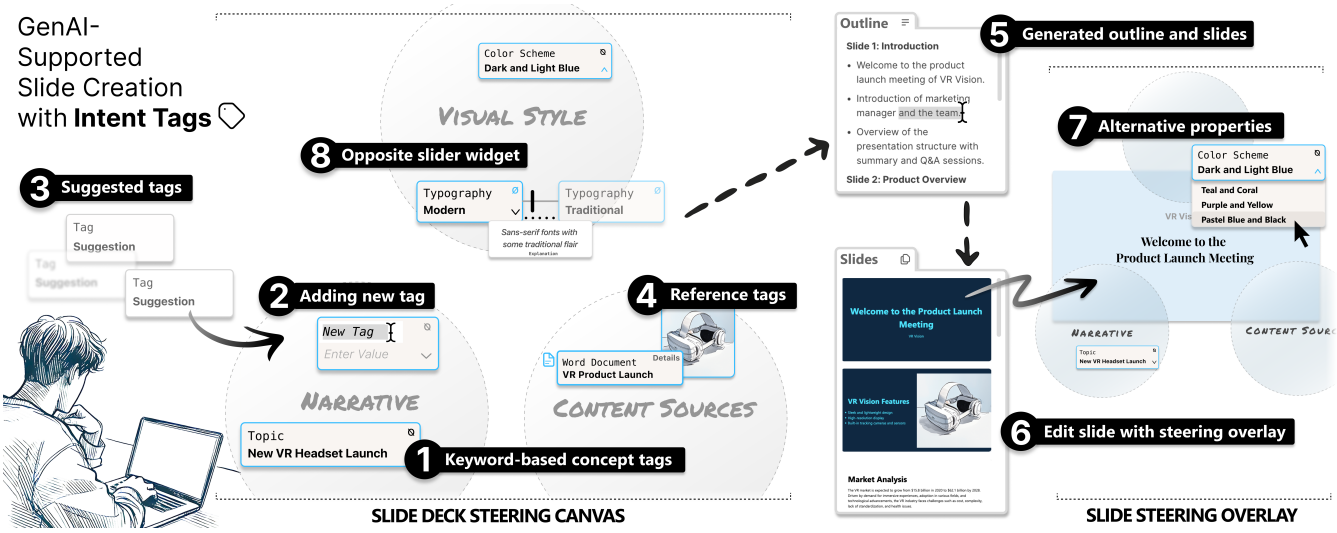


FIGURE 6.1: *IntentTagger*, an intent tag-based LLM-driven slide creation system: Users can steer slide generation with (1) keyword-based concept tags by (2) adding them directly, (3) choosing from suggestions, or (4) adding existing content. (5) A user-editable outline and slide deck are generated based on all selected tags. (6) For fine-grained changes, the user can change individual slides with the same (scoped) intent tagging mechanism. Each tag allows exploring alternative properties in a dynamically generated (7) drop-down list, and tag values can be fine-tuned through the (8) opposite slider widget.

slide decks using intent tags on a 2D canvas interface. In an evaluative lab study with 12 professional PowerPoint users, *IntentTagger* demonstrated several benefits: Participants preferred graphical micro-prompting over traditional chat- and gallery-based GenAI interactions while appreciating its support for non-linear workflows, flexible intent expression, and integrated suggestions that helped them clarify goals and think through slide creation tasks.

To further explore how intent tagging could guide content generation across diverse applications, I sketched alternative user experience (UX) scenarios for rich content generation beyond slide decks, including creating blog posts, videos, and 3D scenes.

Overall, IntentTagger, along with the user study findings and the UX scenarios, illustrates how intent tagging appears to offer a straightforward, generalizable design pattern for a broader application in AI-assisted workflows and scenarios, thereby offering designers a new interaction technique for HCI+AI applications.

6.2 Background

6.2.1 Intent Elicitation in Human-AI Interaction

Technique	**Open Prompt Format**	**System Guidance**	**GUI-based Option Manipulation**	**Continuous Option Representation**
GUI-based/Wizard		✓	✓	✓
Text Prompting	✓			
Chat Dialogues		✓		
Intent Tagging	✓	✓	✓	✓

TABLE 6.1: Outlining the trade-offs between four intent elicitation methods (GUI-based/Wizard Dialogues, Text Prompting, Chat Dialogues, and Intent Tagging); comparing the factors Open Prompt Format (allowing users to express intent freely), System Guidance (guiding users through predefined options), GUI-based Option Manipulation (adjusting single parameters via GUI elements), and Continuous Option Representation (persistent, single mutable objects).

A core challenge in designing effective interactive GenAI-driven systems is ensuring that AI-generated content matches user intentions. This involves two main aspects: the user communicating their intent to the system and the system interpreting the user's intent or guiding them in gradually disclosing it (intent elicitation).

Traditionally, **wizard interfaces** have emerged to guide users through predefined option GUI dialogues to elicit their intent step-by-step for complex tasks [71]. However, natural language processing (NLP) advancements have introduced text-prompting-based GenAI models, such as large-language foundation models (LLMs), that allow users to input their intent as **free-form text prompts** [218] or guide users through a **dialogue** to elicit their intents [55, 275, 291, 312].

However, all such approaches present opposing trade-offs (see Table 6.1): GUI-based menus or wizards can intuitively guide user decisions and reduce ambiguity while allowing for continuous representation of options through graphical manipulatable elements, but they only offer a finite set of options that reflect pre-anticipated use case scenarios by software makers. Free-form text prompting allows users to communicate their intent more flexibly using natural language, but the lack of scaffolding and guidance requires users to know what they want (or need) and how to formulate it as an

effective prompt. While easy for simple tasks, communicating intents can be challenging in complex content creation tasks when users might not be aware of all necessary options upfront and are required to iteratively figure these out (intent exploration). As a compromise, natural language chat dialogues can provide system-guided option exploration and scaffolding. However, such systems can only probe on a limited set of options in a reasonable time frame or number of conversational turns, and they often lack continuous option representation and graphical manipulation.

On the other hand, effectively aligning AI-generated content with user intentions also requires awareness of one's own intentions and understanding of the possible parameter spectrum [352]. Recent work in AI-mediated intent elicitation and sensemaking has explored various strategies and GenAI-driven tools to assist users in more open-ended and iterative ways to elicit and discover their (creative) intent. For example, systems like *Luminate* [341], *Selenite* [216], *Sensecape* [342] and *Graphologue* [166] utilize dynamic LLM-driven GUIs to support users in sensemaking and intent elicitation (e.g., exploring a design space, available options, or topics) by generating and visually clustering semantically related concepts or criteria to help them explore relevant dimensions and parameters matching their tacit intent and needs. Similarly, *SymbolFinder* [281] offers different LLM-driven GUIs to suggest and group related words and images to guide designers in gradually exploring and finding appropriate symbols to illustrate abstract concepts. Other work by *Kreminski and Chung* [191] uses language models and micro-interactions to elicit users' creative intent by formulating and asking open-ended questions about aspects of a game creator's emerging story world. Similarly, *Germinate* [192], a system for game generation, features an interface to help break down the design intents of casual creators by allowing users to specify game design-related parameters, such as game entities or resources, through graphical text-based tags that users can freely define. Based on these user-defined tags, the system attempts to infer additional appropriate tags that users might incorporate or negate (cf. [226]). Similarly, Lin et al. [214] outlined a design space for user interactions with co-creative systems, highlighting the role of explicit communication mechanisms about creative intent, which might be initiated by either the user or the system (mixed-initiative).

Previous work has explored various AI-mediated user intent elicitation approaches and proposed metacognitive user support to improve GenAI-driven tasks. Building atop this prior work, I aim to derive interaction principles for human-GenAI co-creation

that combine the benefits of existing intent elicitation interfaces without their drawbacks while supporting users' intent exploration process through metacognitive support.

6.2.2 Systems for Supporting Text-based Prompting and Steering GenAI

Generative AI *foundation* models, such as large language or diffusion models [40, 307], represent a paradigm shift in artificial intelligence, offering task-agnostic pre-training on large-scale data for various downstream applications [319]. Although extremely versatile, research suggests three properties that make interacting with GenAI challenging: *input flexibility* (in handling free-form language, images, code, etc.), *generality* (applicability to a wide range of tasks), and *originality* (ability to generate novel content) [317].

Prior HCI research has documented interaction challenges in prompting and steering GenAI systems across domains such as coding, illustration design, or engineering [130, 215, 219]. For example, users frequently struggle to craft text-based input prompts that will achieve desired outcomes, and face difficulties interpreting and repairing erroneous outputs [402]. Research has begun to explore mechanisms and interfaces to better support users in working with text prompt-based GenAI models. For example, various works have proposed mechanisms to support users in *prompt engineering* [218] through *"prompt augmentation,"* which automatically modifies and extends a users' input prompt to improve the model's generated output [36, 43, 324]. While such techniques can improve model output quality, users are bound to express their intent in an open-ended text format.

Other work has explored ways of *combining direct manipulation interfaces with text-based GenAI systems* to enable more structured user inputs and reduce semantic ambiguity. For example, *GhostWriter* [396] offers buttons for predefined prompts for LLM-driven text manipulation tasks. While such approaches reduce misalignment and improve feature discoverability by visually exposing model capabilities, they also restrict the available task options to predefined sets and limit the GenAI models' generality.

Recent work has also proposed contextually bounded prompting mechanisms to offer more fine-grained control over Gen-AI-driven content generation. *DirectGPT* [233] explores direct manipulation principles for LLMs, such as the continuous representation of manipulatable objects, physical actions to localize the effect of prompts, and reusable prompts in a toolbar. Similarly, *Cococo* [221] offers users a set of GUI widgets

for steering GenAI music generation, including slider elements to nudge music generation in high-level directions. Other ideas include painting-like interactions, such as how *PromptPaint* [66] enables users to steer diffusion-based text-to-image generation through paint medium-like interactions, and *TaleBrush* [67] supports generative story ideation through line sketching interactions to graphically steer LLM-driven story generation.

Building on this work, I contribute to steering interactions for GenAI by investigating mixed interfaces that blend free text input with dynamic graphical elements. I propose intent tagging as graphical micro-prompting interactions to flexibly support intent expression and elicitation using GUI widgets to allow exploration across various levels of ambiguity.

6.2.3 Non-Linear Content Creation and Iterative Design Workflows

Creating rich content documents, such as blog posts or slide presentations, involves crafting and integrating text, visuals, and multimedia to convey complex ideas while maintaining coherence and audience engagement. Guidelines for creating slide presentations often suggest standardized workflows such as crafting outlines before creating slides [14, 300, 403]. However, other literature emphasizes that slide-creation workflows are less rigid and are mostly shaped by cultural factors and organizational norms [395]. For example, in organizations, people often start or continue presentations from different starting points, such as from existing documents, other slide decks, or templates. Rich content creation processes are also iterative and nonlinear, typically cycling through various stages where creators alternate between exploration and exploitation activities as they make sense of their evolving work [26, 91, 283]. Throughout this process, content creators continuously develop and reflect on their ideas through an ongoing dialogue with their materials, allowing new insights and refined plans to emerge from this interaction [178, 187, 320].

In conclusion, despite recommended workflows for content creation (such as starting with an outline), content creation processes are typically highly situational, nonlinear, and iterative in nature. Therefore, I aim to derive interfaces for GenAI-driven content creation systems to promote non-linear and diverse workflows while supporting reflection on the content creation process.

6.2.4 Systems for Supporting Slide Deck Creation

Creating slide presentations is a multifaceted task that involves various sub-tasks, and previous research has introduced a range of approaches to support different stages of this process.

Besides popular slide-by-slide authoring tools like PowerPoint or Keynote, research has suggested interfaces to support users in **iterative slide creation workflows** by managing multiple presentation versions [98], prototyping slide decks using markup language [101] or authoring slide presentations on a 2D canvas interface [210] for supporting non-sequential workflows.

Besides manual authoring systems, numerous works have explored ways to **automatically generate slide presentations** and rich content documents for single topics [382], or scientific [120], technical [222], educational [73], or semantically annotated source documents [231]. Other systems focus on automatically generating presentation-specific aspects, such as visual-textual layouts [393]. While these systems allow for automatic content transformation and slide creation, they lack methods for users to influence or control the generation process.

Several works have, therefore, proposed mechanisms to enable users to **generate presentations in more controllable and workflow-integrated ways**, such as allowing data scientists to steer the generation of slide presentations from Jupyter notebooks by interactively selecting code cells and slide layouts [373] or through a user-controllable text outline [372]. Similarly, *Knowledge-Decks* [65] allows users to generate slides documenting data science knowledge-discovery processes from automatically collected user behavior events from visual analytic tools.

Previous work has explored various authoring interfaces, automatic slide generation from source documents, and user-controllable slide generation in domain-specific applications like data science. Expanding on these approaches, I aim to explore interaction principles for GenAI-supported slide deck creation across diverse content tasks and workflows. Our approach combines GenAI-driven slide generation from multiple input types (e.g., prompts, text documents, images) with *non-predefined* and *non-linear authoring workflows* in mind that let users seamlessly switch between generation and editing modes across outline, individual slide, or entire deck level.

6.3 Exploring Design Principles for GenAI-supported Rich Content Creation

In this section, I lay out the challenges of rich content document creation in the context of slide creation and how these informed the design principles that led me to intent tagging as an interaction paradigm for enabling novel human-GenAI co-creation workflows.

Slide deck creation is a complex task that requires balancing multiple facets, such as narrative coherence, visual style consistency, and the integration of diverse content sources [14, 300, 395]. These elements must not only align individually but also work together harmoniously to deliver an impactful presentation. The challenges arise from the need to manage and sequence multimedia content while simultaneously crafting a narrative that flows logically across slides. Achieving this requires careful consideration of storytelling, visual design, and the inclusion of relevant data or references, often making the task overwhelming, especially under time constraints or when dealing with open-ended creative briefs.

One of the core difficulties in slide deck creation is the need to manage these distinct yet interdependent elements—*narrative*, *visual style*, and *content sources*—without losing sight of the overall presentation goals.

Traditional tools often compartmentalize these tasks, forcing users to switch between different modes or tabs, which can disrupt the creative process. Linear workflows or predefined templates, while useful in some cases, often fail to accommodate the nuanced adjustments needed to refine a slide deck to a professional standard.

Moreover, as discussed in the previous section, the integration of Generative AI (GenAI) into slide deck creation introduces both opportunities and challenges. Current GenAI approaches tend to offer either (semi-)automated solutions with little ability for users to steer the content generation, design galleries with limited pre-defined option sets, or chat-based interactions that can feel too linear, unstructured, and limited for exploring alternatives.

6.3.1 Design Principles

Based on these requirements, I developed the following interaction design principles for GenAI-supported slide creation:

DP1 Enabling flexible, non-linear, and iterative workflows

To cater to users' diverse slide creation needs and individual working styles, users should be empowered to start and refine their presentations in a variety of ways. Whether beginning with an outline, building on an existing deck, crafting a single slide, or incorporating content from other documents, users should have the flexibility to approach their tasks from any angle. Additionally, the interface should allow seamless transitions between different content views—such as slide, deck, or outline—enabling GenAI-assisted iterative enhancements at any stage of the creation process.

DP2 Accommodating diverse steering input types

Slide creation often requires the integration of multiple sources of information, ranging from explicit, well-defined data (e.g., an annual sales report) to more abstract, nuanced considerations (e.g., audience-specific messaging). To ensure that GenAI-generated outputs align closely with user intentions, the interface should facilitate the input of a wide range of content types, including natural language descriptions, existing documents, images, and other relevant media. By enabling users to provide rich, varied inputs, the system can better interpret and fulfill the specific needs of each presentation.

DP3 Supporting intent expression across varying levels of abstraction

In many slide creation scenarios, users may have clear, specific ideas about certain aspects of their presentation, while other elements remain less defined or harder to articulate. To accommodate this variability, a GenAI interface should allow users to express their intent at different levels of abstraction—ranging from detailed, precise instructions to more general, high-level directives. This flexibility not only helps users articulate their ideas more effectively but also supports iterative exploration and refinement of their presentation content.

DP4 Leveraging AI for contextual content and terminology suggestions

To assist users in overcoming challenges like the "blank page syndrome" or finding terminology to describe their intent, the interface should harness the associative power of LLMs to provide contextual suggestions and inspiration. By offering alternative ideas, vocabulary, and creative prompts, the system can stimulate users' thinking, helping them to refine their content and explore new directions in an iterative process.

DP5 Providing pre-computed real-time previews

To allow users to better and quicker anticipate how specific choices will impact

generated slides, the tool should provide pre-computed real-time previews to, for example, let users rapidly explore how different fonts or colors would look on a given slide.

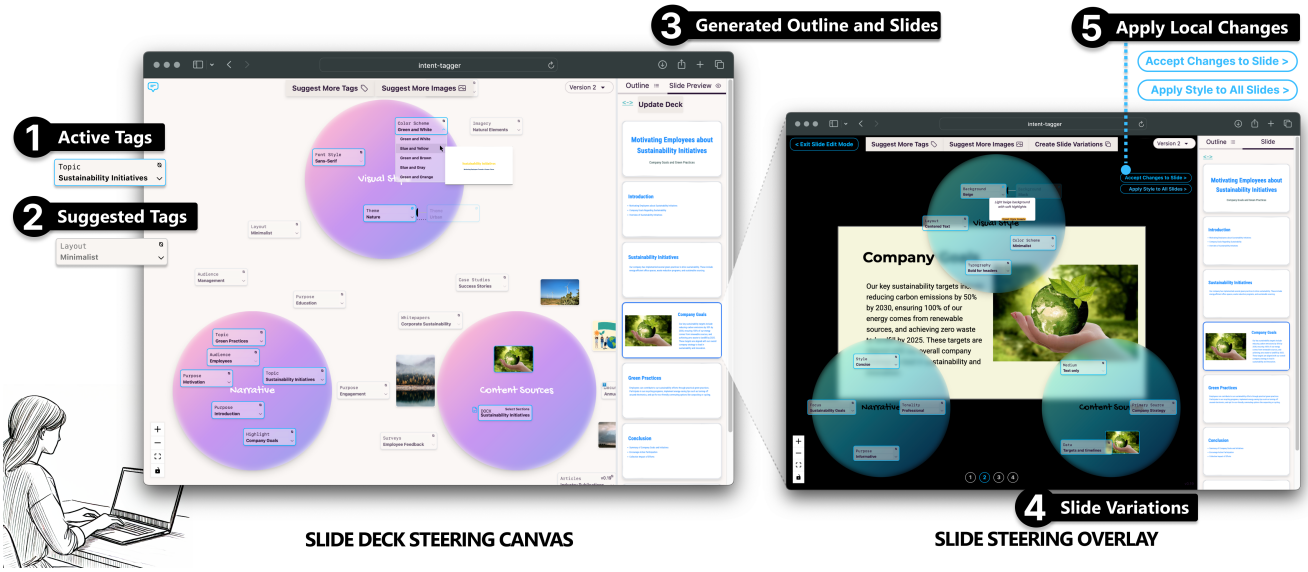


FIGURE 6.2: Screenshots of the *IntentTagger* system: (Left window) A user creates a slide presentation using the Slide Deck Steering Canvas with (1) active tags in the three circular tag groups surrounded by (2) inactive system-suggested tags. (3) The generated outline and slides are displayed in the tabbed sidebar. (Right window) The user makes (scoped) adjustments to a single slide via the Slide Steering Overlay and (4) explores generated slide variations. (5) Then, local slide changes can be accepted or applied to the entire deck.

6.4 IntentTagger: An Intent Tagging-based Slide Creation System

Driven by the requirements of rich content creation tasks and the design principles described in the previous section, I explored novel input strategies through graphical micro-prompting interactions—an interaction notion I coined *intent tagging*. In addition to my design principles, a central inspiration for conceptualizing creative intent elicitation through granular and flexible micro prompts that represent users' intention was the "mood board" technique, in which creatives iteratively compile visual and textual elements into a collage to convey an overarching theme or creative direction [124]. Similar to how mood boards enable capturing and expressing designers' intent

through compiling different media with varying levels of abstractions—such as words and pictures ranging from abstract concepts, stylistic references, or specific material details—intent tagging allows users to express their intent openly through collections of intent tags.

To explore further the idea of intent tagging, I created *IntentTagger*, an LLM-driven system allowing users to iteratively create and modify slide deck presentations using intent tags on a 2D canvas interface (see Figure 6.2). In this section, I demonstrate the utility of *IntentTagger* by first illustrating its functionalities in a use-case scenario. I then follow with a detailed description of the interface mechanisms and their implementation.

6.4.1 Example Use Case Scenario

Lucy is a sustainability manager at a company and wants to prepare a short slide presentation for new employees to inform them about sustainability initiatives at her company.

Lucy starts IntentTagger and clicks on the *"Create a new presentation from prompt..."* button. A modal popup appears, and she types *"a presentation for introducing sustainability initiatives for new employees"* in the prompt field and presses enter. The modal disappears, and the three circular steering groups on the steering canvas pulse, then three concept tags appear in the *"Narrative"* circle with the parameters *"Topic: Sustainability Initiatives," "Audience: New Employees"* and *"Purpose: Introduction."*

Tag Suggestions: To get inspiration for further presentation talking points, she clicks *"Suggest More Tags,"* and shortly after, new tags appear outside of the circular tag groups. Lucy finds some of the suggested tags useful and drags these into the *"Narrative"* group to include them in the generation, such as *"Focus: Eco-friendly Practices," "Highlight: Company Goals," "Section: Key Initiatives,"* and *"Objective: Corporate Responsibility."*

Including External Content: Lucy also wants to include sections from an existing Word document detailing the company's policies. With the mouse cursor, she drags the document from her file browser onto the IntentTagger app, and a new reference tag appears in the *"Content Sources"* group. She clicks on the tag's *"Select Sections"* button, which opens up the tag's Text Selection Widget showing the document's content. In the widget, she highlights two document sections, clicks *"Include Section in Presentation"* from the widget's context menu, and closes the widget.

Generating an Outline: Next, she wants to generate a first draft of the presentation's outline and presses *"update outline from intent tags"* in the outline view panel. The tag groups start pulsing, and then the new outline appears in the outline text editor.

Manually Crafting Concept Tags: Lucy is happy with the structure overall but finds it too long (20 sections). Instead of manually editing the outline, she adds a new concept tag to the *"Narrative"* group and enters *"Number of slides"* into the tag's upper text field and *"10"* into the lower and clicks the *"update outline from intent tags"* button again. The newly generated outline is now shorter and only comprises ten sections.

Generating Slides: Next, Lucy wants to generate a first draft of the slide deck and clicks the *"Update slides from intent tags"* button. Shortly after, the generated slides appear in the slide preview side panel. After evaluating the slides, Lucy realizes that adding some images and adjusting the visual style could enhance the presentation's appeal.

Including Suggested Images: She clicks the *"suggest more images"* button on the tag steering board, and shortly after, new Reference Image Tags appear outside the *"Content Sources"* tag group with eco-friendly-themed stock photos. Lucy drags some of the image tags into the *"Content Sources"* group to add them to the presentation.

Adjusting the Visual Style through Tags' Alternatives Drop-down: To make visual adjustments to the slides, she drags in some of the previously suggested tags, such as *"Typography: Sans-Serif,"* *"ColorPalette: Green and Blue,"* and *"Theme: Nature"* onto the *"Visual Style"* tag group. To get a better sense of typography alternatives, she clicks on the tags' drop-down, and a list of alternative tag values appears, such as *"Serif,"* *"Monospace,"* and *"Handwritten."* While hovering over each option with the mouse, Lucy sees a preview thumbnail of that font style applied to the current slide. Lucy chooses *"Monospace."* Now, Lucy clicks *"Update slides from intent tags"* again, and shortly after, she sees the revised slides reflecting the intended new visual style, including the images.

Adjusting a Single Slide with Tag Steering Overlay: Lucy decides to refine the slide deck further. She clicks on the third slide in the *Slide Preview* panel to open the slide edit view and review its content in detail. The slide contains a text paragraph and an image, but she wants to show a list of bullet points instead. To modify the slide, she clicks on the intent tag icon next to the slide, and the three steering tag groups appear as an overlay containing tags representing the slide's *Narrative*, *Visual Style*, and *Content Sources*. Lucy selects the *"Text Format"* tag inside the *Narrative* group and clicks on its drop-down widget. A list of alternative tag values appears, including *"Bullet Points,"* *"List,"* and *"Table."* Lucy chooses *"Bullet Points."* To adjust the background color,

Lucy creates a new tag in the *Visual Style* group and types in *"Background"* and *"pastel color."* Now, she clicks on the *"Explore slide variations"* button, and the system generates several alternative versions of the slide, each with slightly different text variations and pastel background colors. Lucy clicks through these variations and selects one that better aligns with her presentation's tone.

Applying a Slide's Style to All Slides: To maintain consistency across the entire presentation, Lucy uses the same visual style and selects *"Apply to all slides."*

Satisfied with the changes, Lucy saves the presentation, ready to deliver it in the upcoming meeting with her colleagues.

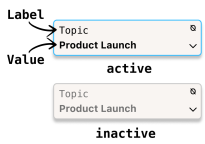
Adjusting the Presentation to New Requirements: After a couple of months, Lucy needs to adapt the presentation for a different audience—existing employees rather than new hires. She clicks on the *"Audience"* tag in the *Narrative* group and changes it from *"New Employees"* to *"Existing Employees."* The system instantly updates the content and tone of the slides to better fit this new audience. For example, introductory sections are replaced with more in-depth discussions of ongoing sustainability projects and their impact.

6.4.2 Interface Features

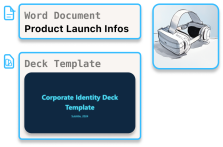
Here, I provide a more detailed explanation of IntentTagger's features:

Intent Tags: Concept Tags, Reference Tags and Groups (DP2)

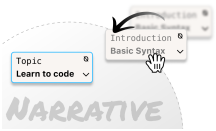
Intent tags are collections of single keywords, phrases, or media artifacts describing users' intended outcomes. Tags serve as granular and flexible micro prompts on a zoomable 2D canvas. Users can steer the generative slide creation system through two types of tags:



(1) Concept Tags: Concept Tags are two-part text micro prompts of the format [label: value], such as *"Topic: Product launch"* or *"Color scheme: Corporate."* Tags can be active or inactive to toggle their influence on generated content.



(2) Reference Tags: *Reference Tags* allow users to provide external documents such as Word documents, images, or other slide decks as references for slide generation. Users can add these files by drag-and-drop from their file browser.



Tag Groups: All active tags are visually clustered into three circular groups on the interface, each representing different slide creation aspects: *Narrative*, *Visual Style*, and *Content Sources*. Users can drag tags in and out of groups to activate and deactivate them.

Deck Steering Canvas and Slide Steering Overlay (DP1, DP3)

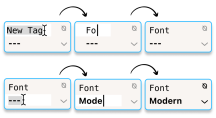
Users can generate and modify single slides or entire slide decks using Intent Tags. The *Deck Steering Canvas* allows users to steer the (re)generation of entire slide decks. For example, specifying the number of total slides or changing the overall tone of the deck's text from brief to more verbose. Alternatively, the *Slide Steering Overlay* allows users to steer the (re)generation and explore alternatives of single slides.

Outline Editor (DP1)

Parallel to slides, the system also generates a presentation outline, which can be optionally edited using the outline editor. This editor also allows participants to start a new presentation by first drafting an outline or letting the system generate an outline from provided intent tags.

Editing Intent Tags, Tag Widgets, and System Suggestions (DP3, DP4, DP5)

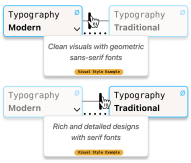
Intent tags allow users to formulate their intent in several ways, such as manually creating and editing tags or choosing from options generated through LLM-driven dynamic adaptive UI mechanisms:



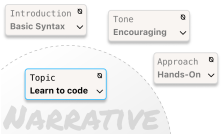
Manually Crafting Tags Users can manually create and modify tags by freely typing words or phrases into its two text fields, such as "Font: Modern" or "Tonality: Engaging."



Drop-Down List Users can also explore alternative properties for each concept tag by browsing its dynamically generated drop-down list, which also provides pre-generated visual previews in a hover tooltip.



Opposite Slider Widget Users can fine-tune tag values with more granularity by using a slider widget. For this, the system automatically generates an opposite value for each tag with a five-step continuous range slider in between, including descriptive examples illustrating each step's value.



Tag Suggestions For each tag group, users can request new tag suggestions, such as additional talking points, visual styles, or content sources, from the system based on already specified tags. Generally, all suggested tags first appear outside the circular groups, and users can drag suggested tags into a group to include them in the generation.

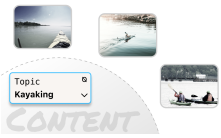


Image Suggestions In addition to tags, users can also request system suggestions for new image tags. When requested, each time the system returns different images from a background online image search related to the board's active tags.

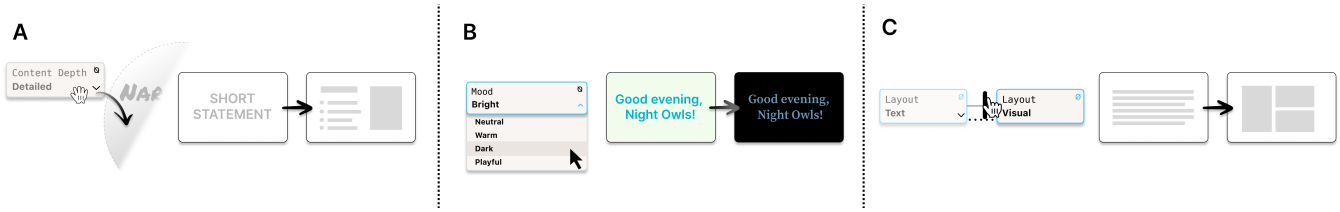


FIGURE 6.3: Examples of steering GenAI slide generation with Intent Tags: (A) Influencing the detail level of the displayed information, (B) steering the "mood" from "bright" to "dark," or (C) gradually blending the layout between text- and image-heavy.

Steering Slide Generation

To summarize the different ways of steering slide generation with Intent Tags, Figure 6.3 illustrates how including a tag suggestion modifies the content of a slide, using the drop-down list changes a slide's mood, and using the opposite slider widget blends between a text- and image-heavy layout.

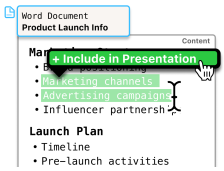
Previews of Slides and Slider Values (DP5)

IntentTagger integrates several preview features, allowing users to explore alternative tag values without committing to long waiting times: The tag's *Drop-down* list shows

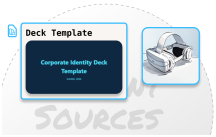
pre-generated previews as tooltips of how the current slide would look like with that tag value applied. The *Opposite Slider* widget shows generated explanations describing the resulting effect if applied to the generation. All tag previews and explanations get asynchronously pre-generated in the background, allowing users to explore these options in real-time.

Referencing External Documents (DP2)

Besides *Concept Tags*, users can also include external files for the generative slide creation as *Reference Tags*.



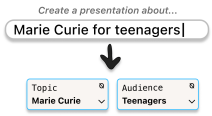
Including Content From Word Documents Users may drag in Word documents as content sources for the generated presentation. Optionally, users can select sections through a text selection pop-up to include only specific parts in the generated presentation.



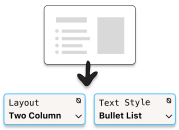
Including Images and Slide Deck Templates Users can drag external images onto the canvas to include these in the presentation. Users also have the option to reference other existing slide decks as visual templates.

Tag Grounding Acts (DP4)

While Intent Tagging presents granular and flexible mechanisms for users to communicate their intentions to the generative AI slide creation system, this principle also works backward in so-called *Tag Grounding Acts*, where the system creates intent tags from user inputs such as longer text prompts or slides.



Grounding from Text Prompt This feature allows users to start a new presentation by providing instructions via a longer conventional text prompt, which the system will then automatically decompose into individual Intent Tags.



Grounding from Slide Each time a user invokes the *Slide Steering Overlay* to adjust single slides, the system first analyzes the slide's *content sources*, *narrative*, and *visual style* and then pre-populates the interface with intent tags representing the slide's current state. This provides an easy starting point for users to make adjustments through Intent Tags.

6.4.3 Implementation Details

IntentTagger is implemented in TypeScript using ReactJS [297] with ReactFlow [116] for the tag canvas interface and Blocknote [37] for the text editor features. Mammoth.js [380] is used for importing Word documents, and slide rendering is based on a modified fork of spectacle.js [118]. For generating slides and adaptive UI, I use the official OpenAI API [267] to execute prompts using "gpt-4o" and the Bing search API [242] for image suggestions. To suggest new tags, I prompt GPT to generate a list of *[label:value]* pairs for each tag group to augment the existing set of active intent tags. For outlines, GPT is instructed to generate a presentation outline in markdown format from a list of a user's active concept and reference tags (including images if present). To generate slides, I then prompt GPT to return a template-based JSON slide deck representation by providing the outline, active intent tags, and deck references (if specified). For generating the drop-down values, slide previews, and slider explanations, the system asynchronously requests these from GPT in the background for each new or modified tag on the canvas. Please see the Appendix for detailed prompts.

6.5 User Study

To better understand the possible benefits and limitations of intent tagging-based interactions, I conducted a lab user study aimed at providing insights into these research questions:

RQ6.1 *What are the key differences between chat-based and intent tag-based interactions with GenAI?*

RQ6.2 *How do people work with intent tags?*

RQ6.3 *What are users' perceived benefits and challenges for GenAI-driven slide creation with intent tags?*

For RQ6.1, I decided to compare intent tag-based interactions with chat-based and design gallery-based approaches (text prompting and choosing from a set of options) since these represent the currently most common forms of interacting with GenAI for content creation in commercial systems, such as in OpenAI's ChatGPT/DALL-E [36], Adobe's Firefly [3] or Microsoft's PowerPoint Copilot [244] and Designer [245] features.

6.5.1 Participants

I recruited 12 participants *(8 self-identified as females, 3 males, and 1 non-binary, age M=31.6 years (SD = 7.65))* with professional slide presentation creation experience via email lists at a large software company. The participant pool consisted of individuals with diverse job titles, and the majority of selected participants used PowerPoint at least multiple times per month in their jobs (see Table D.1 in the Appendix). All participants had previous experience using the *Designer* and *Copilot* features in PowerPoint, ensuring existing familiarity with generative AI functionalities for slide deck creation. Participants signed an IRB-approved consent form and were compensated with a $50 gift card after study completion.

6.5.2 Procedure and Tasks

The lab study was structured into four phases:

1) On-Boarding (20 min): At the beginning of the session, after a general study introduction, participants watched a video tutorial demonstrating IntentTagger's core functionalities with a step-by-step example. Following the video, participants were asked to complete two five-minute guided hands-on structured tasks (editing an existing slide and making adjustments to an entire deck) to familiarize themselves with the tools' interface and operation.

2) Comparative Tasks (2 x 10 min): After the onboarding phase, participants completed two comparative tasks, each lasting 10 minutes, to evaluate their ability to create presentations using IntentTagger and a baseline system. In both tasks, participants were asked to create a 6-slide presentation aimed at educating teenagers about the inventions and scientific discoveries of a historical figure (*"The Discoveries of Marie Curie"* or *"The Inventions of Nikola Tesla"*). Participants alternated between using IntentTagger and Microsoft PowerPoint [243] across the two tasks in randomized order. For PowerPoint, participants were restricted to using only the integrated Copilot feature [244] (chatbot interface with optional document upload) and the Designer feature

[245] (design gallery for slide layouts) for slide generation and modification. As a starting point, participants were provided with a Word document containing the relevant Wikipedia article in both tasks. Participants were required to think aloud during the tasks, and after completing each task, they filled out a survey with attitudinal 6-point Likert scale questions. To mitigate order effects, the order of systems and presentation topics was randomized across participants.

3) Semi-structured Task (10 min): In the third phase, participants were tasked with creating a 7-slide presentation from scratch using IntentTagger. They chose a topic related to a hobby they enjoy, aiming to convince others of its value and explain how to get started. Participants had ten minutes to complete the task, using only the prototype system without directly editing text or images on the canvas. They were required to think aloud as they worked, and before starting, they briefly described their topic and intentions for the presentation. I created this task to encourage participants to engage more freely with IntentTagger's features, focusing on content creation and presentation design related to a topic they are knowledgeable about and emotionally connected to. Since IntentTagger's deck generation time increases per slide, I deliberately limited the number of slides to six and seven per task in phases 2 and 3 to keep the processing time for each cycle within 15 seconds.

4) Exit Interview (20 min): In the final phase, participants participated in a semi-structured interview to provide feedback on their experience with IntentTagger, focusing on its overall utility, comparison with tools like PowerPoint's Copilot, and the effectiveness of intent tagging interactions for slide creation. They shared insights on the tool's strengths, areas for improvement, and potential integration into their professional workflows.

6.5.3 Collected Data, Measures, and Analysis

Across the study, I collected the following data:

- Video, screen, and audio recordings and machine-generated transcripts of the task think-aloud sessions
- Audio recordings and machine-generated transcripts of the post-task interviews
- Post-task survey data
- Participant-created presentations and related IntentTagger project files

To compare chat-based and intent tag-based interactions (Q1), I analyzed the post-task surveys from phase 2 that probed on participants' perceived ease of use, efficiency,

and control over the slide generation process on a 6-point Likert scale. I applied the Wilcoxon signed-rank test to assess statistical significance and calculated 95% confidence intervals for mean differences via bootstrapping with 10,000 replications using R [293]. This approach has been suggested for similar data and studies [232, 409].

To answer how people work with intent tags (Q2), I conducted a video interaction analysis [33] of the video recordings collected in the semi-structured task *(study phase 3)*. I manually coded participants' interactions with IntentTagger's features in Atlas.ti [18], such as their interactions with tags or the moments they triggered the outline and slide generation.

Finally, to investigate users' perceived benefits and challenges of intent tags (Q3), I conducted a reflexive thematic analysis [44] of the interview transcripts. I followed an iterative inductive coding process (using Marvin [228]) and generated themes through affinity diagramming using Miro [247].

6.6 Study Findings

Participants went through the tasks without abandoning them.

6.6.1 Key differences: Chat vs Intent Tag-based Interactions (RQ6.1)

Observations

In the two comparative tasks (phase 2), participants had to create a presentation about Nikola Tesla or Marie Curie on six slides using engaging images and language suitable for teenagers, based on a provided Word document containing information from a Wikipedia article. Here, I report on the observed key differences between participants' processes during these tasks when working with the chat-based system and IntentTagger.

Chat-based System. In the chat-based system, participants typically began by providing a straightforward task prompt, such as *"Create a new presentation from /Tesla.docx about Nikola Tesla for teenagers on six slides."* However, the system consistently generated a 20-slide PowerPoint presentation based on all content from the document in a generic way, including generic and vaguely related stock images. When participants tried to refine their prompts, such as specifying the desired number of slides or requesting specific content to be highlighted, the system frequently ignored these requests and

generated a new slide deck similar to the previous one. As a consequence, participants found themselves repeating their instructions or trying different phrasings to guide the system, often without noticeable changes. The system frequently returned messages like, *"I don't recognize that wording as something I can do,"* offering alternative actions that were not relevant to the task. Consequently, participants spent most of their time trying out different text prompts to get the system to generate a more targeted, shorter presentation; however, in all cases, without success. Occasionally, participants went on to tweak the layout and visual style of single slides using the Designer feature by selecting a layout from a design gallery list. While they managed to adjust the style of single slides in isolation, they could not achieve a consistent look and feel for the entire deck this way.

IntentTagger. With IntentTagger, participants typically began by providing the Word document as a reference tag and then continued by providing further concept tags for key elements like the number of slides, audience, and the focus areas of the content. In addition, many participants used the reference tag's text selection widget to specify specific sections of the provided Word document to ensure that only relevant information was included in the presentation. After specifying tags matching the task brief, most participants also made use of the image suggestion feature from which they added some images to the content source tag group. Then, after generating slides, they either continued to make refinements on single slides and then updated the deck's style based on the adjusted slide, or they continued refining the entire deck by adding or modifying intent tags on the deck steering canvas. Participants also frequently used the system's tag suggestions to refine the presentation's content and visual style.

From a usability perspective, some participants struggled initially with understanding how to effectively use the intent tags, particularly in determining how granular or broad their tags should be or to which tag group certain tags would belong. However, they soon became more confident and many participants commented on the ability to better convey their intent with intent tags and tag groups: *"it definitely feels like I get something close to what I had in mind [...] Looks like something efficient, I will be happy to use it in my work."* (P09)

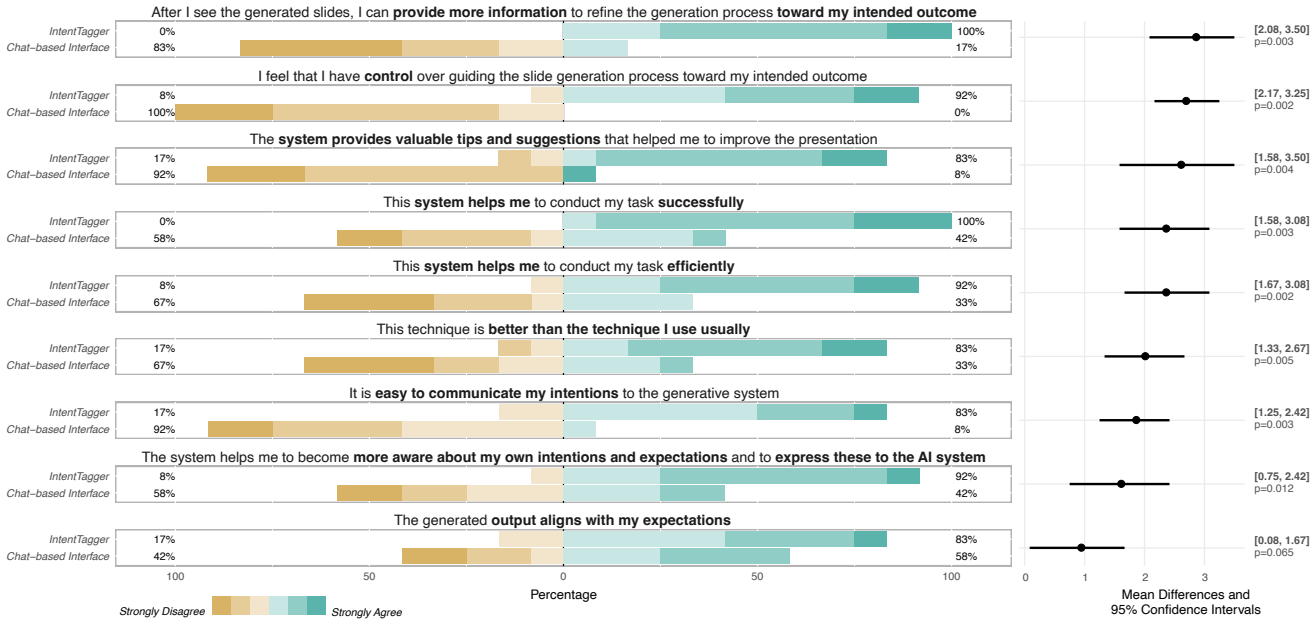


FIGURE 6.4: (Left) Participants' responses when rating the 6-point Likert statements for IntentTagger and Chat-based Interface, ranked from largest to smallest effects; (Right) Dots show the mean difference of IntentTagger compared to Chat-based Interface; Bars are the 95% CIs calculated with the studentized bootstrap method.

Questionnaire

In the comparative post-task surveys, participants rated all nine 6-point statements higher for Intent Tagger (Figure 6.4). Participants expressed a **significant improvement in intent elicitation and alignment with IntentTagger** over the chat-based system regarding the **iterative process** (*"After I see the generated slides, I can provide more information to refine the generation toward my intended outcome"*, $MD = 2.83$), **control over guiding the system** (*"I feel that I have control over guiding the slide generation process toward my intended outcome"*, $MD = 2.67$), **intent communication** (*"It is easy to communicate my intentions to the generative system"*, $MD = 1.83$), **meta intent elicitation** (*"The system helps me to become more aware about my own intentions and expectations during the task and to express these to the AI system"*, $MD = 1.58$) and **alignment with their expectations** ($M = 0.92$). Participants also valued the **guidance for slide creation** of IntentTagger higher (*"The system provides valuable tips and suggestions that helped me to improve the presentation"*, $MD = 2.58$) and felt they were able to **conduct the task more successfully** ($MD = 2.33$) and **efficiently** ($MD = 2.33$) than with the chat-based system. They also felt that IntentTagger would be **better than the technique they usually use** ($MD = 2.0$).

Table 6.2: Overview of semi-structured task results (phase 3); reporting the self-chosen topic of the slide deck and task process summary statistics; *Note: During P04's session, the image suggestions failed to load due to an API outage.

ID	Slide Deck Topic	Task Time (mm:ss)	# Outlines Generated	# Decks Generated	# Total Tags	# CncptTag Sugg Included	# ImgTag Sugg Included	% Sugg Included	# Drop-Dwn Included
P01	Hiking	11:19	3	4	14	3	4	50%	3
P02	Yoga	06:32	3	3	21	12	5	81%	2
P03	Kayaking	07:01	3	2	25	13	5	72%	1
P04*	Homemade Pizza	08:55	2	3	20	17	0*	85%	4
P05	Growing dahlias	08:12	2	5	16	6	5	69%	1
P06	Racing road bikes	08:20	2	1	14	4	3	50%	2
P07	Skateboarding	07:53	1	3	12	8	3	92%	2
P08	Dungeons and dragons	06:42	2	1	21	9	6	71%	0
P09	Practicing Taekwondo	11:43	5	3	11	5	3	73%	2
P10	Tennis	09:01	3	4	16	8	3	69%	0
P11	Why trombone is cool	09:07	3	2	16	8	4	75%	0
P12	Paddle boarding	09:07	2	2	24	17	6	96%	2
Mean		*08:39*	*2.6*	*2.8*	*17.5*	*9.2*	*3.9*	*73.5%*	*1.6*

6.6.2 Observed Content Creation Workflows Using Intent Tags (RQ6.2)

During the semi-structured task *(study phase 3)* participants created a presentation from scratch about a personal hobby by using IntentTagger within 7 – 12 minutes (see Table 6.2). Most participants started by using the *Grounding from text prompt* feature to formulate an initial longer text prompt, which the system decomposed into an initial set of intent tags from where the participants continued to work on the Deck Steering Canvas (see Figure 6.5 upper lanes). Only two participants (P03 and P10) decided to start by adding concept tags manually. On average, at the end of their sessions, users had 17.5 active intent tags (concept and image tags combined) and had generated 2.6 outlines and 2.8 slide decks. Of those tags, on average, 9.5 were tags users took from the system's suggested concept tags, and 3.9 were image tags (ranging between 50–96% of active tags that users chose to include from system suggestions). Optionally, I included

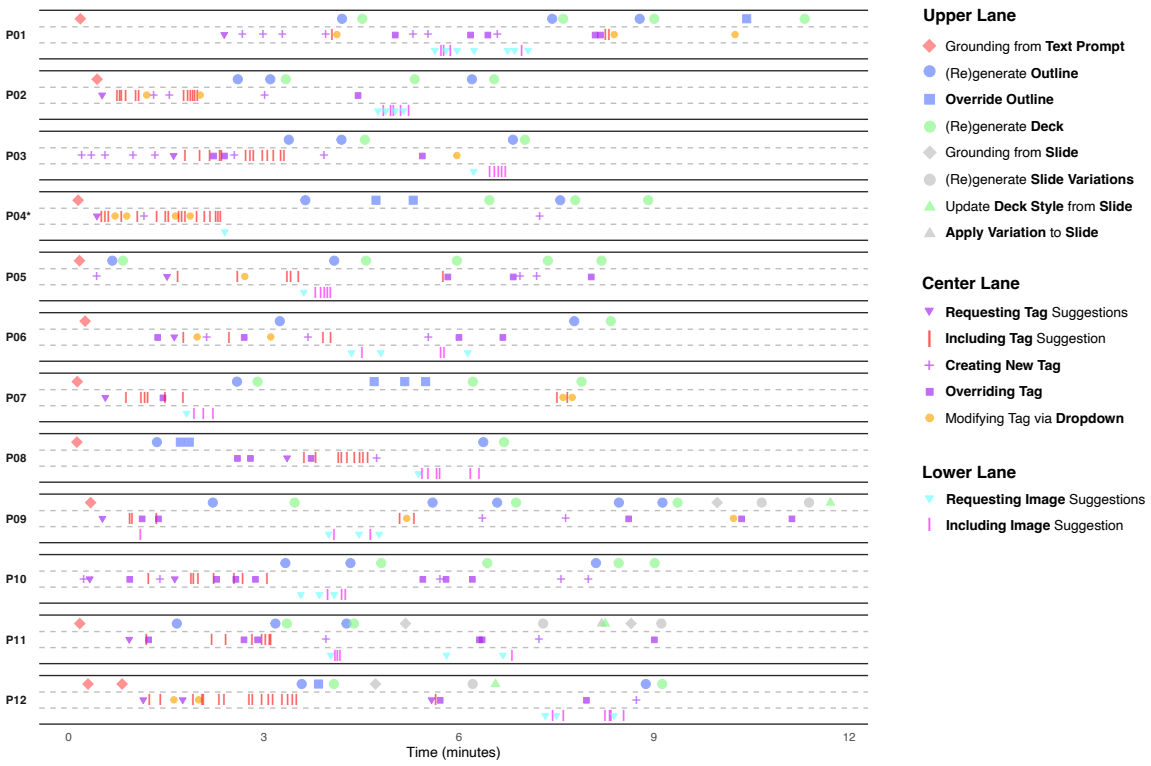


FIGURE 6.5: Timeline plots visualizing participants' interaction events of the semi-structured task (phase 3); upper lanes show events related to outline and slide generation; center lanes show interactions with concept tags; lower lanes show interactions with image reference tags; tag modifications via the *Opposite Slider Widget* are omitted since only a few participants occasionally explored slider options but without applying these to the slide generation (see section 6.6.2) *Note: During P04's session, the image suggestions failed due to an API outage.

screenshots in the Appendix to give an impression of the variety of participant-created slide decks and tag steering boards.

Workflows

All participants generated an outline before generating a slide deck (see Figure 6.5 upper lanes). While this order was partially driven by IntentTagger's requirement for an outline to enable slide deck generation, many participants also expressed in the post-task interviews that this order felt intuitive to them (see section 6.6.3). Almost all users first specified several intent tags for the narrative before generating a first outline after 2-4 minutes. However, two users (P05 and P08) generated a first outline before making substantial tag edits and then refined these using tags. Most participants did exclusively steer the outline and slide deck generation using intent tags, while three

users (P04, P07, P08) made manual edits to the outline at a point. In addition, most users worked exclusively in the Deck Steering Canvas, but three users (P09, P11, P12) also used the Slide Steering Overlay to adjust single slides and then updated the deck's style from that slide. Overall, a common pattern was adding or editing intent tags in the Narrative tag group and then optionally generating a first outline and slide deck. From there, most people continued adjusting the visual style through tags, and lastly, they added image tags to the content sources group.

Prompting With Intent Tags

Participants demonstrated a variety of approaches when prompting with intent tags, particularly when dealing with ambiguous or open-ended tags like color or style. For example, one participant specified the header color as *"Gold"* and the color scheme as *"Dark and Moody"* (P11), while another participant used hexadecimal notation *("#FF33CC")* for body color (P05).

Users' Initial Uncertainty Around Tag Naming. Despite the flexibility, some participants experienced uncertainty about how to "correctly" name certain tags, particularly when it came to specifying the number of slides. The task required a seven-slide presentation, and users created tags like *"Number of Slides: 7," "Slide number: 7 slides," and "length: 7 slides in total."* During the think-aloud sessions, participants expressed hesitation about whether they were using the right terminology. However, after seeing the generated slide deck, they gained confidence in the system's ability to interpret a variety of tag expressions effectively.

Users' Initial Uncertainty Around Tag Placing. Interestingly, while most participants placed the tag for the number of slides within the narrative tag group, there was some variation in how users organized their tags. For instance, one participant placed the slide number tag under visual style (P08), while another placed it in the content sources group (P10). Regardless of where the tag was placed, the system consistently produced a seven-slide presentation. This adaptability underscored the system's capacity to handle user input flexibly, ensuring that the end result matched the users' intent even when the tags were categorized differently.

Flexible Handling of Longer Prompts. One participant created an exceptionally long and detailed prompt when using the initial grounding through text prompt, which

led to the generation of tags containing chains of comma-separated phrases. While these tags were visually less ideal and cluttered the workspace, the system still managed to process them effectively (P08).

User-Driven Tag Hierarchy and Organizational Strategies. In addition to varying naming conventions, some participants developed a sense of structure and hierarchy by visually organizing the tags within the tag groups. Although the generative system currently does not inherently recognize such tag hierarchies, it was interesting to observe how participants imposed their own sense of order. Another strategy for creating hierarchy through tags was observed when P01 explicitly labeled tags to represent an ordered flow, such as *"Topic 1: hiking"* and *"Topic 2: hiking shoes."* Overall, the variety of tag naming and placement strategies showed that while some users were initially unsure about the "right" way to prompt the system, they quickly became more comfortable with the process and how to utilize the tag's granularity and various levels of ambiguity.

Interactions With System Suggested Tags. Participants actively engaged with system-suggested tags early in their content creation process, with all users requesting tag suggestions within the first three minutes. This was typically followed by an intense phase of tag selection and editing that lasted 1 – 3 minutes (see clusters of red vertical strokes ||| in Figure 6.5). For example, in a presentation about road cycling, the system suggested visual style tags like *"fonts: sporty and dynamic"* and *"backgrounds: scenic roadways,"* which the participant included (P06). Similarly, in a kayaking presentation, the system suggested *"Font: Casual and Readable"* and *"Icons: Outdoor activity symbols,"* demonstrating the system's ability to offer contextually relevant suggestions that aligned with participants' content themes (P03).

Users leveraged suggestions to guide and refine the narrative structure. For narrative content, the system's suggestions also played a pivotal role in helping participants develop the flow and focus of their presentations. For instance, during a presentation on the *"Introduction and benefits of yoga,"* the system suggested tags such as *"Audience: Beginners"* and *"Benefits: Mental Clarity,"* which the user incorporated into their narrative structure (P02). These suggestions helped guide users in tailoring their presentation content to the intended audience and goals, adding more depth to the narrative while reducing the cognitive load of coming up with all the necessary elements

from scratch.

Users frequently explored and chose drop-down suggestions. I observed several distinct patterns in how participants interacted with and utilized the suggested tags: In some cases, users directly dragged the suggested tag into a relevant group and then changed its value by selecting an option from the alternatives drop-down widget (see sequences of a red vertical stroke followed by an orange circle |● in Figure 6.5, center lane). For example, when the system suggested *"cycling: cycling for fitness"* for the topic of road riding, one participant dragged it into the narrative group and then refined it by selecting *"cycling for environmental benefit"* from the dropdown (P06, see Figure 6.5 around minute 2).

Users frequently overrode system suggestions. Another pattern I frequently observed was that participants often took more exploratory approaches, such as dragging the tag into the group and examining the available dropdown values before manually overriding the tag's value or label to better suit their vision (see sequences of a red vertical stroke followed by a purple square |■ in Figure 6.5, center lanes).

Participants requested image suggestions in quick succession. Image suggestions triggered by the system followed different workflows, as the system provided five different images each time, prompting some participants to drag and drop the suggested images into the content sources before triggering the system again for more options. (see sequences of a cyan triangle followed by pink vertical strokes ▼ | in Figure 6.5, lower lanes)

Users explored the slider but not for steering. Additionally, while participants occasionally explored slider options to see the opposite ends of a spectrum, this was done more for curiosity than for direct steering slide generation. This indicates a potential area for further refinement in how the slider functionality is integrated into the generation process.

In the Slide Steering Overlay, users primarily used "Grounding Tags" for adjustments. Lastly, those participants who used the Slide Steering Overlay to adjust single slides primarily engaged with the tags that the system had generated through its grounding from slide feature (see sequences of a grey diamond in the upper lanes

followed by events in the center lanes in Figure 6.5, upper lanes). For example, when P09 invoked the *Tag Steering Overlay* for adjusting a single slide, the system created multiple tags representing the slide's current state, one in the visual style group stating *"Color: Red and Black Text."* To make adjustments, the user first explored the drop-down widget's alternative suggestions and then overrode the tag's value with *"blue, red, and white,"* resulting in a slide variation with red and white text on a blue background.

6.6.3 Users' Perceived Benefits and Challenges for Working with Intent Tags (RQ6.3)

I clustered the perceived benefits and challenges for users working with intent tags into three themes: *Expressing intents with tags*, *catering to users' presentation needs and workflows*, and *meta-intent elicitation*.

Expressing Intents with Tags

All participants found intent tags advantageous compared to chat-based prompts, appreciating their conciseness and the ease of specifying inputs without needing to write out long sentences: *"it's quite like fun to be able to sort of click and not have to write out a lot [...] I think the benefit of having these labels [over chat-based] is the conciseness. I don't have that pressure to have to form a sentence."* (P02)

Users liked micro-prompting for its granularity and control. Many participants also appreciated the granularity and control of intent tags while being able to quickly see which presentation-related aspects they can specify: *"I really liked that granularity where you could specify, like, this is exactly the font style and I want the kind of theme of it. I like that aspect a lot [...] visually seeing these buckets of information where I can specify different tags. I think that is helpful."* (P07)

Users valued the structure that tag grouping provided. Many participants found the tag groups (e.g., narrative, content source, visual style) intuitive and helpful for sequencing and structuring their process and thinking more explicitly about presentation-related aspects: *"I think that narrative is super cool in terms of thinking about the audience [...] it helped me be more explicit about who my audience is in the deck that I'm creating. And it helped me make the right deck faster."* (P11)

Users expected to define their own tag groups. However, other users questioned the necessity of pre-defined groups, suggesting the ability to customize or even work without tag groupings, which might better align with their workflow, as their needs can vary between different slide decks: *"But also I would want the ability to just make my own [groups] because I think between different slide decks, it would change"* (P08)

Users needed a moment to get used to tags' flexible terminology. Some participants experienced initial uncertainty around the terminology of intent tags, particularly when classifying and naming tags or choosing appropriate groups: *"So I found it slightly intimidating at the beginning because there is some kind of arbitrary way of classifying."* (P09) However, with experimentation and practice, they realized these decisions were not critical as the system does not rely on precise syntax but allows for flexible ways of specifying tags: *"But after doing it for a while, it seems like it's not that important either. Like, this decision of what to put in the title and this kind of things... As far as I'm putting text, I figure that it does something."* (P09)

Users appreciated the preview features. Many appreciated the preview feature as it provided a sense of security and confidence when using the system. Knowing that they could see a preview before finalizing their choices reduced the fear of unexpected outcomes but also helped their imagination: *"I thought that the slider was neat. [...] I do appreciate how I'm not just imagining, like on a scale from one to five, how much detail do you want? I don't know... So, I appreciate that it has a little preview down there. That's neat."* (P05)

Users expressed a desire for predefined tags. Some participants also noted that starting with a blank tag canvas could be overwhelming, so having a set of predefined tags (such as audience, narrative style, and number of slides) would provide guidance and help users get started: *"Let's say empty tags where there's maybe already kind of like a label... like audience, but there's no audience mentioned yet. But for some people, they don't think about audience before they make a PowerPoint. So, just to be like, hey, who is this for? Is sometimes the most helpful feedback you can give someone. So you can get it here."* (P05)

Users expected to rank tags. Some participants also expressed the need for a feature that allows them to rank or prioritize tags based on their importance to the presentation. One participant (P03) mentioned how they tried to emphasize certain points by grouping similar tags together, such as audience and objective, and incorporating system-suggested tags. They also noted that some tags, like colors or themes, were less important to them.

Supporting Diverse Presentation Needs and Workflows

In the interviews, participants also reflected on their general professional slide-creation experience and how IntentTagger could enhance or alleviate their process. Many described the challenges of traditional tools where the manual slide-by-slide process of creating a deck often feels tedious and uninspiring: *"This was a really fun way to create a deck. I think it would appeal to most people that work in PowerPoint, because I don't think most people in PowerPoint creating decks just don't enjoy it unless they do."* (P02)

Users valued the system for overcoming 'blank page' barrier. Many participants found the system particularly helpful in overcoming the initial challenges of starting a presentation, which they identified as one of the hardest parts of any creative project. The ability to quickly generate a basic draft allowed them to move past the initial "blank page" barrier, making it easier to start thinking about additional content and refinements. This was especially valuable for those who often start presentations from scratch, as the system provided a visual outline that lowered the barrier to getting started, allowing them to focus more on editing and refining the content: *"... I feel like the system got the basics going, which then prompted me to think of additional things to add on top."* (P05)

Participants appreciated IntentTagger's flexibility in supporting diverse workflows. Participants appreciated the system's ability to support various workflows, whether by starting with a detailed outline, a brain dump, or a narrative-first approach. Some participants highlighted the system's flexibility in allowing them to begin with content and then refine it iteratively, while others valued the ability to focus on narrative and then move on to visual elements. The system's adaptability to different working styles, such as sequential versus big-picture thinking, was also noted as beneficial.

Participants appreciated the outline editor for structuring ideas and overview. Additionally, the Outline Editor was particularly praised for helping users map out their ideas and providing a clear presentation overview, making it easier to refine individual slides or the entire deck as needed. One participant (P06) also preferred a more sequential workflow, suggesting that instead of showing all tag groups at once, they could focus on one at a time.

Users saw value in tags for tailoring presentations to different purposes. Furthermore, some users mentioned the system's tag-based approach could be useful for creating different versions of presentations tailored to specific audiences or contexts. The ability to easily drag tags in and out allowed for quick adjustments, making it convenient to include or exclude certain content based on the needs of a shorter talk, a quick checkup, or a presentation adapted for a particular audience: *"Sometimes I need to make different versions of presentations [...] This tag system of dragging things out would be an easy way to include and exclude some things if I needed to."* (P03)

Participants appreciated the fluid integration of deck-wide and slide-specific edits. Participants valued the system's ability to manage both the entire deck and individual slides, appreciating the flexibility to make broad changes across the deck or to focus on specific slides for detailed adjustments. This dual approach was seen as more efficient compared to traditional slide authoring workflows, where changes often have to be made slide-by-slide: *"I think that having this create the whole deck in one shot is super time-saving, and I don't expect it to be hit in one shot. And having the option to click into each slide, and if I like what I do here and want to apply it to the rest... I like the ability to go from everything, update everything, zoom in on one, and then experiment on that one. And then again, update all."* (P02)

Users demanded more control by locking specific design elements. While they appreciated the flexibility and ambiguity offered by the system, participants expressed a desire for more control over certain aspects of their presentations by being able to lock or pin specific slide elements (like background colors, fonts, or other design choices) once they were satisfied with them to prevent them from changing unexpectedly during further generation cycles: *"I do love the slider. I only just don't love that if I slide it one way and I get the color I want that it can subsequently change. I want to be able to anchor it."* (P11)

Meta-Intent Elicitation: Helping Creators Figure Out What They Want and Need

One of the key benefits highlighted by participants was the system's ability to help them discover and refine their intentions when creating presentations. Many mentioned that in their daily work, the process of selecting the right content and design elements was often difficult without clear guidance or inspiration. The system's suggestions helped participants think more critically about their choices and provided new ideas they might not have initially considered. This process of "meta-intent" elicitation allowed users to identify important elements that they may have overlooked or not realized they wanted to include, making it easier to achieve their desired outcome: *"I thought it was cool and interesting because it suggests things that I just didn't think about... And I like these suggestions because I don't always know what are the best style choices for something." (P03)*

Participants appreciated the system guidance's gradual granularity Participants appreciated the system's converging granularity through its suggestions, allowing them to gradually refine their presentations in more specific and thoughtful ways:*"I feel like [IntentTagger] suggests a lot more things that I can control. Versus with [the chat-based system], there are like four prompts and they're like buckets of things you can do rather than the specific things you can tweak. And so for me, this system of these tags feels like I can get more granular towards some kind of vision that I want because I think inherently I'm making some of those choices."* (P12) This guidance made the process more engaging and aligned with their vision while also simplifying the decision-making process by presenting contextually relevant options and ideas to respond to: *"It kind of breaks down those decision points that I'm doing sort of subconsciously in a way that is fairly easy. And, like, it suggests a lot of different things [...] it feels like I can get to a decent outcome fairly quickly."* (P12)

Participants imagined Intent Tagging to be beneficial for prototyping presentations and initial brainstorming. For some participants who reported frequently struggling to organize their thoughts or create structured presentations, the system could also serve as a useful tool for brainstorming and organizing their thoughts. Through the drag-and-drop interface, participants imagined they could quickly include or exclude elements, allowing them to organize and prototype their presentations more intuitively and efficiently: *"I can organize my thoughts down and drag things in I want to*

include, drag things out that I want to exclude." (P03)

Some users felt overwhelmed by visual clutter and suggestions. However, while the majority of participants found the amount and visual arrangement of the system-suggested tags helpful, a few noted that too many suggestions could feel visually cluttered and overwhelming. Some expressed a preference for a more simplified or list-based representation, allowing them to focus more easily on the suggestions without feeling overwhelmed by the options: *"...so a bit more visual hierarchy, maybe less overloaded with suggestions."* (P01)

Participants praised the slider for stimulating their critical thinking. Additionally, features such as the opposite slider helped participants think more deeply about the intention behind their choices. By offering a way to explore different dimensions of a tag or design element, participants were able to visualize various possibilities and fine-tune their decisions in a way that aligned with their goals and encouraged reflection on the impact of their choices: *"I really like the slider idea of like... 'oh, you have this idea, but then you want to see what the opposite end is,' and you can put a slider on that value [...] it just makes you think more about your intention of what you're trying to do."* (P04)

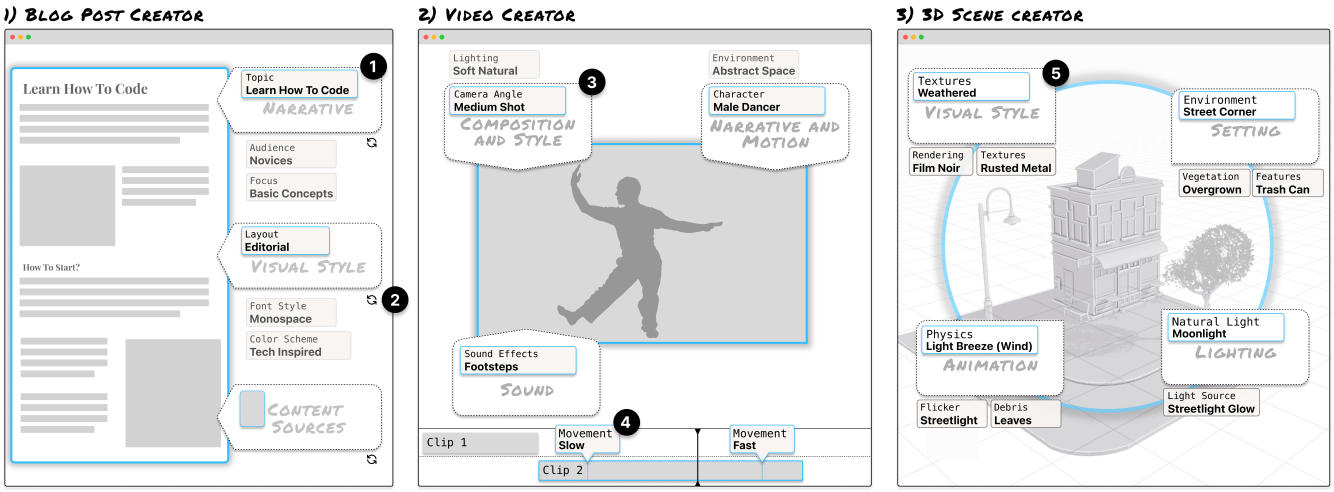


Figure 6.6: UX interface sketches illustrating alternative use-cases for Intent Tagging-based steering of GenAI content creation; (Left) Blog post creator: (1) Steering of a post generation through intent tags in a side panel; (2) new tag suggestion can be requested per group; (middle) Video creator: Steering of a selected video clip through intent tags; (4) additionally, Keyframe Tags allow to control temporal aspects of the clip generation in the timeline view; (Right) 3D scene creator: (5) Steering of a 3D scene generation through intent tags.

6.7 Intent Tagging Interactions Beyond Slide Deck Creation

To illustrate how intent tagging could be utilized to facilitate human-GenAI co-creation interactions across diverse rich content creation tasks beyond slide deck creation, I explored alternative **user experience (UX) interface scenarios** for rich content creation applications. Some of these were inspired by participants' speculations during the interviews on how IntentTagger could be helpful in applications like Excel for complex data visualizations, educational materials, and tasks such as creating videos or book reports.

I sketched out three UX interface scenarios for intent tag-based steering of GenAI-driven creation of blog posts, videos, and 3D scenes (Figure 6.6). I defined different tag groups for each application, matching the creation task's high-level categories. I also explored how to possibly expand the "tag vocabulary" and their functionalities by, for example, envisioning *Keyframe Tags* to allow control over temporal aspects of clips in the video creator (Figure 6.6-4). Overall, while only sketches, these explorations aim to

highlight the versatility of intent tagging for steering rich content generation across a range of applications and interfaces to be further explored in future research.

6.8 Discussion and Design Considerations

In the following sections, I discuss the findings from the user study and UX scenarios, reflecting on *Intent Tagging as an enabler of non-linear GenAI workflows* and its role in *promoting Human-AI Co-creation and reflection.* Further, I will discuss the challenges of *blending steering interactions alongside manual content editing* and, lastly, reflect on *trade-offs between UI customization and user distractions.* Each section discusses challenges, design considerations, and opportunities for future work.

6.8.1 Intent Tagging Enables Non-Linear GenAI Workflows

My findings demonstrate how IntentTagger was able to facilitate flexible, non-linear, and non-destructive workflows with high degrees of freedom: Once users became comfortable with the tool, they were able to seamlessly switch between the Deck Steering Canvas, Slide Overlay, and Outline Editor and steering the GenAI system without having to follow a rigid sequence (see Section 6.6.2). However, this non-linear steerability comes with a cost: While participants valued the system's flexibility, they also expressed frustration about a need for greater control by "locking" or "pinning" specific content or design elements, such as background colors and fonts, once satisfied to prevent unintended changes during further generation cycles (see Section 6.6.3). While it is a powerful concept to enable multi-modal and multi-directional steering of GenAI content generation between tag, outline, slide, and deck levels, we currently lack interfaces allowing users to gradually lock elements across such dimensions. As a starting point, future research in this direction could be partially informed by design space exploration and parametric design systems that support users navigating and gradually reducing the degree of freedom across large option sets [235, 399].

In terms of user experience, Intent Tagging presents a novel way of steering GenAI systems and creating slide decks that are new to users. My findings showed that while all participants appreciated the flexibility to specify tags with different levels of specificity, many were initially unsure about the tags' correct terminology and group associations (see Section 6.6.2). This uncertainty posed initial challenges, which participants quickly overcame after experiencing the systems' flexibility in interpreting their intent

after the first generations. To better support user onboarding, I suggest that future intent tagging-based interfaces should more explicitly scaffold that learning process, for example, by partially pre-defining common tag labels (such as *"audience"* or *"length"*) or offering optional "tag templates" for specific content creation scenarios to help users better get started and learn intent tagging.

6.8.2 Intent Tagging Promotes Human-AI Co-Creation and Reflection-In-Action

Figuring out one's intentions and requirements is often the hardest part of complex tasks like slide deck creation. Generally, *Schön* describes this process as "reflection-in-action" [320], where creators interact with the material at hand while the material "talks back" to them, creating an iterative process of acting and reflection. When intent tagging, the tags on the canvas talk back to the user, whether self-generated or system-suggested, helping them (along with the outline and slides) tackle the cognitively demanding task of clarifying their intent. Furthermore, the more task-relevant tags a user specifies, the more accurately the system can provide contextually tailored suggestions, better aligning with the user's objectives. As users add more tags, they convey a fuller picture to the GenAI system, which in turn enhances the quality and precision of the generated content. The system and user engage in a close feedback loop, allowing users to refine their intentions while the AI system responds dynamically—like connecting a mind map to a GenAI content creation system. This process relates closely to previously framed "enactive" models of Human-AI co-creation [79], relating to the theory that cognition arises from improvised interactions shaped by feedback from the surrounding environment.

In this version of IntentTagger, I decided that users must explicitly request tag and image suggestions from the system. On the one hand, this approach ensures persistency between requests, allowing users to explore suggested tags without the risk of suggestions disappearing without their consent. On the other hand, tag suggestions also offer an exciting possibility of being combined with mixed-initiative interactions in the future, where the system can automatically initiate tag suggestions in response to the user's active tags [214].

In terms of user-system communication, in this study, I focused on analyzing users' general interactions with system-generated tags (see Section 6.6.2). Future work could delve into a deeper semantic analysis, exploring how users expressed their intent and

what specific tags the system suggested in response. Such insights could further improve the effectiveness of tag-based human-AI co-creation systems.

Furthermore, while many participants experimented with the *Opposite Slider widget*, no one used it to steer slide generation (see Section 6.6.2). However, several participants voiced the clear value of this feature for prompting reflection on their design choices (see Section 6.6.3). This suggests that, while not used directly for steering generation, the slider still serves as a valuable tool for encouraging and steering deeper thinking. This bears interesting further implications for designing future GenAI steering interfaces, which might consider integrating steering controls alongside "intent exploration" UI to promote reflection independent of content generation.

These findings also revealed that users often got inspired by system suggestions but then manually overrode the tag's value field to better align with their vision. This interaction pattern hints at an effective combination of guided intent exploration through system-generated options alongside the flexibility for users to override suggestions if desired. Adding to this observation, a recent study in the context of AI-assisted writing found that participants, despite the many integrated AI suggestions, felt ownership over the content *"because of the numerous authorial decisions"* they made [335]. Together, such findings suggest that it is critical for effective human-AI co-creation to enable users to manually edit and override system suggestions to foster a sense of ownership and control.

My research indicates that user interactions with IntentTagger's LLM-generated *tag suggestions*, *tag grounding*, and *pre-generated real-time previews* can help align user expectations with GenAI systems' capabilities and recognize their limitations (see Section 6.4). While some suggested tags referred to aspects beyond the slide generation capabilities (such as icons or background textures), interestingly, users were not significantly frustrated by that. Instead, they found the suggestions helpful for improving their presentations, regardless of whether the system could fully execute them. This tolerance was likely influenced by the pre-generated real-time tooltip previews, which helped users quickly grasp the system's limitations without frustration, as they had not invested much time in these explorations.

Adding to previous work on surfacing affordances and providing feed-forward mechanisms [42, 352, 366], I promote dynamically **pre**-generating slide previews (or other content) on a granular level (such as only alternating a slide's font or text format) to support users' intent exploration and foster greater tolerance when GenAI falls short by

reducing emotional investment in misaligned expectations. Additionally, *tag grounding*—such as the grounding from text prompt or slide features—might also improve alignment between user expectations and system capabilities by helping users quickly understand how the system interprets their inputs. Lastly, presenting distinct differences between versions of generated content may further enhance reflection-in-action during the creation process, building on insights from Drucker et al. [98]. All these mechanisms show promising potential for enhancing user alignment in GenAI workflows.

6.8.3 Blending Steering Interactions and Manual Content Editing

The IntentTagger prototype system is designed to probe the concept of intent tagging-based interactions for steering slide deck generation and, therefore, does not support rich content editing that full-fledged applications such as PowerPoint offer. In the study, the system allowed participants to create slide decks with GenAI support with more control and satisfaction than with the chat-based system. However, going further, I envision that IntentTagger would also integrate rich content editing capabilities of slides in addition to AI-driven generation. Enabling users to interweave manual and AI-driven editing activities opens up many interesting questions relating to the orchestration of the hand-off between user and system. For example, devising mechanisms for identifying situations in which steering GenAI actually becomes more tedious than directly editing content and surfacing cues to users nudging them towards direct editing is an intriguing direction to explore.

6.8.4 Trade-offs between UI Customization and UI Management

Lastly, I reflect on requests I gathered from participants to enable them to customize IntentTagger more. A salient example is the requests from a few participants for enabling customizable groupings of tags rather than the three fixed groups the current system offered (see Section 6.4.2). While relatively straightforward to achieve, such added capability may come as a tradeoff between user control of the interface and its management.

While giving users the ability to create their own tag groups' structure would allow for greater flexibility—especially for catering to users' specific needs and creation process—it would also add an additional cognitive layer by requiring users to reflect

on what such process is and how to best articulate it in the interface. This leads to the classic question of identifying which interface aspects are best given to the users to shape and which to leave in the hands of design and system architects. Probing this delicate balance between supporting flexibility and minimizing effort, enabling rich interactions, and making them simple to use in the context of generative AI paves the way for future research in our field.

6.8.5 Limitations of the Evaluation

I highlight three limitations of this work: Firstly, all participants were recruited from the same large technology company, and while they represent a diverse set of job titles and backgrounds, it is possible that participants from other companies or from non-corporate environments might provide additional insights not captured in the current study. Secondly, the study's lab setting, limited task times, and constrained number of slides allowed me to study Intent Tagging in a controlled and concise format. However, future work is needed to study the impacts of Intent Tags on creation processes *in-vivo* and over *longer periods of time* to gain further insights into real-life content creation scenarios. Lastly, I deliberately limited the comparison of IntentTagger with the widespread commercial Copilot and Designer features in PowerPoint. Future comparisons with other tools and interfaces may surface a broader range of benefits and tradeoffs.

Chapter 7

MindKicks: Pluggable Cognitive Sidekicks for Enhancing AI-Driven Workflows

7.1 Overview

Throughout the previous chapters, the projects in this dissertation have shown that interfaces designed to support metacognition can help people work more effectively with Generative AI systems. Across these projects, interactive cognitive support helped users notice tacit assumptions, explore alternatives, and better articulate their goals, which, in turn, led to more satisfying interactions and better outcomes with GenAI systems in design and content-creation workflows. However, these projects, alongside other recent research in that area [160, 400], have also surfaced recurring broader challenges and design tensions that can dampen the effectiveness of such cognitive supports. Here, I specifically highlight three of these challenges alongside underexplored design opportunities:

Challenge 1: Support move selection
Providing effective cognitive support is highly context-dependent as it varies by user, task, and moment. This makes it often hard to determine the most effective support move for a given situation.

Design Opportunity 1: Presenting Parallel Support Strategies
Instead of trying to infer one single support move, what if we surface several different cognitive support strategies and let users choose what is useful? Thus, users might benefit from different support moves in parallel, while retaining agency over which ones to pick up.

Challenge 2: Proactive support often conflicts with user attention
Effectively delivering proactive support is challenging, and past research on proactive cognitive support has repeatedly struggled with this. Major design tensions especially revolve around user attention: Proactive support can easily interrupt users' workflows, but when it is too subtle, it can get ignored. At the same time, when too much support appears at once, it can often overwhelm users.
Design Opportunity 2: Adaptive UI
As a strategy to better fit into users' attention, can we better utilize adaptive UI mechanisms (such as inline cues, progressive disclosure, ranking, and filtering) to design proactive support that is noticeable and situated while maintaining an effective balance between being too distracting, ignorable, or overwhelming?

Challenge 3: Workflow-specific Support
Current cognitive support systems are often domain- or tool-specific, like SocraBot for mechanical design or IntentTagger for slide presentations. However, this causes fragmentation and application silos.
Design Opportunity 3: Cross-workflow Modular Support
Rather than building separate support systems for each tool and domain, can we instead build universal cognitive support systems that plug into existing tools and interface affordances?

Motivated by these three challenges and design opportunities, this project explores how existing AI-assisted workflows can be augmented with multiple cognitive support strategies in parallel, broadening users' thinking without overloading them. To investigate this more specifically, I developed *MindKicks*, a plug-in architecture for integrating proactive, context-aware cognitive "sidekicks" into users' existing AI-driven workflows and host applications. To further demonstrate this concept, I instantiate MindKicks in an AI-assisted data analysis computational notebook environment, where cognitive support sidekicks complement an existing chat-based coding agent interface by surfacing potential pitfalls, alternatives, Socratic questions, reframings, and best-practice prompts. In a between-subjects study with data analysts, I found that MindKicks significantly increased analytical AI-engagement and helped participants identify more underlying data issues. At the same time, the study also revealed trade-offs among perceived cognitive overload, deference to AI, user control, and design considerations for integrating MindKicks into chat-based coding agent workflows. Overall, this chapter contributes a modular architecture for embedding cognitive support into existing AI-assisted workflows, a data-analysis support system MiKi$^{\texttt{Data}}$ to augment chat-based

coding agent interfaces, and empirical insights into how proactive cognitive sidekicks can augment existing AI-assisted expert workflows. Besides this specific instantiation in the context of data science, I propose MindKicks as a universal plug-in architecture for designing and integrating proactive cognitive support sidekicks into other kinds of applications and domains. Therefore, to demonstrate this generalizability, I furthermore provide an outlook on how MindKicks can be used to create cognitive sidekicks in other GenAI-assisted applications, such as for mechanical design and product planning.

7.2 Background

7.2.1 Supporting Cognitive Engagement in AI-Assisted Programming and Data Analysis

Today, in many programming and data analysis tasks, generative AI assistance is widespread, ranging from inline code completion to chat-based coding agents and AI-augmented authoring systems for visualizations [62, 385]. In these AI-assisted workflows, analysts and programmers increasingly shift from directly authoring code toward instructing, steering, interpreting, and validating code and analyses generated by AI assistants [137, 208]. However, while AI systems can often accelerate production, they also move important cognitive work into activities such as formulating intentions, specifying context, monitoring AI outputs, and verifying whether AI-generated artifacts actually satisfy the user's goals.

Prior work in **AI-assisted programming** documents this tension between productivity and cognitive engagement. A recent survey shows that developers value AI programming assistants for reducing keystrokes, accelerating task completion, and helping recall syntax, but also report difficulty controlling assistants and obtaining outputs that satisfy functional and non-functional requirements [208]. Similarly, AI code recommendations introduce non-trivial cognitive work around reading, evaluating, and integrating suggestions [251] and agentic coding assistants can reduce software engineers' cognitive engagement [58]. Similar concerns arise in **AI-assisted data analysis** where GenAI can support the information foraging and sensemaking process, but also introduce new cognitive barriers around query formulation, context specification, and result verification [97].

Overall, as discussed previously in this thesis, a growing body of HCI work frames

these difficulties as **cognitive and metacognitive demands created by GenAI systems**, with GenAI shifting substantial cognitive load onto users' metacognition—including planning, monitoring, evaluating, and regulating reliance—while often providing little direct support for such cognitive processes [109, 349].

Subramonyam et al. [339] have framed this cognitive demand as "gulf of envisioning" in prompt-based interaction with GenAI systems: users must translate goals into actionable intentions, determine how to instruct an LLM, and anticipate what output would satisfy their goals. Studies have also found that LLM usage can lead to homogenization in programming and data analysis [204, 315].

In response, researchers have called for designing GenAI interfaces that can better support metacognition and critical thinking [58, 109, 130, 314] and have begun **designing interventions that promote critical thought, verification, and reflection**. For example, prior work shows that cognitive forcing functions can reduce overreliance on AI recommendations [47] and that step-by-step interactive dialogue and scaffolded task decomposition can foster deeper engagement and improve steering and verification in data analysis [172]. Similarly, AI-generated provocations, such as brief critiques and alternatives attached to AI suggestions, can promote critical and metacognitive thinking [96], and showing contextual fairness and bias notifications in JupyterLab can nudge data scientists toward more careful treatment of protected classes, proxy variables, and demographic performance differences [144]. In the context of LLM-assisted programming, explicitly stating possible underlying assumptions and offering different possible approaches helps users explore broader alternatives [400].

Taken together, prior work shows that effective AI assistance in programming and data analysis relies on the design of interfaces that help users form intentions, steer assistance, verify outputs, reflect on alternatives, and remain cognitively engaged throughout the workflow. However, existing cognitive support systems are typically designed to support users in domain-specific tasks, workflows, and tools while often relying on a single support strategy. The work in this chapter builds on and extends prior work by providing a generalizable system architecture for cognitive support sidekicks that can run multiple support strategies in parallel, thereby complementing and enhancing existing AI-assisted workflows across domains, including programming and data science, with proactive cognitive support.

7.2.2 Parallel Support and Multi-Agent Systems

As highlighted in the previous section, cognitive support can take many forms, for instance, questioning users' assumptions, proposing alternatives, prompting verification, or offering critique [47, 96, 172, 400]. At the same time, determining which cognitive support strategy is most helpful in a given moment is highly context- and task-dependent, and reliably inferring a single best intervention remains challenging in open-ended workflows [12, 155, 160]. On the flip side, rather than committing to a single intervention, presenting users with multiple alternatives in parallel can likewise have benefits. For example, in the context of design, parallel prototyping has been shown to improve design outcomes and self-efficacy [95]. Building on this principle, several interactive systems have been developed to support the generation and comparison of alternatives. For example, *Juxtapose* [145], a code editor and runtime environment for exploring multiple interface alternatives, or *Scout* [344], a system to support rapid exploration of design layout alternatives through mixed-initiative interaction with high-level constraints and design feedback. Similarly, related work on variation and example-based ideation emphasizes maintaining distinct exploration branches intact before selecting or combining them [94, 329, 353]. Recent LLM-based multi-agent systems extend this idea by giving agents different roles that provide complementary perspectives. For example, agent-based systems have been proposed to support co-creative interaction [211, 301], brainstoring and ideation [121, 128, 292], and tackling product design problems [387]. However, studies also show that presenting output from multiple agents can add substantial coordination and interpretation burdens for users and lead to information overload [254, 400].

Taken together, prior work has explored how users can be supported through interfaces and agents to promote exploration of alternatives in design-related contexts. However, little existing work has explored how to offer multiple *cognitive* support strategies in parallel within the broader context of AI-assisted workflows. Grounded in and expanding prior work, this chapter investigates how systems can be designed to offer multiple cognitive support strategies in parallel and examines their impact on users in the context of AI-assisted data analysis.

7.2.3 Design Tensions of Proactive Mixed-Initiative Systems

Proactive interactive systems initiate suggestions based on inferred user needs, extending beyond the reactive interaction model of direct manipulation [325]. Historically, in

the research context on **mixed-initiative systems**, this is often framed as a dynamic allocation of control between user and system, emphasizing reversibility and user authority [10, 12, 155]. However, proactive system behavior also creates central **design trade-offs** regarding user attention: Interventions must be noticeable enough to influence users, yet salient or too frequent interventions can interrupt ongoing work and cause task-distracting information-processing burden for users [160, 400]. Conversely, if support is too peripheral, it may preserve users' focus on the main task but be overlooked [53, 237, 238, 296]. Furthermore, research on user interruption shows that disruption depends strongly on timing and task state: Interventions delivered near task boundaries are generally less costly than those presented during cognitively intense activity [2, 161–163].

To address these user attention-related design tensions, prior research on **intelligent and adaptive interfaces** has explored various techniques over the past decades [160]. For example, showing proactive suggestions in *side panels* can reduce obstruction but may also introduce additional shifts in user attention, whereas *inline cues* can preserve proximity to relevant content [22]. Similarly, *progressive disclosure* allows systems to signal relevance while keeping deeper engagement under user control [326]. Furthermore, techniques such as *ranking* and *filtering* can reduce information overload, but adaptive-interface research shows that poor predictions and inconsistent presentation can also undermine predictability and control [112, 113, 122].

Building on prior work on proactive systems, this chapter investigates intelligent and adaptive interface strategies for improving cognitive support systems. Specifically, I investigate how techniques such as inline cues and code highlights can enhance the noticeability and contextualization of cognitive support suggestions without negatively disrupting users' workflow, and how filtering and ranking mechanisms can support prioritization of incoming system suggestions. Within this scope, the chapter offers techniques, empirical insights, and design considerations for augmenting existing AI-assisted workflows with multiple cognitive support strategies in parallel, broadening users' thinking without overloading them.

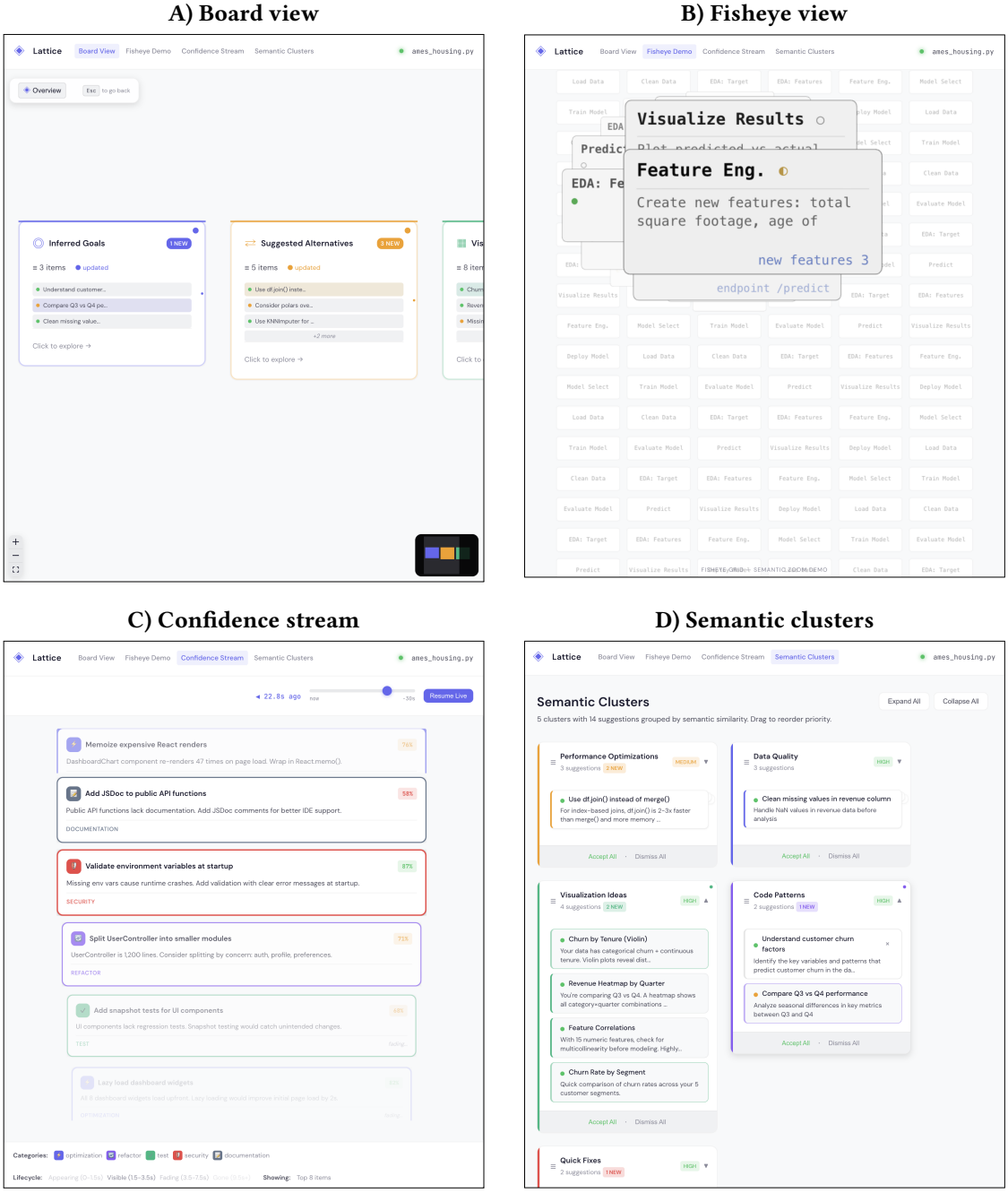


FIGURE 7.1: Four design probes used in formative expert sessions to explore adaptive UI mechanisms for proactive cognitive support: (A) a board view for spatially organizing parallel support streams with semantic zooming/progressive disclosure, (B) a fisheye view for focus+context browsing, (C) a confidence stream for dynamically prioritizing recent items, and (D) semantic clusters for grouping related suggestions across topics.

7.3 MindKicks: A Universal Plug-In Architecture for Embedding Cognitive Support Into Existing AI-Assisted Workflows

7.3.1 Design Rationale

The motivation for *MindKicks* builds on prior work highlighted in the previous section and the insights gained in earlier studies in this thesis. Chapter 4 identified different metacognitive support strategies, such as asking reflective questions, prompting planning and sketching, and surfacing suggestions that helped designers during ongoing GenAI-supported design tasks. These studies showed that such strategies can help users notice assumptions, revisit goals, and recover from flawed task decisions. Still, they also revealed that no single support strategy is uniformly effective across users and shifting task phases. Chapter 5 then demonstrated with *SocraBot* that proactive, real-time cognitive support can be implemented in a live AI-assisted workflow using LLMs, while also revealing how difficult it is to infer the most effective support move from an unfolding user interaction. This led to the central motivation of this project:

> *Instead of committing to one cognitive support strategy, this project explores whether existing AI-assisted workflows can be augmented with multiple cognitive support strategies that broaden users' thinking without overloading them.*

Throughout the previous chapters in this thesis, the cognitive support systems were directly tied to particular applications and domains, such as Fusion 360 Generative Design for mechanical design, or for steering GenAI models, such as Intent Tagger, in the context of LLMs for slide presentation authoring. However, through these prior explorations, the question emerged of how we might create more generalizable cognitive support systems that can augment people's workflows across applications and domains. To achieve such more general support, it first requires separating the cognitive support strategy from the specific task domain and host application. Furthermore, another important aspect of effective support is its contextualization: Previous chapters have shown that, to provide meaningful support, the system must understand the user's current context (see Chapters 4–6). For example, the system must know what domain and task the user is working in, what phase of the task they appear to be in, and what has changed in the evolving artifact; but it must also know how the host application can surface suggestions without overloading users' attention. However, as

highlighted in Section 7.2.3, cognitive support systems face a recurring design tension: Support that is too direct can interrupt users in frustrating ways, while support that is too subtle can remain unnoticed. A side panel can keep suggestions peripheral and easy to skim, but it may be ignored when users are focused on the main workspace; inline or on-canvas cues can make support more situated and timely, but they depend on the host tool's affordances. Therefore, a generalizable cognitive support system should treat general cognitive strategies, user-specific domain context, and tool-specific UI affordances as separable layers, so that a strategy such as Socratic questioning or alternative suggestions can be reused across different domains and task applications such as writing scientific manuscripts in a text editor, developing software in a programming environment, sketching architectural plans on a 2D whiteboard canvas, and designing mechanical parts in a 3D design tool.

7.3.2 Design Principles

Motivated by these challenges, prior work on proactive cognitive support systems (Section 7.2) and the support strategies and systems explored in the previous chapters of this thesis, I began by exploring different adaptive UI approaches for generalizable proactive cognitive support interfaces through iterative design sprints over three weeks. In a parallel-prototyping fashion [95], I developed four alternative concept prototypes (Figure 7.1) and used them as design probes in two semi-structured expert evaluation sessions with senior HCI researchers to evaluate the concepts, refine assumptions, and validate emerging principles. Specifically, the sessions and prototypes explored how proactive cognitive support could be surfaced without becoming either ignorable, disruptive, or too information-processing intensive. These sessions supported three direct design implications: (1) keeping multiple support strategies available in parallel, (2) adding filtering, prioritization, and task-phase awareness so suggestions remain timely and digestible, and (3) combining peripheral overview spaces with situated or inline cues when support needs to be grounded in the user's current focus.

From these collective activities, the following design principles emerged for a generalizable cognitive support system architecture:

DP1 Offering diverse cognitive support strategies in parallel
Different cognitive support strategies help different users, tasks, and moments, but inferring the "right" suggestion is challenging. So instead of trying to infer a single support move, the system should offer a variety of cognitive support

strategies, and users might benefit from different support moves in parallel, while retaining agency over which ones to pick up.

DP2 Providing proactive, context-aware, and governable ambient support
To enable context-aligned proactive cognitive support, agents should monitor the user's ongoing work and surface timely suggestions in the background, while still allowing the user to notice, ignore, dismiss, or prioritize them without losing control of the workflow.

DP3 Fitting into users' tools, workflows and attention
Cognitive support should blend into users' existing workflows and tools while surfacing cues that can be interpreted with minimal context switching. Depending on the user's main application, this may mean a peripheral side panel, an inline code highlight, a notebook-cell marker, a canvas annotation, or a 3D-model annotation.

DP4 Grounded in users' evolving work artifacts and task phases
To make cognitive support targeted and relevant, suggestions should be grounded in what the user is currently making, editing, or analyzing. The system should therefore monitor and reason about changes to the artifact and the user's apparent task phase and ground its support within this context.

DP5 Allowing generalizability through separated support strategy, domain, and tool layers
A general support strategy, such as suggesting alternatives, should not need to be reimplemented for every domain or application. Instead, portability should come from separating the reusable support strategy specification from domain context and tool integration, so that new workflows can be supported by swapping in domain- and host-specific adapters rather than rewriting the support strategy itself.

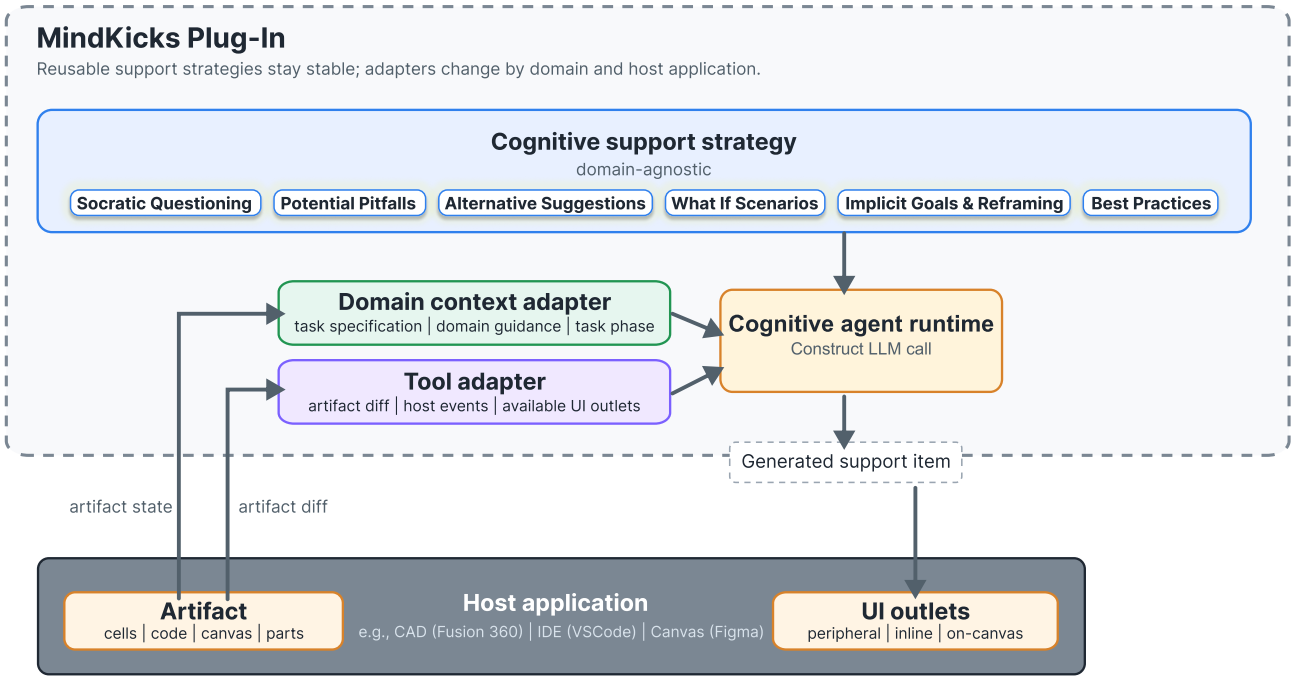


FIGURE 7.2: Diagram of MindKicks plug-in architecture. The host application exposes the user's evolving artifact and available UI outlets. Inside the MindKicks plug-in, the domain context adapter adds task specifications, domain guidance, and task-phase information, while the tool adapter adds artifact diffs, host events, and UI outlet availability. The general support strategies remain domain-agnostic; the parallel agent runtime combines the strategy, domain context, and tool context into an LLM call and routes the generated support item back to the host UI outlets.

7.3.3 MindKicks Plug-In Architecture

From the design principles stated above, I derived a layered plug-in architecture for integrating cognitive *"sidekicks"* into existing software workflows, called **MindKicks** (Figure 7.2). The architecture consists of the following three layers:

Layer I: Cognitive Support Strategy (DP1, DP5). At the top level, each *MindKick* represents a reusable *cognitive support strategy*, such as Socratic questioning, highlighting potential pitfalls, alternative suggestions, or counterfactual reframing. This strategy layer specifies *what kind* of cognitive move the support should make, but not the full details of the user's domain, artifact type, or host interface.

Layer II: Domain Context Adapter (DP4). To provide domain-specific context, the *domain context adapter* layer supplies the domain- and task-specific information that makes these strategies useful in a concrete setting. This includes task specifications, domain knowledge, and a task-phase provider that estimates what the user is currently working on or trying to do. For example, in the context of a computational notebook data analysis task described in the next section, this adapter provides context around data cleaning, exploratory analysis, feature engineering, model evaluation, or result interpretation.

Layer III: Tool Context Adapter (DP3, DP4, DP5). Lastly, the *tool adapter* connects MindKicks to the user's specific host application. This layer exposes the user's evolving work artifact, such as the current state alongside a recent diff, and also declares the available UI outlets through which support can be displayed. For example, in a computational notebook environment, these UI outlets may include a side panel, an inline code marker, and a code-cell-level canvas cue. In other host applications, the corresponding UI outlets might be a design-canvas marker, an IDE annotation, or a 3D-part highlight.

Dynamic Composer Runtime (DP2, DP3, DP5). Finally, the runtime combines and runs all the configured agents in parallel. To generate a new support message, the runtime combines the core support strategy prompt with domain and tool adapter context, then surfaces new support items in UI outlets that match the host application's affordances.

Table 7.1: Cognitive support strategies employed by MindKicks.

	Cognitive support strategy	**Related concepts and theories**
Pitfalls Monitor	Flags assumptions, inconsistencies, or validity risks and asks the user to inspect their consequences.	Metacognitive monitoring [115, 141]; cognitive forcing functions [47]; assumption inspection and verification [172, 349]
Alternative Suggestions	Broadens the user's search space by offering one plausible alternative path, representation, criterion, or approach.	Divergent exploration and parallel prototyping [26, 95]; fixation mitigation through reframing questions [69]; creativity support and design-space exploration [327, 400]
Socratic Questioning	Prompts self-explanation through a focused question tied to the user's current decision.	Socratic questioning [102, 106]; self-explanation and deep-reasoning prompts [74, 135]; reflection-in-action [320]
What If Scenarios	Recasts the situation under a plausible alternative condition to expose hidden assumptions.	Counterfactual thinking in design [269]; alternative mental models in analysis planning [136]; reflective move experiments [320]
Implicit Goals & Reframing	Names an implied goal or frame and invites the user to revise or confirm it.	Problem framing [89, 187]; sensemaking and mental models [178, 283]; implicit decisions and goal alignment in AI-supported work [136, 400]
Best Practices	Surfaces a relevant practice, checkpoint, or norm that can help the user structure the work.	Scaffolding [383]; self-regulated learning [29, 273]; task decomposition and planning guidance [172, 257]

7.3.4 MindKicks' Cognitive Support Strategies

MindKicks provides six cognitive support strategies (see Table 7.1): *Pitfalls Monitor*, *Alternative Suggestions*, *Socratic Questioning*, *What If Scenarios*, *Implicit Goals & Reframing*, and *Best Practices*. These strategies were chosen to cover complementary forms of (meta)cognitive scaffolding surfaced across the earlier chapters of this thesis and related work on reflective human-AI interaction: reflective questioning from *SocratAIs* and *SocraBot* (Chapters 4 and 5), planning-oriented and suggestion-based support from *HephAIstus* (Chapter 4), and the broader need for systems that help users monitor assumptions, decompose work, and remain critically engaged with GenAI outputs (see Section 7.2).

7.4 MiKi$^{\texttt{Data}}$: A MindKicks-based Data Analysis Support System

This section describes the design and implementation of MiKi$^{\texttt{Data}}$, a *MindKicks-based Data Analysis Support System*. MiKi$^{\texttt{Data}}$ is a concrete instantiation of the MindKicks framework introduced in Section 7.3. Overall, AI-supported data analysis workflows provide a suitable test bed for studying how to augment analytical open-ended AI-assisted workflows with proactive cognitive support. As in many open-ended creation and problem-solving settings, analysts iteratively seek evidence and refine hypotheses [283]. For example, seemingly small judgments about data quality can propagate and influence downstream analyses and the decisions built on top of them. Furthermore, as of today, LLM-based coding agents are widely used in these workflows, which allows me to explore how cognitive support agents, such as MindKicks, can complement and augment existing LLM-based workflows. Lastly, applying MindKicks in this domain setting also allows me to test whether the cognitive support strategies developed across the earlier chapters can generalize to a design-adjacent domain such as data analysis and programming.

I implemented the MiKi$^{\texttt{Data}}$ support system atop Marimo [5], an open-source web-based Python environment that resembles common computational notebook-based data science workflows (such as Jupyter or Colab) while offering integrated LLM-assisted coding support. Marimo represents a fitting host for MindKicks because it aims to augment existing coding agent workflows rather than replace them: The analyst can work in a notebook environment with chat-based coding agent support available, while

MindKicks runs in the background and surfaces additional cognitive support through dedicated adaptive UI elements. Within this context, designing a MindKicks-based system for data analysis atop Marimo allows me to examine how multiple cognitive support strategies can be orchestrated around a complex, real-world GenAI-assisted task. Before presenting the system features and implementation details, I first walk through an illustrative use-case scenario.

7.4.1 Example Use Case Scenario

Maya is a data analyst at a U.S. state agency. She has been asked to evaluate whether a property housing dataset *(Riverdale County Properties 2006–2010)* is suitable to train a machine learning model to identify potentially undervalued properties for tax reassessment. She opens the dataset in a Python notebook editor. Her usual workflow relies on the editor's integrated LLM coding agent, but today she also has the *MindKicks* side panel enabled, providing additional cognitive support through six different proactive sidekicks.

Inline Notebook Chips: To start, Maya instructs the coding agent to read the CSV and show a preview. As the new cell finishes running, an *inline notebook chip* appears beside it. The *Pitfalls Monitor* has noticed that `district_id` has the cardinality and distribution of a postcode-like identifier and warns that it may act as a proxy for location-based inequities downstream. The chip shows a one-line label (*"District proxy?"*) and expands on click to a short rationale anchored to the cell.

Side Panel Inbox: While Maya scrolls through the data dictionary and writes a few exploratory code cells, a MindKick *Implicit Goals & Reframing* card appears in the side panel: *"You appear to be evaluating whether* `district_id` *is safe as a feature. Have you stated the deployment use case explicitly?"* The card inspires Maya to write a short Markdown cell at the top of the notebook that describes the reassessment scenario, which, in turn, changes the framing of the rest of her analysis. When she later pauses to think, the inbox also surfaces an *Alternative Suggestions* card about checking whether `sale_price` behaves differently across the 2006–2010 time span.

MindKicks Code Generation: Maya clicks the *Alternative Suggestions* card's *Try Me* button. The system generates a new code snippet and shows the resulting plot inside the card. The faceted plot shows a clear shift in median sale price after 2008. Maya clicks *Add to Notebook*, and the code is inserted as a new cell in her notebook.

What-If Suggestion Cards: A *What If Scenarios* card asks: *"What if certain* `sale_condition` *values are concentrated in specific districts?"* Maya writes a small `groupby` and

discovers that one *sale-condition* flag is heavily over-represented in two districts (a data anomaly she would not have looked for on her own).

Inline Code Highlights: Later, when Maya asks the coding agent to fit a baseline regression, and it generates a vanilla `train_test_split(...)` call, the *Pitfalls Monitor* attaches a *MindKicks inline code highlight* to the split call and surfaces an inline notebook chip next to the code cell. The chip explains: *"Given the 2006–2010 span and the apparent post-2008 shift, a random split is likely to leak future information into training."* Maya replaces the random split with a year-based holdout.

Socratic Questioning Cards: As Maya writes a first draft of her conclusion, a *Socratic Questioning* card appears in the side panel: *"What evidence would convince you the dataset is not fit for this purpose?"* Maya pauses, opens a new Markdown cell, and lists three falsifying conditions, two of which she has not yet checked. She does so before committing to a recommendation.

Cross-Card Merging: Two cards (one from *Pitfalls Monitor* about district-level concentration and one from *What If Scenarios* about sale-condition imbalance) refer to the same underlying potential issue. The system merges them into a single card, which now displays a "*merged from 2 cards*" indicator. Maya can undo the merge from the notification banner if she disagrees, but the consolidated card better reflects what she now considers a single concern.

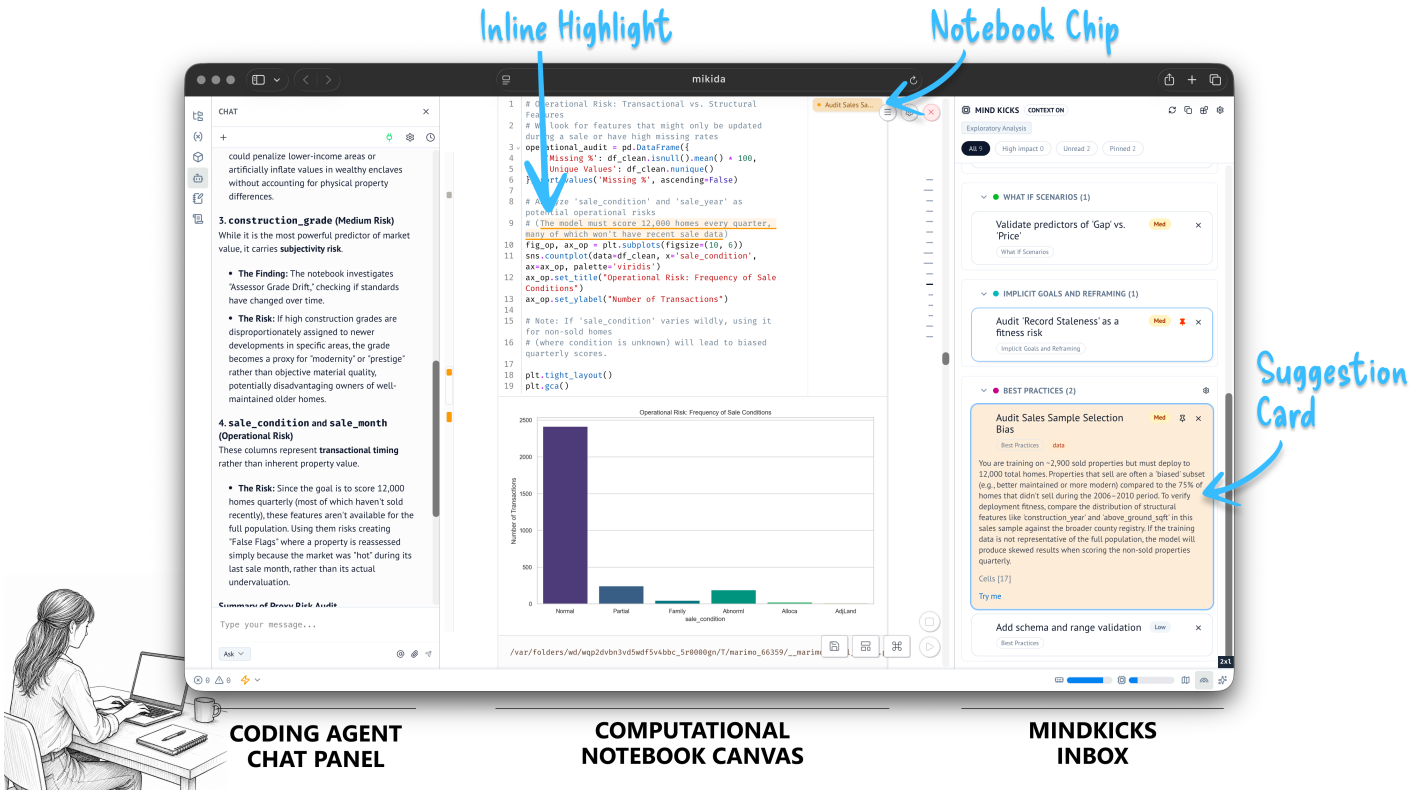


FIGURE 7.3: Screenshot of MiKi$^{\texttt{Data}}$ — MindKicks deployed inside Marimo. The interface combines the notebook canvas, Marimo's coding agent chat panel, and the MindKicks inbox for cognitive support. On the notebook canvas, an inline code highlight and a notebook chip indicate where MindKicks has surfaced suggestions related to the current analysis.

7.4.2 Interface Features

This section provides a detailed explanation of the MindKicks interface and features:

Main Interface Layout

The main interface of the MiKi$^{\texttt{Data}}$ Support System is a three-pane layout (Figure 7.3). The central pane is the computational notebook canvas. The left sidebar contains an integrated coding agent that can edit or insert new notebook cells based on the user's instructions. The right sidebar contains the MindKicks inbox pane, where MindKick suggestions appear.

Side panel MindKicks inbox

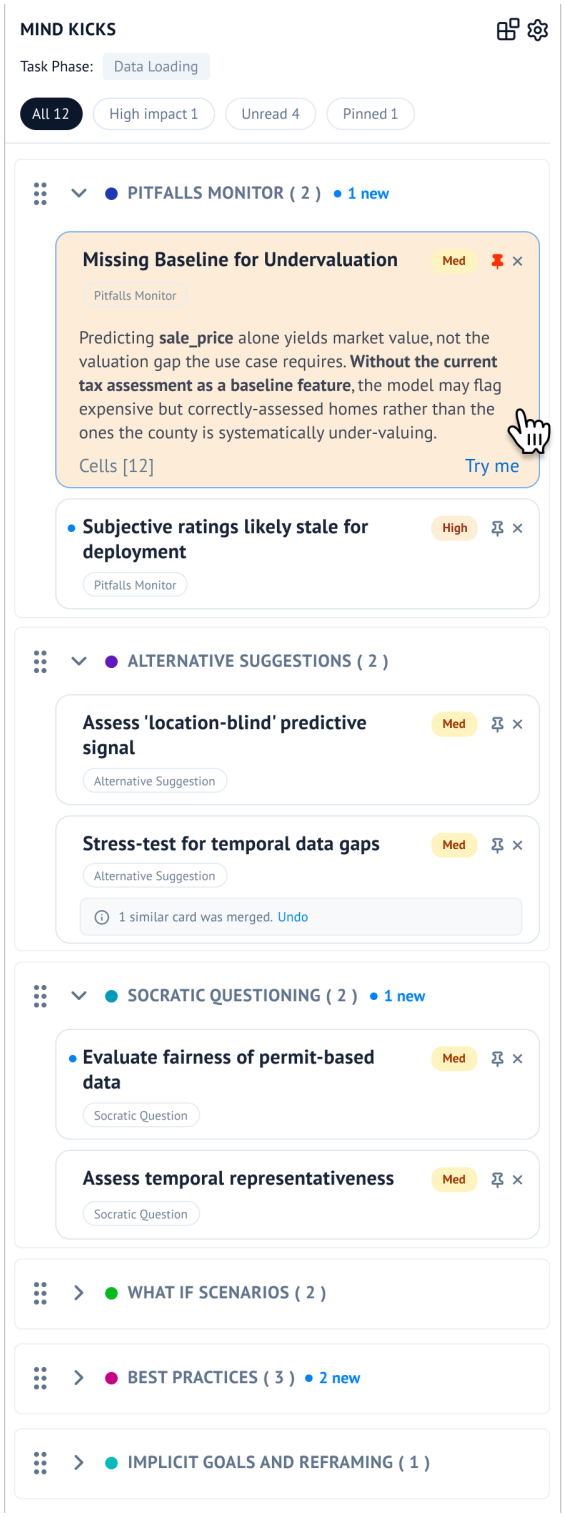


FIGURE 7.4: MindKicks side panel inbox.

The side panel serves as a peripheral inbox for MindKicks-generated support items. Suggestions are grouped by support type, such as *Pitfalls Monitor*, *Alternative Suggestions*, or *Socratic Questioning*, and each group can be collapsed to reduce visual density. The panel header provides filters for high-impact, unread, and pinned cards, allowing users to switch between a broad overview and a smaller set of priority items.

Suggestion Cards

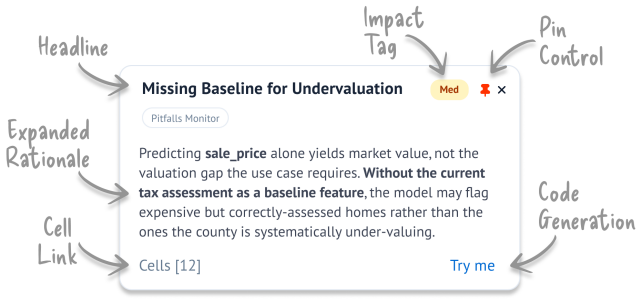


FIGURE 7.5: Structure of a MindKicks suggestion card.

Suggestion cards are the basic unit of MindKicks (Figure 7.5). By default, each card shows a short headline and an impact tag. Clicking the card reveals a detailed description and a follow-up action to generate a code snippet that the user can run (progressive disclosure).

Cards can be pinned to keep them visible or deleted when the suggestion is no longer useful.

Code generation follow-up actions: *Try Me* and *Add to Notebook*

In addition to providing suggestions, cards also allow users to directly operationalize a given suggestion (Figure 7.6). Clicking the *Try Me* action button generates a code snippet to further investigate the card's claim, runs that snippet in the notebook context, and displays the resulting output inside the card. Clicking *Add to Notebook* inserts the generated code as a new cell to the main notebook.

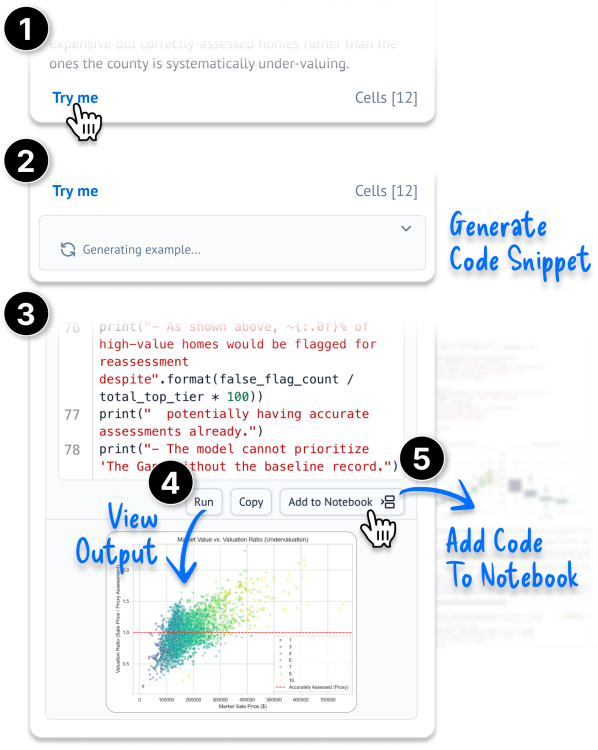


FIGURE 7.6: Card-level follow-up actions for turning a suggestion into executable analysis.

On-canvas UI outlets

For better visibility, each suggestion also appears as a visual cue in the notebook, in addition to the side panel.

A) Inline code highlights.
Inline code highlights provide the most local form of MindKicks output by marking the exact code expression or parameter a card refers to (Figure 7.7). For example, a pitfall related to temporal leakage can point directly to a random train/test split code line rather than just describing the concern in the side panel.

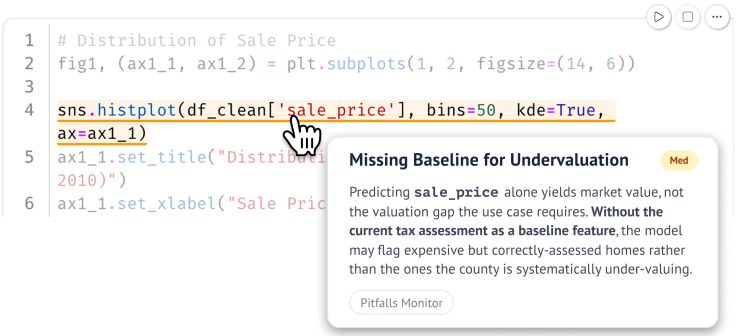


Figure 7.7: Inline code highlight outlet.

B) Inline notebook chips.
Inline notebook chips attach a compact MindKicks cue to a specific notebook cell (Figure 7.8). A notebook chip links an entire cell to a MindKicks suggestion. The chip shows a short label and can be opened to reveal the corresponding card or rationale. Chips can stack up in a dedicated column in the notebook gutter, allowing quick scanning of which cells have associated MindKicks items and jumping from the cell to the side panel for details.

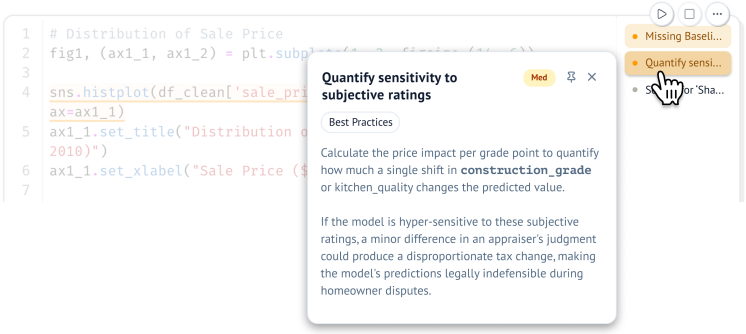


Figure 7.8: Inline notebook chip outlet.

Orchestration mechanisms: Impact-ranking, Merging, Aging-Out

Because MiKi$^{\texttt{Data}}$ runs six MindKicks support strategies in parallel, the interface includes adaptive UI orchestration to improve discoverability of support items. The core underlying mechanism is an **impact-ranking mechanism**: Periodically, all generated cards are automatically ranked by their potential impact level *(high, medium, and low)*, with high-ranking cards shown first in the side panel inbox.

Merging: Furthermore, when multiple cards identify the same underlying issue, such as a pitfall card and a what-if card both pointing to temporal leakage, the system can merge them into a single visible item while preserving the contributing cards' labels[1].

Aging out: Low-impact cards that remain unread and unpinned over time can also age out, with an undo affordance to prevent automatic cleanup from undermining user control.

Implementation Details

MindKicks is implemented as a fork of Marimo [5] with a TypeScript/React frontend. Suggestion generation, orchestration, task-phase classification, and *Try Me* code generation are routed through Marimo's LLM API endpoint. `Gemini-3.1-pro-preview` was used as the underlying LLM for all coding agent interactions and MindKick features.

Artifact Diff (Task Context). To ground its support in the user's current task context, the MindKick runtime receives the current notebook state, a recent diff, and previously generated MindKicks support items, allowing it to reason about recent unaddressed changes rather than repeatedly flagging the same parts of the analysis notebook.

Task Phase Provider. An independent LLM-based task-phase provider periodically classifies the user's current task phase (such as data loading, cleaning, exploratory analysis, modeling, evaluation, or interpretation), which is included in the runtime's LLM call for suggestion generation.

[1] Merged cards include a visual indicator that allows users to undo the merge if desired.

7.5 User Study

To understand the extent to which proactive cognitive support from MindKicks improves chat-based coding agent-assisted data analysis workflows, I conducted a remote between-subjects lab study comparing an LLM-assisted computational notebook workflow with and without MindKicks. Participants worked on a provided data fitness assessment task in which they had to determine whether a property-housing dataset was appropriate for training a machine learning model for a real-world downstream application. Specifically, I investigated:

RQ1 **Outcomes:** *Does MindKicks help discover more dataset issues?*

RQ2 **Process:** *How do people integrate MindKicks into their coding agent-assisted workflow?*

RQ3 **Analytical Engagement:** *How does the integration of MindKicks impact analytical engagement with AI outputs?*

7.5.1 Participants

I recruited 20 participants with data analysis experience across the United States through universities and Prolific. All participants had at least three years of data analysis experience, intermediate or advanced Python skills, prior familiarity with LLM-assisted computational notebook workflows, and experience in training machine learning models. Participants were between 18 and 63 years old *(M = 27.3, SD = 9.5, see Table E.1)*. All participants signed an IRB-approved consent form *(IRB# STUDY2021_00000202)* and were compensated with a $40 Amazon gift card.

7.5.2 Task Design

In the study task, participants were asked to assess whether a given dataset was suitable for training a machine learning model for property tax reassessment in a fictional U.S. county (Riverdale County). See Appendix E.2 for study materials. The task required participants to produce two deliverables: **(1) a computational Python notebook** containing their exploratory analysis and evidence, and **(2) a structured memo** summarizing their findings and overall assessment. The task was designed to resemble a real-world early-stage data science workflow, modeled after familiar data science competitions (such as Kaggle's *House Prices Advanced Regression Techniques*[2]), where

[2] https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques

the central challenge is determining whether the available data should be trusted for a downstream decision-making use case. I iteratively refined the task through consultation and piloting with data scientists to ensure that it was feasible within the session's limited time while still requiring participants to reason beyond apparent predictive performance.

7.5.3 Dataset

The task used the Ames Housing dataset [81] as a foundation, which offers sufficient real-world complexity to support rich exploration and multiple plausible analytical paths. However, for the task, I obfuscated the original dataset as a fictional dataset called *Riverdale County Properties 2006–2010* (see Appendix E.2.2 for the data dictionary). Specifically, I reframed the data around a county property tax reassessment scenario, renamed columns, and injected and amplified specific data anomalies. These changes aimed to reduce participants' and LLMs' potential reliance on prior familiarity with the popular Ames Housing dataset. Furthermore, these changes made the dataset more suitable for studying AI-assisted analysis, because an LLM coding agent could easily infer many surface-level properties from the renamed columns, while discovering other deeper anomaly-related issues would require writing and running dedicated analysis code.

TABLE 7.2: Pre-identified finding inventory used to score participant discoveries. Tier 1 were surface-level findings (column-inferable); Tier 2 were deeper-level findings (requiring computational evidence). †Injected anomaly.

ID	Tier	Analytic work	Finding
T1-A	1	Column-inferable	`district` as proxy / fairness risk
T1-B	1	Column-inferable	`basement_quality` missingness
T1-C	1	Column-inferable	`sale_condition` label contamination
T1-D	1	Column-inferable	Operational unavailability of sale-time features
T1-E	1	Column-inferable	General temporal stability concern
T2-A†	2	Computation-reliant	Assessor grade inflation
T2-B†	2	Computation-reliant	`sale_condition` district concentration
T2-C†	2	Computation-reliant	`lot_area` unit outliers
T2-D	2	Computation-reliant	`renovation_year` redundancy
T2-E	2	Computation-reliant	Quality-block multicollinearity
T2-F	2	Computation-reliant	`district` × `construction_year` confound
T2-G	2	Computation-reliant	Abnormal-sale price discount quantified
T2-H	2	Computation-reliant	Predictive variance estimate

7.5.4 Tiered Finding Structure

To quantify participants' discoveries, I extensively piloted the task and developed a pre-identified inventory of possible findings, organized by the analytic work needed to surface each issue (Table 7.2; detailed description in Appendix Table E.3). **Tier 1 surface-level findings (column-inferable)** were schema-level issues identifiable from the data dictionary, column names, or task framing (e.g., district identifiers as proxies for location-based inequities). **Tier 2 deeper-level findings (computation-reliant)** required running notebook analyses to substantiate. The final inventory included five Tier 1 findings and eight Tier 2 findings.

The task also included a plausible distraction: by instructing the coding agent, participants could quickly build seemingly strong ML models *(e.g., R^2 around 0.80)*, potentially masking unresolved dataset issues. Together, Tier 1 inferability and Tier 2 verification created an environment where participants could appear productive without performing deeper investigations. This design let me test whether MindKicks encouraged broader, more critical issue discovery.

7.5.5 Memo Deliverable

As part of the exploratory notebook task, participants completed a structured memo summarizing their findings in the computational notebook. The memo asked for five sections: **(1)** an issue inventory, **(2)** a severity assessment, **(3)** likely downstream consequences, **(4)** tradeoffs, and **(5)** an overall fitness assessment (Appendix E.2.3), culminating in a recommendation about whether the dataset was fit for the reassessment modeling use case. This deliverable reflects real-world data science reporting and helped distinguish issues participants noticed from those they considered important enough to communicate.

7.5.6 Procedure

All study sessions were conducted remotely over MS Teams and lasted approximately 120 minutes. Participants shared their screens throughout the task, and the sessions were recorded with screen, audio, and webcam capture. Participants were asked to think aloud [363] during the main analysis task to capture their reasoning, their use of the coding agent, and, in the treatment condition, how they responded to MindKicks suggestions. The study was structured into three phases:

1) Onboarding (30 min): At the beginning of the session, participants completed a short tutorial walkthrough and practice task to familiarize themselves with the system. Participants in the MindKicks condition also received an introduction to the MiKi$^{\text{Data}}$ features and were told that they could ignore, read, expand, or act on suggestions as they saw fit.

2) Main task (60 min): Participants then completed the data fitness assessment task while thinking aloud. All participants had access to the same pre-configured starter notebook file, referencing the dataset, data dictionary, and task prompt. Participants in the baseline condition worked only with the notebook and coding agent. Participants in the MindKicks condition additionally received proactive suggestions through MindKicks.

3) Post-task questionnaire (30 min): After the task, participants completed a post-task questionnaire. The questionnaire included Likert-scale questions about their experience with the AI-assisted workflow. Finally, participants provided open-ended voice responses reflecting on their experience, perceived benefits and limitations of the AI support, and how the support affected their analytical process.

7.5.7 Collected Data

Across the study, I collected the following data:

- Screen, audio, and webcam recordings of the main think-aloud task
- Computational notebook files containing participants' code, outputs, exploratory analyses, and a structured memo summarizing participants' findings
- System interaction logs
- Post-task survey responses

7.5.8 Measures and Analysis

To answer the research questions, I used a mixed methods approach, combining task outcome evaluation, video interaction analysis, post-task survey comparisons, and reflexive thematic analysis of participants' open-ended reflections.

Outcome Analysis

To assess what participants discovered during the task (RQ1), each participant's memo and notebook were compared against the pre-identified finding inventory described above (Tier 1 and Tier 2 categories, see Section 7.5.4 and Table 7.2). *Tier 2* memo claims had to be evidence-grounded in the computational notebook, while *Tier 1* findings required a clear statement of the issue and its relevance to the reassessment use case (see Appendix E.3). I used an LLM-as-a-judge pipeline[3], followed by a human audit. The LLM judge prompt was calibrated against five human-audited anchor cases, and novel findings were manually verified by checking whether the claimed data issues actually existed. I compared discovery outcomes between conditions using descriptive statistics, Welch's t-tests, Mann-Whitney U tests where appropriate, effect sizes, and 95% bootstrap confidence intervals with 10,000 replications.

Video Interaction Analysis and Process Measures

To understand how participants worked with MindKicks (RQ2) and how these interactions shaped their analytical reasoning (RQ3), I conducted video interaction analysis [33] using synchronized screen recordings, think-aloud transcripts, and system logs.

[3] `opus-4.7` was used for the LLM-as-judge pipeline, which deliberately differs from the task coding agent's and MindKick's underlying LLM family (`gemini-3.1-pro-preview`).

Table 7.3: MindKicks interaction codes and score weights.

MK Interaction Code	Observed User Action	Score Weight
Glanced	Brief visible orientation to the card, such as a hover or short look (< 2 secs).	0.2
Read	Opened, pinned, expanded, or stayed with the card long enough to plausibly read its content (> 2 secs).	0.5
CodeGen	Requested card-generated code through the card's follow-up action.	0.8
CodeAdded	Added card-generated code into the notebook workflow.	1.0

The unit of analysis was a human-AI interaction event: an interaction with a MindKicks card or the coding agent. For each event, I coded three aspects of the surrounding interaction:

1. **MindKick Interaction Event:** the observed level of MindKicks card interaction;
2. **Coding Agent Request Event:** the participant's coding agent request type; and
3. **Analytical Reaction Event:** the participant's analytical reaction to 1 or 2 (a MindKick suggestion or coding agent output).

Following prior work that treats AI-assisted programming interactions as help-seeking processes with different levels of delegation and inquiry [32, 302], all Coding Agent Request Events were categorized into *execution-oriented* or *exploratory* requests (see full process codebook in Appendix E.3). In total, the analysis covered approximately 20 hours of task recordings, yielding 760 coded events. Based on these coded events, I derived two participant-level process measures: (1) MindKicks Interaction Score (MKIS), which summarizes the breadth and depth of MindKicks card interaction, and (2) AI-Engagement Score (AIES), which captures higher-order analytical reasoning in response to AI output. I keep these measures separate because frequent AI use can be shallow, while even a few AI-mediated events can prompt substantial critique or synthesis.

Measuring MindKicks Interactions (MKIS) To measure participants' frequency and depth of their interactions with MindKicks, I first coded how deeply participants engaged with each MindKicks card (see Table 7.3): from *Glanced* and *Read* to requesting code (*CodeGen*) and integrating it into the notebook (*CodeAdded*). However, participants often revisited the same MindKick suggestions, progressing from glancing at a card to reading it closely, and sometimes generating or adding its code. To separate

breadth (unique cards) from depth (engagement), I kept only the highest interaction level observed for each unique card (with higher levels subsuming lower ones).

Finally, to derive a comparable score for each participant, I then summarized all interactions as a participant-level MindKicks Interaction Score (MKIS):

$$MKIS = 0.2\,N_{Glanced} + 0.5\,N_{Read} + 0.8\,N_{CodeGen} + 1.0\,N_{CodeAdded},$$

where N_ℓ counts cards whose highest interaction level was ℓ (placing the most weight on *CodeAdded*). I chose this interaction score weighting because it enables the comparison of the breadth and depth of interactions with MindKick suggestion cards between participants using a single score. The score gives little weight to cards a user only briefly and lightly engaged with ($Glanced = 0.2$), and gives more weight to deeper engagement by reading a card ($Read = 0.5$), and the highest weight to cards where the follow-up code-generation was used ($CodeAdded = 1.0$). Note that this is an author-defined descriptive index for comparing participants' interaction breadth and depth.

Table 7.4: Analytical reaction codes and inclusion in the AI-Engagement Score.

Reaction Code	Observed User's Analytical Response to an AI Output Event	Bloom Alignment	Part of AIES
Surface	Minimal acknowledgment that an AI output appeared.	–	✗
Observation	Described a result, variable, plot, or suggestion without task implication.	Comprehension	✗
Interpretation	Explained what an output means for the dataset or fitness assessment.	Application/Analysis	✓
Critical	Questioned, qualified, or identified a limitation in an AI output or emerging claim.	Evaluation	✓
Synthesis	Connected multiple findings or evidence sources into a broader assessment.	Synthesis	✓

Measuring Analytical AI-Engagement (AIES) To measure analytical depth in response to AI outputs, I coded participants' reactions to MindKicks suggestions and Coding Agent outputs using an adaptation of Bloom's taxonomy for AI-assisted analysis [13, 38, 96]. Because utterances can mix reasoning types, I coded the highest clearly evidenced reaction level per AI-output interaction. Reaction codes ranged from simple exposure to integrative reasoning (Table 7.4): *Surface* captured mere acknowledgment of an AI output, *Observation* captured description of AI output without task implication, *Interpretation* captured any meaning-making of AI output for the dataset or fitness

assessment, *Critical* captured questioning or qualification of AI outputs, and *Synthesis* captured connecting AI outputs with prior findings or evidence sources.

Notably, AIES counts only analytically meaningful reactions (*Interpretation*/*Critical*/*Synthesis*) and is reported separately for coding agent outputs (AIES-CA), MindKicks cards (AIES-MK), and their sum (AIES-TOT). I explicitly exclude *Surface* and *Observation* because they do not distinguish analytical engagement from passive exposure. Notably, AIES is conditional on AI interactions; analytical work that occurred independently of AI output (e.g., inspecting summaries or notebook results without responding to an AI suggestion) is not captured.

Event timeline visualizations

To gain a clearer visual understanding of the sequence and timing of participants' interactions with MindKicks and the coding agent, I created timeline plots to summarize these events within each study session. The plots were generated from system logs and the coded events using ggplot2.

Post-task questionnaire analysis

To understand how participants perceived the AI-assisted workflow, I analyzed the post-task questionnaire items rated on a 6-point Likert scale. I compared Likert responses using descriptive statistics, Welch's t-tests, Mann-Whitney U tests where appropriate, Hedges' g, 95% bootstrap confidence intervals with 10,000 replications in R [293], and Benjamini-Hochberg adjusted p-values across the shared item family.

Thematic analysis of open-ended reflections

To analyze participants' open-ended reflections on their experience, I conducted a reflexive thematic analysis [44] of their responses, following an iterative inductive coding process. I then clustered codes into themes through affinity mapping and used the resulting themes to contextualize the outcome, behavioral, and survey findings.

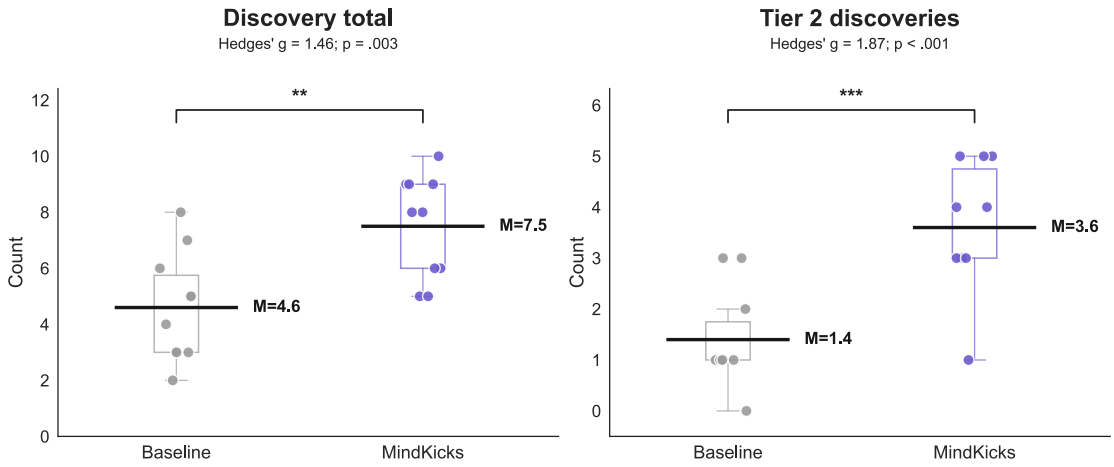


FIGURE 7.9: Discovery outcomes by condition. Each dot is one participant; black bars mark group means.

7.6 Study Findings

7.6.1 How MindKicks Expands Data Issue Discovery and Exploration (RQ1)

MindKicks substantially expanded users' analytical explorations and the discovery of data issues compared to the chat-agent-only baseline group (Figure 7.9). Looking at what analysts discovered, people in the MindKicks group reported 1.6 times as many dataset issues as participants in the baseline condition (Discovery Total: Baseline $M = 4.6$, $SD = 1.96$; MindKicks $M = 7.5$, $SD = 1.84$; $\Delta = +2.9$; Hedges' $g = 1.46$; Welch $p = .003$, Fig 7.9). Notably, this effect was especially pronounced for the deeper, less-obvious data issues (Tier 2), where MindKicks users found 2.6 times more underlying data issues (Baseline $M = 1.4$ ($SD = 0.97$), MindKicks $M = 3.6$ ($SD = 1.26$), $\Delta = +2.2$; Hedges' $g = 1.87$ (large); Welch $p < .001$; Mann-Whitney $U = 90$, $p = .002$; bootstrap 95% CI of mean difference $[+1.20, +3.10]$ (excludes zero)). This suggests that MindKicks helped participants move beyond the more obvious analytical paths and broaden their analytical exploration.

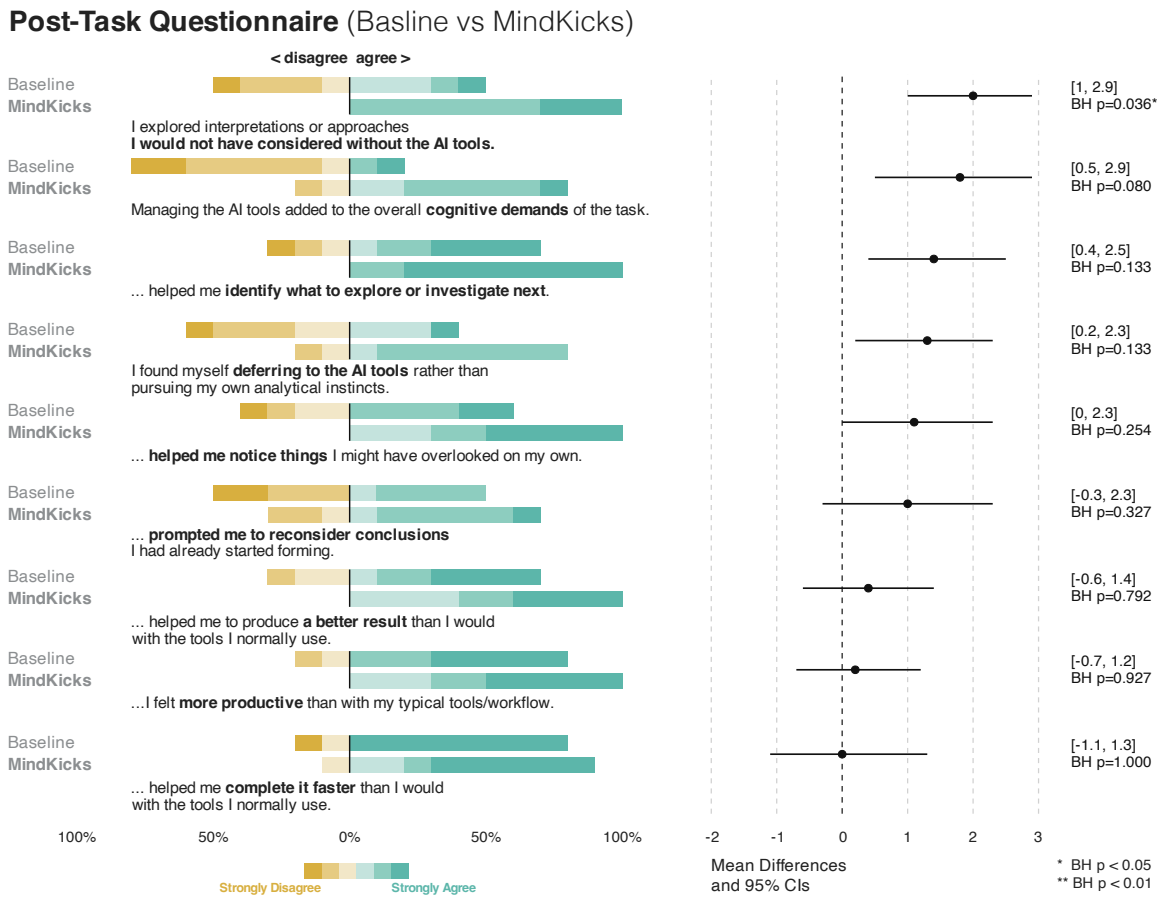


FIGURE 7.10: Post-task Likert comparison between MindKicks and Baseline (Coding Agent only) groups. Left: 6-point Likert response distributions for each condition. Right: MindKicks–Baseline mean differences with 95% bootstrap percentile confidence intervals. Items are ordered from largest to smallest observed mean difference; reported p-values are BH-adjusted Welch two-sample tests.

Participants experienced MindKicks support as beneficial for data issue discovery.

The higher discovery rate in the task outcomes also matches participants' perceptions in the post-task questionnaire (Figure 7.10): Compared to the chat-agent only group, MindKicks users more strongly reported that the AI tools helped them *explore approaches they would not have considered* ($MD = +2.0$, 95% CI $[+1.0, +2.9]$, Hedges' $g = 1.65$, Welch $p = .003$, $p_{\mathrm{BH}} = .036$), *identify what to investigate next* ($MD = +1.4$, $p_{\mathrm{BH}} = .133$) *notice things they might have overlooked* ($MD = +1.1$, $p_{\mathrm{BH}} = .254$), *reconsider conclusions* ($MD = +1.0$, $p_{\mathrm{BH}} = .327$), *overall produce a better result* ($MD = +.4$, $p_{\mathrm{BH}} = .792$) and *felt more productive* ($MD = +.2$, $p_{\mathrm{BH}} = .927$).

MindKicks helped participants maintain the bigger analytical picture. In the post-task reflections, several participants described MindKicks as useful because it helped them to periodically step back from the immediate coding task and keep track of the broader assessment. For example, P27 explained that the system helped because *"it gives [...] the overall picture of [the task] and when you're so much involved in just building out individual features [...] you lose track of the entire picture of what is being done"* (P27). Similarly, P20 described reaching a moment where they *"hit sort of a mental roadblock"* and then finding MindKicks' suggestions more useful than the chat-based interaction: *"when I started using [MindKicks] I felt like the results it was turning up [were] significantly more impactful and influential to the task."* (P20).

MindKicks helped participants notice issues they were not already looking for. Participants also described the suggestions as valuable because they surfaced issues outside their current frame of attention. P03 reflected that MindKicks pointed them toward *"some things that I was not even thinking about regarding the analysis of the data"*, including assumptions that had seemed plausible at first but became more complex after inspection (P03). For example, they initially treated one basement-quality pattern as missing data, but MindKicks helped them see that it *"correlated with not having a basement"* and therefore they *"couldn't just like eliminate that parameter"* (P03). Another participant similarly valued this kind of analytical redirection, noting that the system suggested *"what could be the variables that might be a proxy for demographic variables"* and where the data itself *"might have the bias"* (P22).

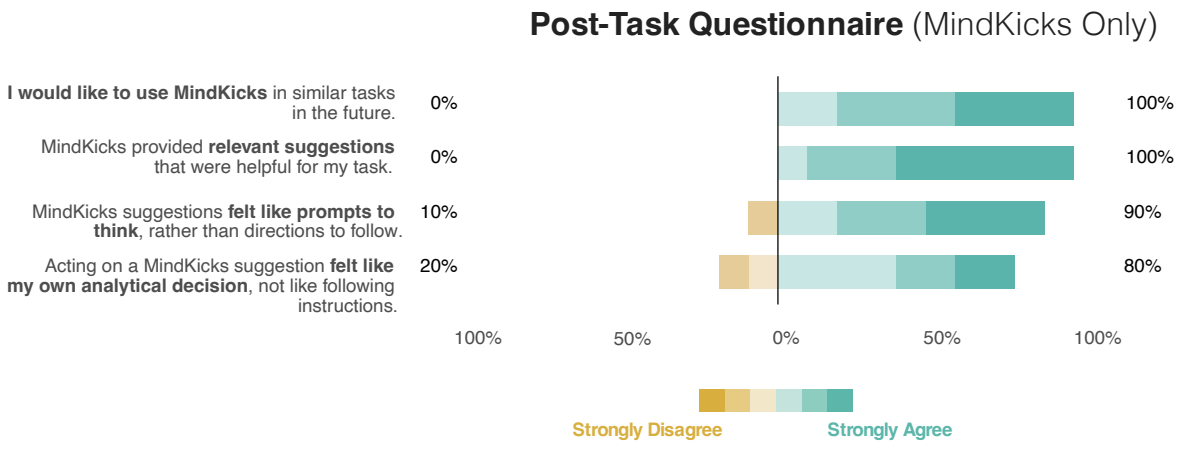


FIGURE 7.11: Ratings of MindKicks-related items. Percentages show the distribution of 6-point Likert responses among MindKicks-condition participants ($n = 10$).

TABLE 7.5: Per-participant process statistics, grouped by condition and by computed usage style/pattern. MK Interaction Intensity columns report card-level engagement (Glanced / Read / CodeGen / CodeAdded). CA Req Types report the count of each request subtype (Execute / Verify / Challenge / Explore Direction / Synthesize), grouped as *Execute* and *Explore* request families. Conditional intensity shading in # CA Req and MK Interact. indicates relative value within each measure; categorical color codes link participants to the MK Integration Pattern and CA Usage Style groups discussed in §7.6.2.

PID	Cond.	Dur. (m)	# MK Cards Gen.	MK Interact.	MK Interactions				MK Pattern	# CA Req	CA Req Types					CA Style
											Execute			*Explore*		
					Glanced	Read	CodeGen	CodeAdded			Execute	Verify	Challenge	Expl.Dir.	Synthesize	
P03	MK	63	132	14.0	8	6	3	7	High Integr.	9	3	2	0	2	2	Mixed
P06	MK	55	100	10.4	9	2	2	6	High Integr.	12	3	3	0	4	2	Mixed
P10	MK	60	133	10.9	17	1	0	7	High Integr.	5	1	1	0	0	3	Mixed
P11	MK	61	94	1.9	2	3	0	0	Low Integr.	20	8	1	3	1	7	Mixed
P17	MK	53	76	7.6	1	2	3	4	High Integr.	12	1	4	1	3	3	Mixed
P27	MK	59	90	10.9	12	7	0	5	High Integr.	9	3	1	0	2	3	Mixed
P18	MK	59	44	2.2	1	0	0	2	Low AI Use	4	3	0	0	1	0	Exec hvy
P20	MK	60	157	13.6	12	6	4	5	High Integr.	16	8	2	0	6	0	Exec hvy
P21	MK	58	168	7.5	15	5	0	2	Low AI Use	4	2	1	0	1	0	Exec hvy
P29	MK	50	97	1.0	0	0	0	1	Low Integr.	16	7	3	0	3	3	Exec hvy
Mean (MK)		*57.8*	*109.1*	*8.0*	*7.7*	*3.2*	*1.2*	*3.9*		*10.7*	*3.9*	*1.8*	*0.4*	*2.3*	*2.3*	
P07	B	59	0	0.0	0	0	0	0	—	20	1	7	2	7	3	Mixed
P16	B	60	0	0.0	0	0	0	0	—	9	3	1	2	2	1	Mixed
P22	B	60	0	0.0	0	0	0	0	—	21	7	2	0	6	6	Mixed
P25	B	50	0	0.0	0	0	0	0	—	9	1	3	0	4	1	Mixed
P28	B	40	0	0.0	0	0	0	0	—	7	0	0	0	2	5	Mixed
P09	B	64	0	0.0	0	0	0	0	—	6	4	1	0	0	1	Exec hvy
P12	B	63	0	0.0	0	0	0	0	—	9	5	1	0	3	0	Exec hvy
P13	B	59	0	0.0	0	0	0	0	—	8	6	1	0	1	0	Exec hvy
P14	B	46	0	0.0	0	0	0	0	—	9	9	0	0	0	0	Exec hvy
P24	B	53	0	0.0	0	0	0	0	—	0	0	0	0	0	0	No-Use
Mean (B)		*55.4*	—	—	—	—	—	—		*9.8*	*3.6*	*1.6*	*0.4*	*2.5*	*1.7*	

7.6.2 How Analysts Integrate MindKicks into Their Workflow (RQ2)

The majority of analysts actively employed and appreciated MindKicks' features—often moving back and forth between MindKicks suggestions and the Coding Agent—and these active users were also the ones who discovered the most data issues overall. On average, MindKicks users interacted with (M = 16.0, SD = 10.1) unique suggestions per session, ranging from 1 (P29) to 27 (P20). The computed MK Interaction Score (the weighted measure of MindKicks uptake and interaction intensity; Section 7.5.8) ranged from 1.0 (P29) to 14.0 (P03), with (M = 8.0, SD = 4.8), see Table 7.5.

When asked about their specific experience working with MindKicks, participants rated it positively (Figure 7.11). They expressed strong agreement that the system provided *relevant and helpful suggestions* ($M = 5.5$, $SD = 0.71$) and that they would *want to use MindKicks in their future analytical tasks* ($M = 5.2$, $SD = 0.79$). Reflecting on using MindKicks, P21 noted: *"I really like the [MindKicks] feature. It was like a live correction and suggestion system, which I do not have as a part of my work yet. So I would really like to [have] it as part of my routine"* (P21).

Similarly, P10 stated: *"I would keep something like [MindKicks], which will give me a head start to think in the directions, especially the what-if scenarios, all that. Having different categories like alternative ways... all that is really helpful to think ahead and start. Nowadays, we have like a copilot or a chat already in place, but this was something different and really helpful"* (P10).

Table 7.6: Overview of participants' MindKicks integration patterns.

Integration pattern	Count	Description	Participants
● **High MindKicks integration**	$n = 6$	High MK interaction	P03, P06, P10, P17, P27, P20
● **Low MindKicks integration**	$n = 2$	Low MK interaction; high CA reliance	P11, P29
● **Low AI use**	$n = 2$	Low engagement with both CA and MK overall	P18, P21

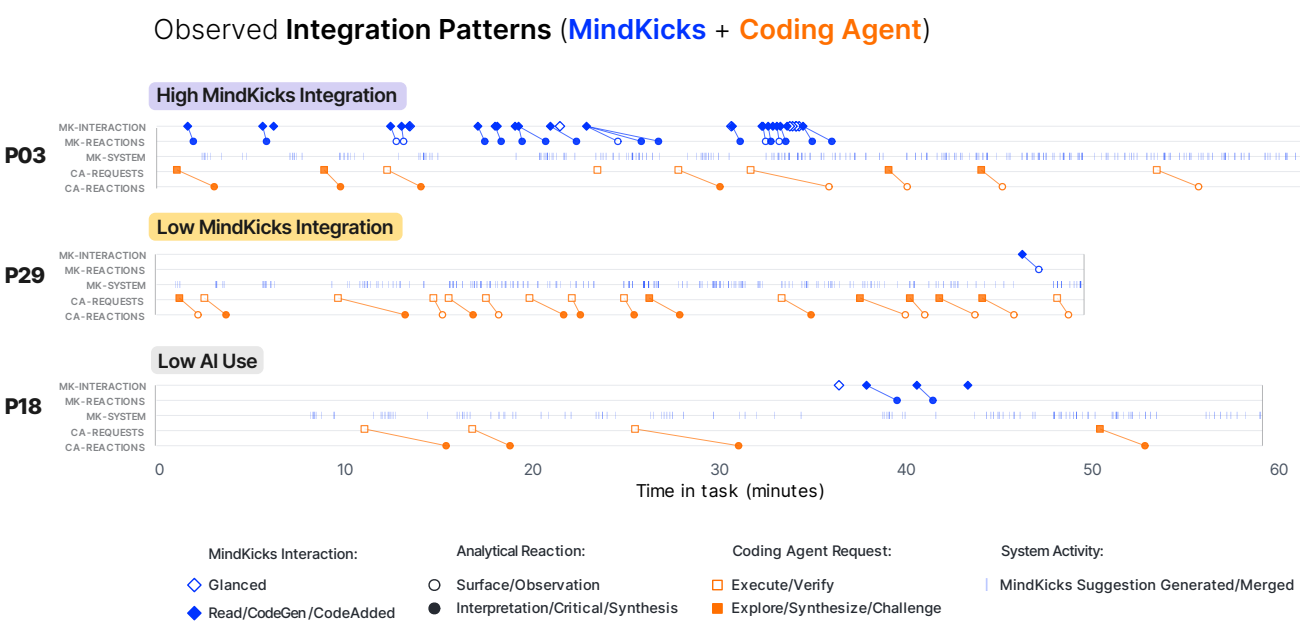


FIGURE 7.12: Representative MindKicks timeline examples illustrating the three usage patterns: P03 as a *High MindKicks Integration* participant, P29 as a *Low MindKicks Integration* participant, and P18 as a *Low AI Use* participant. Blue diamonds indicate MindKicks card interactions, blue circles indicate analytical reactions to MindKicks cards, light blue ticks indicate system-generated MindKicks cards/merges, orange squares indicate coding agent requests, and orange circles indicate reactions to coding agent responses. Outlined markers indicate lighter engagement events (e.g., glanced cards, surface/observation reactions, or Execute/Verify coding agent requests), while filled markers indicate deeper uptake or more analytically engaged events (e.g., Read/CodeGen/CodeAdded cards, Interpretation/-Critical/Synthesis reactions, or Explore/Synthesize/Challenge coding agent requests). See Appendix Figure E.2 for all participants' session timelines.

7.6.2.1 The majority of users actively integrated MindKicks into their workflow.

Looking at the participant-level interactions with MindKicks in more detail, several integration patterns emerged, with the majority of users highly integrating MindKicks into their workflow (Table 7.5 and Table 7.6). 6 of 10 participants (P03, P06, P10, P17, P20, and P27) showed a ***High MindKicks Integration* pattern**. All of these participants used MindKicks frequently as an integrated part of their workflow, reading suggestions and incorporating the code generated by the suggestion cards into their notebooks. Throughout the session, these users interacted with a relatively large number of surfaced suggestions (ranging from 10 to 27 unique cards) and often used the cards'

code-generation features (ranging from 5 to 7 added code cells). For example, P03 repeatedly moved between MindKicks and the coding agent throughout the session and interacted with many surfaced cards in a deeper fashion: As apparent in the timeline plots (Figure 7.12 and Appendix Figure E.2), most MindKicks interactions were cards that were read, used for code generation, or added into the notebook (filled blue diamonds ⧫) rather than only glanced at (◊). These deeper card interactions were then also often followed by higher-order analytical reactions (shown as filled blue circles •), where P03 interpreted, critiqued, or synthesized the suggestion's relevance for the analysis.

7.6.2.2 While few analysts used MindKicks less, active users discovered the most data issues.

Besides the majority of active MindKicks users, a few analysts made less frequent use of MindKicks suggestions. P11 and P29 showed a ***Low MindKicks Integration* pattern**, where they interacted with only a few surfaced MindKicks cards while frequently utilizing the coding agent, overall relying more heavily on the coding agent for support, with MindKicks playing a comparatively peripheral role (see Figure 7.12).

Similarly, P18 and P21 exhibited a ***Low AI Use* pattern**, in which they briefly interacted with a few MindKicks cards and made only a few coding agent requests throughout the session. For example, as apparent in the timeline, P18 used both AI features sparingly, with only a few coding agent requests and a small number of MindKicks card interactions (see Figure 7.12).

However, unlike these less active MindKicks users, participants who actively integrated MindKicks into their workflow identified the most data issues. Within the MindKicks condition, the MK Interaction Score (the volume-and-depth-weighted card-engagement measure introduced in Section 7.5.8) correlated strongly with the total number of discovered issues ($r = +.75$, $p = .012$, $n = 10$), see Figure 7.13.

7.6.2.3 Examining the interplay of MindKicks and the Coding Agent.

Many participants used the coding agent in parallel with MindKicks, creating interleaved sequences of suggestion uptake and coding agent requests. For example, as apparent in the timeline of P03's session (see Figure 7.12), several clusters of deeper MindKicks interactions and reactions appear alongside coding agent requests (□/■), showing how P03 treated MindKicks as an additional stream of analytical directions in combination with the coding agent.

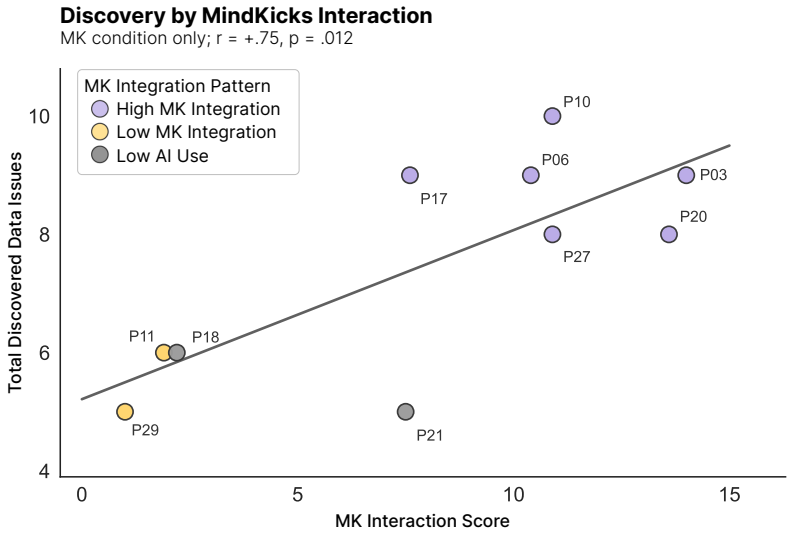


FIGURE 7.13: MindKicks Interaction Score versus discovered data issues (discovery total) within the MindKicks condition. Points represent individual participants; the trend line shows the positive within-condition association between higher MindKicks integration and the number of discovered data issues.

Participants valued MindKicks and the coding for complementary purposes. In many cases, MindKicks and the coding agent played complementary but overlapping roles: MindKicks was used to explore or validate analytical directions, whereas the chat agent supported code generation, articulation, and clarification. For example, P03 often read and pinned MindKicks cards while the coding agent generated responses, later pasting card suggestions into the agent for explanation; P17 used a MindKicks-generated analysis code and then copied its output to the agent for verification; and P20 pursued a MindKicks suggestion before asking the agent to execute follow-up analysis.

Reflecting on their process, P21 described the different AI usages as *"both very useful as MindKicks gave some really good suggestions [for] the data analysis"* while *"I [used] the chat agent to better frame my answers in the memo"* (P21). P11 similarly described how the chat agent often would help with clarifying MindKick suggestions: *"I just [used the chat agent to] read back the suggestions from [MindKicks] and it helped clarify them"* (P11).

Participants rapidly switched between Coding Agent and MindKicks to keep work moving. Often, participants used the Coding Agent and MindKicks in parallel, rapidly multitasking between their own analysis, the chat agent, and MindKicks to avoid waiting or to cross-check outputs. For example, P03 frequently inspected and

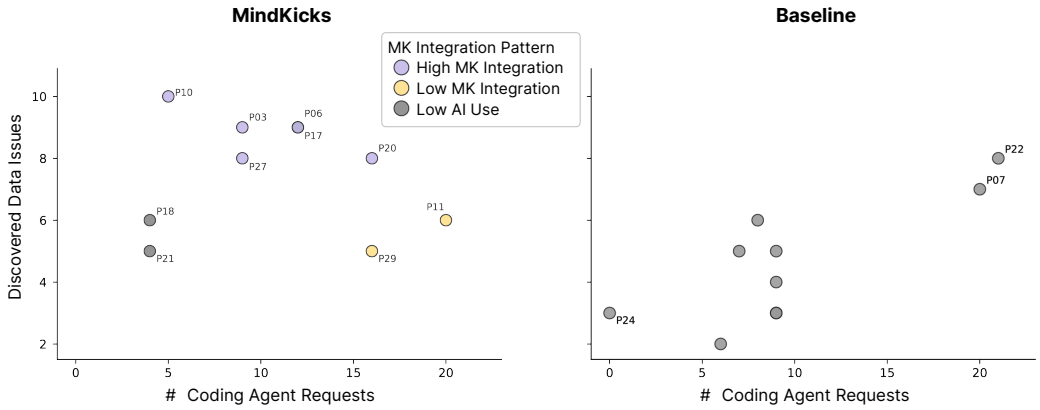


FIGURE 7.14: Number of discovered data issues (discovery total) by coding agent request volume, separated by condition. Point color in the MindKicks panel indicates each participant's MindKicks integration pattern. In the baseline, the highest discovery rate occurs among participants who sent more coding agent requests than High MK Integration users, yet they were at or below the High MK users' discovery level. This suggests that MindKicks helped, especially for these High MK users, to broaden their analytical investigation and identify more data issues by combining MindKicks with the coding agent.

pinned relevant cards while waiting for coding agent output, then used those card ideas as follow-up material for additional analysis or exploration. P03 described this as a time-management strategy: *"I don't want to waste time on actually waiting for the chat agent answer so I just try to focus on like different tasks that I'm doing and then come back to it when it has the answer"* because otherwise *"I'm just waiting for the chat agent to respond [which is] just a waste of time"* (P03).

Participants wanted more shared context across MindKicks and the coding agent. Several participants found the cognitive sidekicks helpful but demanded tighter integration with the chat agent. P20 said that the main improvement would be *"a much richer context across the [two] AI tools"* because *"the chat agent and MindKicks are sort of disconnected. They do not have a shared context."* (P20). This disconnection produced extra coordination work: *"the [coding agent] produced results that sort of were disjointed. That caused sort of like a break in context flow in my mind"* (P20). Other participants wanted the coding agent to help them make sense of MindKicks' suggestions. P21 imagined that *"if the chat agent could go through the [MindKick] suggestions"* and verify *"which ones are the good ones and how we could improve our data analysis based on suggestions given by MindKicks"* (P21).

7.6.2.4 MindKicks interaction and discovery outcomes in relation to Coding Agent usage.

Across both conditions (MindKicks and baseline), participants used the coding agent similarly in terms of the number and type of agent requests. For request volume, there was no significant difference between groups (overall $M = 10.2$, $SD = 5.8$; Baseline $M = 9.8$, $SD = 5.7$; MindKicks $M = 10.7$, $SD = 6.0$), and also the overall distribution of agent request types (what people asked the agent to do) was similar (see Appendix Figure E.1 and Appendix Table E.4 for all coding agent requests). However, a more nuanced analysis of the interplay between MindKicks and the Coding Agent revealed two interesting effects:

1) Actively combining MindKicks with the Coding Agent often led to more discovery, even with fewer Coding Agent requests. As shown in Figure 7.14, active MindKicks users achieved the highest discovery scores when pairing MindKicks with the Coding Agent, despite issuing only low-to-moderate numbers of Coding Agent requests. In contrast, in the baseline condition, the highest discovery rates were observed among participants who sent more Coding Agent requests than active MindKicks users, yet they discovered fewer issues. Taken together, this pattern suggests that MindKicks—particularly for active users—helped broaden and diversify analysts' investigations, thereby making the use of the Coding Agent more productive.

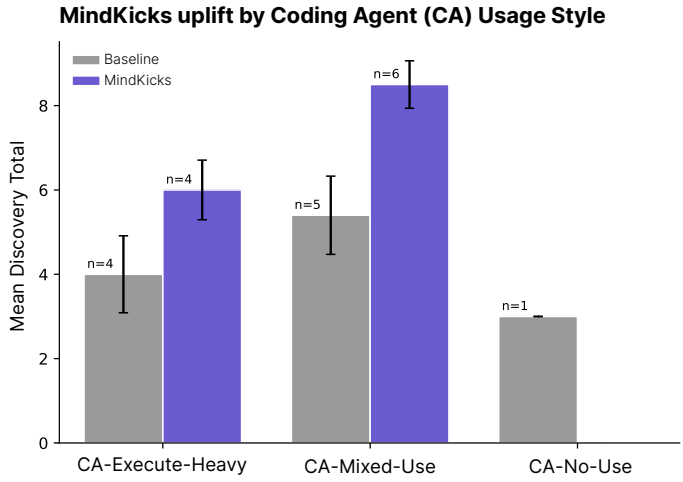


FIGURE 7.15: Mean discovery total by Coding Agent (CA) Usage Style and condition; error bars are ±1 SEM.

The largest uplift appeared among participants in the *CA-Mixed-Use* (+3.1).

2) MindKicks boosted discovery more for participants who used the coding agent for both exploratory and execution purposes. Throughout the sessions, analysts used the coding agent (CA) in different ways. To capture these differences, I grouped participants by the share of execution-oriented CA requests (Execute + Verify + Challenge out of total CA requests; threshold $\sim$60%). This yielded three high-level CA usage styles: **CA-Mixed-Use** ($n = 11$; share $< 60\%$), **CA-Execute-Heavy** ($n = 8$; share $\geq 60\%$), and **CA-No-Use** ($n = 1$; zero CA requests, $P24$). Across the first two groups, MindKicks was associated with higher discovery (Figure 7.15). However, the largest uplift appeared among participants in the *CA-Mixed-Use* ($+3.1$). A plausible interpretation is that *CA-Execute-Heavy* participants treated the coding agent primarily as an implementation aid, using it to translate an already existing analysis plan into runnable code, whereas *CA-Mixed-Use* participants also used the coding agent to surface alternatives (e.g., checks, comparisons, or reframings). In this framing, MindKicks slightly helped execution-heavy users discover more data issues, but it was most effective when paired with a coding agent workflow that already included exploratory, divergence-promoting prompts.

7.6.2.5 How people worked with and experienced proactive suggestions.

Participants appreciated MindKicks' inline features. Many participants emphasized that MindKicks' advantage was not only that it offered suggestions, but that its suggestions were spatially anchored in the evolving notebook, as P10 described: *"having [MindKicks] inside the main coding section where it highlights [...] code parts [...] was really helpful. As soon as I saw [the MindKicks code highlights] I felt like digging more into it"* (P10).

Participants needed time for making sense of and prioritize suggestions in the Inbox. While many valued MindKicks for surfacing useful directions without being asked, some noted they also had to spend extra effort deciding which suggestions were worth their attention. As a strategy, several participants developed lightweight filtering practices, often by first reviewing high-impact suggestions or skimming for the most relevant cards. However, managing the suggestions sometimes felt like a task in itself, as P17 described how they mainly checked the high-impact suggestions because they *"could not actually read through all of it"*, yet even this filtered view could still feel like *"a task to clear"* (P17). Other participants similarly treated the suggestions as material to triage rather than to exhaustively work through. P21 said they ignored some suggestions *"because there were a lot"* and because *"it was basically impossible in the timeframe*

of one hour to cater to all of them"; their strategy was therefore to *"skim through it and see which ones were the most useful"* and implement those (P21).

Participants wanted more control over the flow and timing of suggestions. While the majority of participants found the proactive suggestions useful, many also noted that their number and placement made it difficult to keep track of what applied where. As P21 reflected: *"MindKicks' support was overall pretty good but [...] it was a little bit hard to monitor because there were suggestions all over the place"* (P21). Similarly, P27 asked for more direct control over when MindKicks were active: the system should *"give me more control over how I would want to use it rather than just popping up suggestions"* and described the current experience as *"overwhelming me as a user with myriads of suggestions"* (P27).

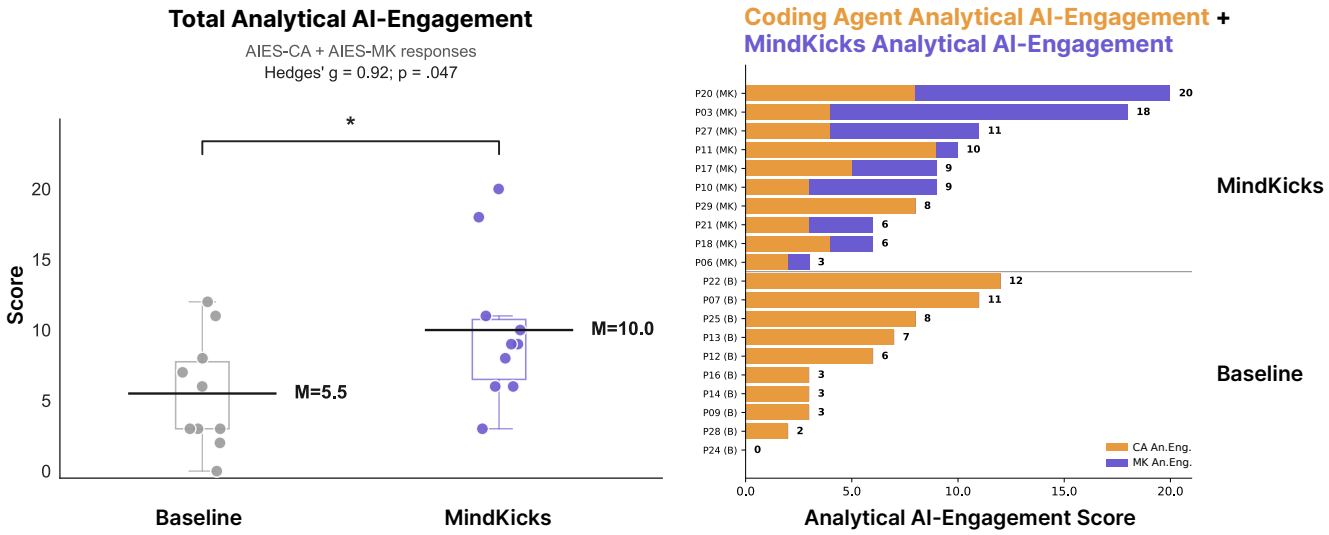


FIGURE 7.16: Analytical AI-Engagement Score by condition. (Left) AIES-TOT (CA + MK responses) per participant; black bars mark group means. (Right) Per-participant decomposition of AIES-TOT into MK-mediated (purple) and CA-mediated (orange) components, ordered within condition by total (Baseline participants appear as CA-only bars).

7.6.3 How MindKicks Impacts Analytical AI Engagement (RQ3)

MindKicks substantially increased participants' total analytical engagement with AI-generated outputs. As seen in Figure 7.16-left, MindKicks participants showed 82% higher combined analytical engagement across MindKicks and Coding Agent outputs than Baseline participants (Baseline $M = 5.5$ ($SD = 3.98$), MindKicks $M = 10.0$ ($SD = 5.29$),

Welch's $t = 2.15$, $p = .047$, Hedges' $g = 0.92$ (large)). While participants in both groups engaged analytically with Coding Agent responses at a similar level, MindKick users in parallel analytically engaged with the cognitive sidekicks, resulting in the overall increased effect (Figure 7.16–right). Taken together, these results indicate that MindKicks created additional, meaningful opportunities for analytical engagement, thereby helping expand participants' analytical reasoning. This interpretation is reinforced by a strong positive correlation between the combined analytical AI-engagement score and the number of discovered data issues across participants in both conditions ($r = +.59$, $p = .006$, $n = 20$). In other words, deeper analytical engagement with AI outputs (via the coding agent, MindKicks, or both) was generally associated with greater discovery in this task.

7.6.3.1 Surfaced tensions between productive cognitive demand, overload, and agency.

While overall MindKicks increased analytical AI engagement, which often led to more discovered data issues, this benefit also came with notable trade-offs and mixed feelings about participants' cognitive load and agency. For example, in the post-task survey, MindKicks users rated the *additional cognitive demand of managing the AI tools* slightly higher than Baseline participants ($MD = +1.8$, $p_{\mathrm{BH}} = .080$, see Figure 7.10). Some of this increase likely reflected interface overhead: participants had to manage many proactive cards under time pressure while also adjusting to a new interface. However, at the same time, the added demand may also have been productive: participants needed to read suggestions, assess their relevance, and decide how to test them—effort that may partly explain why they discovered more data issues. As P03 reflected on this tradeoff, *"it was a bit overwhelming but it did have some very good suggestions that I was able to like figure out and that I was not thinking before... so I feel it did help me"* (P03).

Another surfaced tension relates to users' felt agency: While MindKicks supported their analysis process, it sometimes made participants feel less in charge of their own thinking. For example, in the post-task comparative survey, MindKick users more strongly reported *greater deference to the AI tools over their own analytical instincts* than Baseline participants ($MD = +1.3$, $p_{\mathrm{BH}} = .133$, see Figure 7.10).

However, these reported attitudes are contrasted by other MindKicks-specific survey items where MindKick users tended to experience the suggestions as **autonomy-preserving prompts** (Figure 7.11), with the majority reporting that the *suggestions felt like prompts to think rather than directions to follow* ($M = 4.9$, $SD = 1.29$), and that *acting*

on a suggestion felt like their own analytical decision rather than following instructions ($M = 4.3$, $SD = 1.25$).

Reflecting on their process, P10 described that tension between productive attention draw and perceived limited agency: *"as soon as I saw some keywords that [caught my attention] I'm like 'yeah this is an interesting pattern' that I usually would have taken a longer time to come up with"* and summarized the progress they were able to make with MindKicks as *"very fast"* (P10). However, the same participant then noted that *"it didn't let me think more on my own"* because *"those were [all] really great suggestions"* (P10). Similarly, P18 described MindKicks' support as useful overall, but also noted: *"sometimes I feel like I relied more on the AI because I'm not quite used to dealing with this domain knowledge"* and *"I wanted its expert opinion instead of thinking a bit more and relying on my instinct"* (P18). P20 expressed a similar discomfort, saying that *"somewhere along the way I sort of lost my own analytical thinking"* and that *"I [sometimes] went with MindKicks a little more than I should have"* (P20).

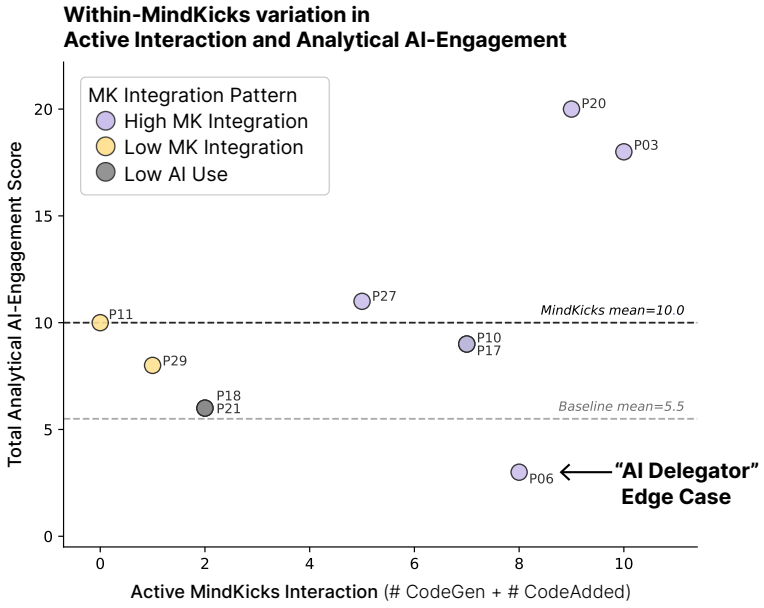


FIGURE 7.17: Active MindKicks card engagement versus Combined Analytical AI Engagement (AIES-TOT) within the MindKicks condition. Point color indicates the MK Integration Pattern groups from Table 7.5. Dashed horizontal lines indicate the MindKicks and Baseline mean AIES-TOT scores.

7.6.3.2 The "AI delegator" edge case.

For most participants, active MindKicks use also led to greater analytical engagement with AI outputs (see Figure 7.17). But the study also revealed an interesting "AI Delegator" edge case (P06), in which active MindKick use did not translate into deeper analytical engagement. This participant frequently used MindKicks card actions to rapidly generate and add code to their notebook with little reading and reflection on the cards' text or code. Instead, they instructed the coding agent to interpret their notebook with the MindKicks-generated code and to produce the memo text for them. P06 described their delegation process as: *"MindKicks gives me a lot of suggestions [so] that I don't need to work [with] all the guidance [from] the chatbot manually"* and *"I just need to add all the high-level suggestions [from MindKicks] to my notebook"* (P06). This use case illustrates how some users might use the system in a more delegation-oriented way, relying on it to generate outputs they then integrate into their workflow without deeply cognitively engaging with its suggestions. However, in terms of discovered data issues, this delegation-oriented use of MindKicks yielded only a mid-range discovery outcome, which supports the interpretation that the actual human analytical engagement with MindKicks' suggestions drives the discovery of deeper-level data issues in this task.

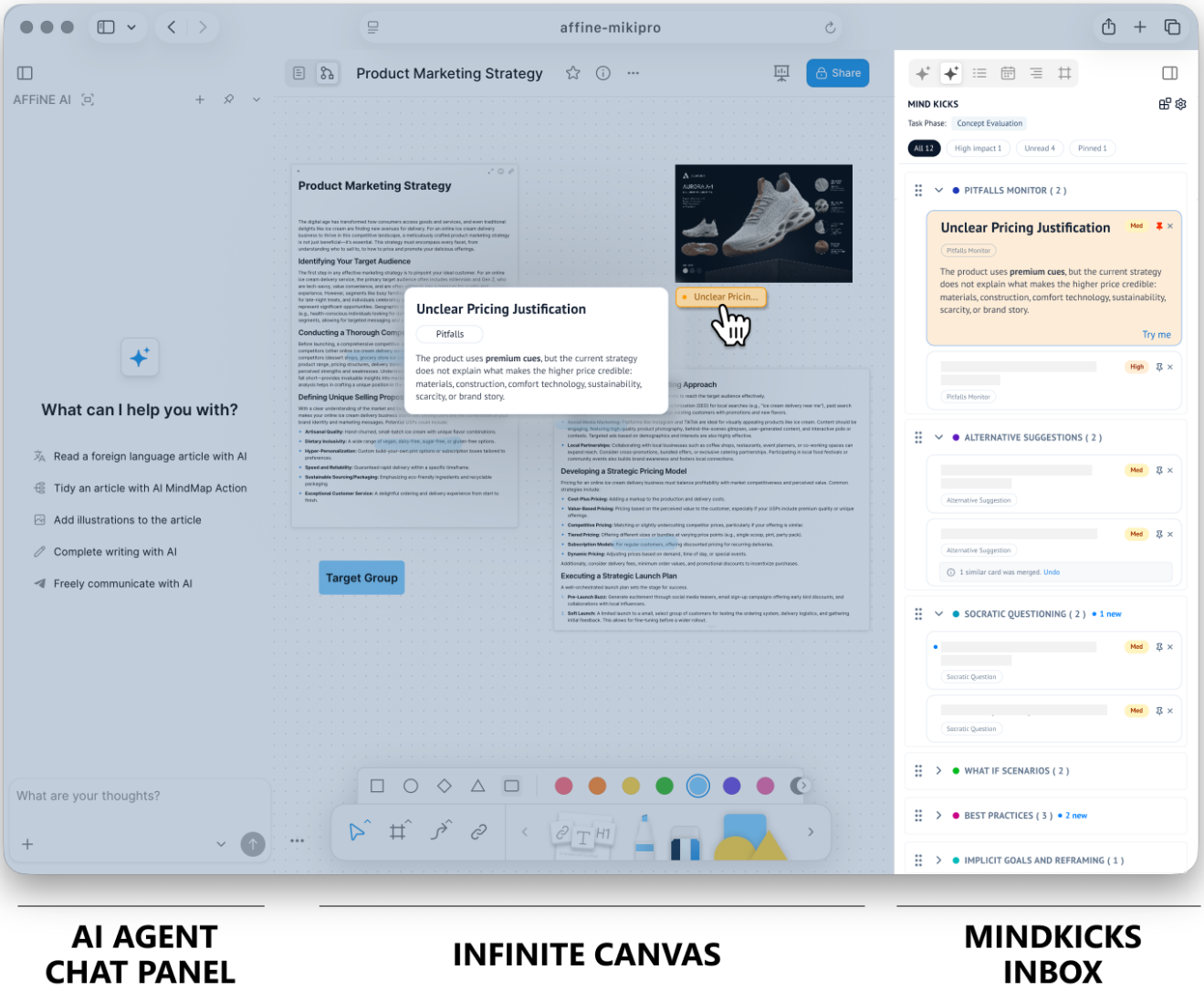


FIGURE 7.18: MiKi[Prod] — a MindKicks adapter for AI-assisted product planning on an infinite canvas.

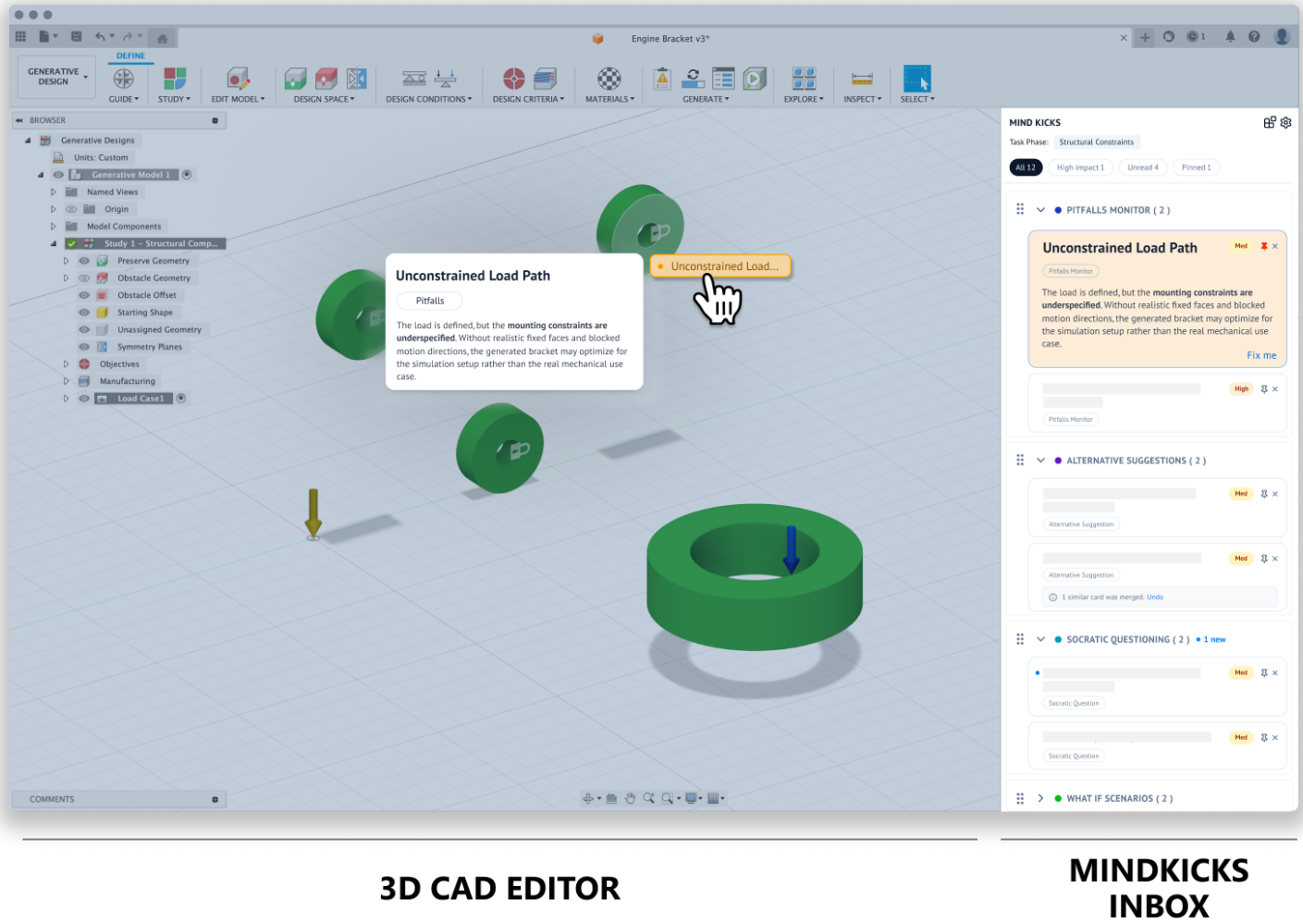


FIGURE 7.19: MiKi$^{\text{Mech}}$ — a MindKicks adapter for generative mechanical design in Fusion 360.

7.7 Outlook: Alternative MindKicks Adapters MiKi$^{\text{Prod}}$ and MiKi$^{\text{Mech}}$

While this project focuses on the implementation and study of MindKicks in the context of data analysis with MiKi$^{\text{Data}}$, the MindKicks architecture was intentionally designed as a modular plug-in framework (Section 7.4). This architecture allows the same six cognitive support strategies to be employed in other domains and host applications through dedicated domain context and tool adapters. To demonstrate this portability, I

have begun implementing two additional MindKicks adapters. MiKi$^{\texttt{Prod}}$ supports strategic product-planning workflows (Figure 7.18). It is a MindKicks plug-in for AFFiNE, a Miro-like 2D canvas environment with an integrated AI chat agent, where suggestions can be anchored to artifacts in an infinite canvas. The second example is MiKi$^{\texttt{Mech}}$ (Figure 7.19), a MindKicks plug-in for Fusion 360 that supports mechanical designers during generative-design workflows, where cards and spatial highlights can surface constraints, pitfalls, and alternative modeling decisions in relation to the evolving 3D artifact.

Designing and implementing these adapters provided a concrete preliminary test of the plug-in architecture's generalizability. MindKicks' universal layer provided the cognitive support move, and the domain adapter then adds the surrounding context while UI outlets define how interventions are surfaced. Therefore, a *Pitfalls Monitor*, for instance, must check data validity in MiKi$^{\texttt{Data}}$, pressure-test product assumptions in MiKi$^{\texttt{Prod}}$, and point out possible constraints and simulation setup issues in MiKi$^{\texttt{Mech}}$. In MiKi$^{\texttt{Data}}$, UI outlets map onto the notebook via cards, cell-level chips, and inline code highlights. However, in MiKi$^{\texttt{Prod}}$, support has to latch onto artifacts and clusters on an infinite canvas, where work is spatially distributed. Lastly, in MiKi$^{\texttt{Mech}}$, side-panel cards must be coordinated directly with on-canvas highlights in the 3D model, since key assumptions often reside in the geometry and simulation configuration. Overall, these early prototypes suggest that MindKicks' portability is feasible and that the cognitive sidekicks concept's versatility enables cognitive support for users across various domains and tools. However, future studies are needed to examine the extent to which the same support strategies and adaptive UI mechanisms improve cognitive engagement, discovery, and outcome quality in these domains.

7.8 Discussion

In the following sections, I discuss the study's findings, reflecting on *MindKicks as a support for cognitive engagement and discovery* and its *implications for expanding chat-based LLM interactions*. Further, I discuss resulting *design considerations for cognitive support systems*, including trade-offs around proactivity and adaptive UI.

7.8.1 Supporting Cognitive Engagement and Discovery with MindKicks

Overall, the study's findings show that cognitive sidekicks can deepen existing AI-assisted data analysis workflows. MindKicks augmented the chat-based coding agent workflow with an additional stream of proactive suggestions that helped participants reconsider assumptions and pursue new analytical directions in parallel. As a result, participants in the MindKicks condition were able to discover substantially more underlying dataset issues than Baseline participants, even though both conditions had access to the same coding agent (Section 7.6.1). Especially the increase in deeper-level findings (Tier 2) suggests that MindKicks helped participants progress from easier-to-spot surface-level findings toward more hidden issues that required deeper analytical probing. The process analysis furthermore revealed that MindKicks participants showed higher combined analytical engagement across MindKicks and coding agent outputs than Baseline participants, while their analytical engagement with the coding agent itself remained comparable (Section 7.6.3). Overall, MindKicks fostered additional moments in which participants interpreted, questioned, or synthesized AI-surfaced analytical directions. MindKicks, therefore, contributes to the broader HCI question of how GenAI systems can augment open-ended complex exploratory workflows while supporting users' metacognition and keeping them actively engaged in sensemaking and verification [47, 96, 204, 315, 349]. Specifically, the study demonstrates that presenting *different* cognitive support strategies to users in parallel via workflow-embedded adaptive UI elements can be an effective way to provide proactive cognitive support and thus improve human-AI interaction.

However, the study has also shown that this proactive support can sometimes be experienced as an added cognitive burden or reduced felt agency (Section 7.6.3). This pattern does not necessarily mean that MindKicks made the task worse; rather, it highlights a demand for a more fine-grained account of *what kind of effort* the system elicits. Because while some of the added load may have been *extraneous* interface-management work (e.g., triaging an overly dense stream of cards), other effort may have been *germane* analytical work [345, 346] as a result of reading suggestions and judging their relevance. Such a distinction is important because the study intentionally used a relatively high suggestion frequency ($M = 109$ suggestions per session; Table 7.5), and participants had only one hour to learn a new system while completing the task. Going forward, inspired by research in cognitive and learning sciences [271, 370] and recent calls in HCI [189], future work should therefore develop more nuanced measures that

better separate overload from productive cognitive engagement—and thereby assess whether users become less overwhelmed as proactive cognitive support becomes part of routine AI-assisted analytical practice.

7.8.2 Implications for Expanding Chat-Based LLM Interactions

A core motivation of this work has been to explore how existing AI interfaces, particularly for chat-based LLM coding agents, might be expanded with parallel, proactive, and adaptive UI mechanisms that complement the predominant chat paradigm. Current LLMs for programming and data analysis are increasingly powerful, but their chat interfaces remain limited to reactive and sequential interactions, requiring users to know what to ask, wait for responses, and then evaluate the generated outcomes. Prior studies show that such workflows impose substantial cognitive effort on query formulation, context specification, and output evaluation, and that engagement can decline when agentic systems offer limited affordances for reflection [58, 97, 137, 251]. Within this framing, our findings suggest that the concept of UI-embedded cognitive sidekicks can add a complementary layer to chat-based exploratory coding agent workflows, which works by providing an additional proactive stream of context-aware, thought-provoking suggestions that can surface in parallel to the existing workflow through different adaptive UI outlets outside of the chat. As seen in the study, participants who highly integrated MindKicks into their workflow often interleaved MindKicks and chat agent interactions by inspecting card suggestions while the coding agent generated code, then turning card ideas into follow-up analyses or verification requests (Section 7.6.2). Because participants could also directly generate data-analysis code from MindKicks suggestions, active and high-performing users often also used this path while waiting for the chat agent to finish code generation in parallel. As a promising starting point, informed by how participants utilized MindKicks and participant reflections, this study has opened up a rich space of future design opportunities around tighter integration of proactive cognitive sidekick support with chat-based coding agent assistance (Section 7.6.2). For example, future systems could let users drag a suggestion card into chat, invoke a card-specific conversation directly from the card, or ask the coding agent to compare pinned suggestions. Similarly, a richer shared context, ways for the chat agent to skim, highlight, or curate relevant cards, and smoother transitions from a card into a follow-up conversation. Future work should also further explore how the concept of proactivity could be further expanded to encompass autonomous cognitive agents that can not only provide suggestions but also conduct

research and execute actions in the background regarding the suggested topics on behalf of the user while presenting the summarized results to the user.

7.8.3 Design Considerations for Cognitive Sidekicks and Adaptive UI

Reflecting on the study findings, several design considerations for proactive cognitive support systems and adaptive UI mechanisms have also surfaced beyond this particular data analysis implementation. Generally, the underlying idea behind the MindKicks architecture is to separate reusable cognitive support strategies from domain-specific context and the host application's UI, enabling six different cognitive support sidekicks to run in parallel while grounded in the evolving data analysis notebook workflow (Section 7.3). A core motivation of this project has been to better understand the extent to which such a parallel sidekick approach is actually beneficial for providing cognitive support to users. Overall, the study suggests that this multi-support approach constitutes a practical design pattern, but also that its effectiveness highly relies on the careful design of adaptive UI mechanisms to help users manage surfaced support.

For example, the findings suggest that MindKicks' **side-panel inbox and on-canvas cues were an effective way for balancing peripheral support with situated noticeability**. Prior work shows that surfacing support peripherally can reduce interruption costs but may be ignored when too peripheral [220, 236, 272, 288, 296]. In this context, the combination of a side-panel inbox with inline code highlights and notebook cell-level anchors seems to provide a usable design pattern that makes AI suggestions easier to interpret when tied to specific context on the work canvas (Section 7.6.2).

Furthermore, **the inbox metaphor was useful but incomplete.** The side-panel inbox offered a persistent place to skim and read, as well as to pin and filter suggestions. While effective, participants often worried about losing items that could become relevant later (Section 7.6.2). In this context, "deleting" suggestion from the inbox may be too strong a metaphor, whereas snoozing or archiving cards could preserve recoverability while reducing immediate visual clutter. Additionally, the interaction could be further improved by giving users more control over the organization of suggestions, for example, allowing users to cluster suggestions by analysis topic, variable, or notebook cell.

Similarly, the **impact-ranking mechanism** helped participants triage suggestions. Many users described starting with high-impact cards or skimming headlines, suggesting that the ranking mechanism and priority cues made proactive support easier to handle (Section 7.6.2). However, when too many cards appeared, the stream could still feel like another task to manage. Future systems should further explore mechanisms that allow pairing priority cues with more dedicated user controls. For instance, keeping only high-impact cards on the canvas while directly moving lower-priority suggestions to a searchable inbox. Another strategy to further explore would be to surface suggestions based on users' cognitive states. As suggested by recent calls for cognitively aligned AI systems [7], fewer cards could appear during focused coding activities, while more appear during idle time or during review sessions.

Going forward, these design considerations offer a promising starting point for creating effective pluggable cognitive sidekicks. Future research should further examine how this approach, combined with improved adaptive UI techniques, works in other areas, across different application workflows, and in real-world use over longer periods.

7.8.4 Limitations of the Study

I highlight three main limitations of the study. First, the sample was small (N=20; 10 per condition). While this was sufficient to reveal clear behavioral differences, the quantitative results should be treated as preliminary evidence. In particular, the correlation and subgroup analyses, such as MindKicks usage, coding agent styles, and analytical AI engagement, should be interpreted as exploratory and would benefit from replication with larger, more diverse samples. Second, the study task was restricted to one hour. While this made the sessions comparable and required participants to make quick decisions during the analysis, real data work usually happens over days or weeks, which allows for more opportunities to reflect and revisit assumptions. Longer-term studies should therefore examine how MindKicks works in such settings and whether the cognitive load associated with using the interface decreases as users develop routines for working with proactive suggestions. Third, the task focused on a single domain (assessing data fitness for a property dataset in a housing-price prediction scenario). Participants likely had different levels of familiarity with that domain, which in turn may have influenced what they noticed and how quickly they understood the data. Although I did not observe a clear pattern linking domain expertise to performance, future research should test MindKicks in other domains and explicitly measure or account for domain familiarity.

Chapter 8

Conclusion and Future Directions

The north star of this dissertation has been the idea that computational systems should augment human thought and ability, ultimately empowering people to tackle more complex problems with AI. Across studies in mechanical design, presentation authoring, and data analysis, this dissertation demonstrates that cognitive support can improve how people formulate intentions, explore problems, and evaluate AI-generated outcomes. It further shows how such support can be translated into novel interfaces and interactive systems that move beyond conventional chat-based interaction. Together, these findings open opportunities for AI systems that augment human thinking throughout complex workflows, supported by new prototyping methods for designing and evaluating context-sensitive interactive AI behavior.

When I began my PhD in 2021, much of today's generative AI landscape still felt distant: Powerful general-purpose models, such as GPT or Claude, were not yet part of everyday writing, coding, design, and knowledge work. However, over the course of this dissertation, that changed rapidly. Systems that once seemed years or decades away became widely available, with capabilities repeatedly exceeding what I and many others had imagined possible at the start of this work. But this progress in itself did not eliminate the core challenges of human-AI interaction at the center of this dissertation. On the contrary, in many ways, it made these issues more urgent [202, 289]. As AI systems became better at producing text, slides, code, and mechanical parts, the limiting bottleneck became less about whether something *could be* generated with AI and more about whether people could think through *what should* be generated, *why* it matters, *how* it should be evaluated, and what *tradeoffs* should guide its refinement. Anyone who has had to distill a complex topic into a ten-minute presentation knows that making the slides is rarely the hardest part. The hard part is deciding what the argument is, what to leave out, how to structure the flow, and how to revise the material until it

serves a specific audience and purpose. Once that thinking is clear, recreating the artifact is comparatively easy. As emphasized throughout this dissertation, this distinction matters because many of today's AI interfaces make production feel effortless while leaving the deeper cognitive work of framing, specification, reflection, and judgment unsupported.

However, not only did the technological landscape change throughout this dissertation, but the research focus changed as well. Initially, I began my research by asking: *What challenges do designers face when learning to co-create with computational AI tools? What are effective strategies to support designers in learning to co-create with computational AI tools?* In my first study, I identified opportunities to support better human-AI co-creation by scaffolding users' learning about AI capabilities and limitations, prompting planning and reflection, improving AI systems' contextual awareness of users' tasks and objectives, and enabling richer multimodal communication between people and AI systems. In a way, these early findings became the dissertation's agenda: a map for exploring how interactive systems might help people remain cognitively engaged and think better while working with increasingly capable AI tools. However, with each step I took, new open questions emerged that partially reframed my initial assumptions. Of course, this is what research is all about. Consequently, the work moved from observing designers learning to co-create with AI, to probing voice-based metacognitive support agents, to building methods for prototyping multimodal real-time agents, to designing graphical micro-prompting interfaces for rich content creation, and finally to integrating proactive cognitive sidekicks into existing data analysts' AI-assisted workflows. Below, I highlight and reflect on the three most central takeaways from this process.

For complex creative work, AI interfaces need to provide stronger affordances for structuring, externalizing, and revising people's thinking. My formative studies in Chapter 3 made this central claim first visible in AI-assisted mechanical and material design. These tools promised to make difficult tasks easier by generating complex outcomes from user-specified goals and constraints. However, in the study, many participants struggled because the interface made the work seem like simply entering a few parameters and waiting for results. In reality, specifying criteria such as loads, clearances, material assumptions, manufacturing, and constraints required designers to carefully reason through the problem before the AI could be useful. When that thinking did not happen, polished outputs created an *illusion of success*: users accepted plausible-looking results without fully understanding whether the design satisfied the actual problem.

This realization became a major driver of the thesis. In Chapter 4, reflective questions, mental simulation prompts, planning, and sketching helped designers formulate intent, explore the problem space, and evaluate AI-generated outcomes more effectively. Later, in Chapter 6, *Intent Tagging* extended this lesson to slide authoring with LLMs that, instead of chat, use intent tags to make users' emerging goals visible and manipulable through graphical objects.

Across these studies, I discovered that the "holy grail" of AI interface design is not radical simplicity in the sense of removing thinking from the task. Instead, it is designing interfaces that make the *right* thinking easier. While the techniques explored in this dissertation are grounded in specific domains, they hint toward *broader design opportunities* for GenAI interfaces that help people externalize, inspect, and revise their thinking.

Cognitive in-the-moment support can only be effective when it fits gracefully into the user's workflow. This key learning emerged early on as I moved from identifying reflective prompts as a support strategy in Chapter 3 to testing voice-based metacognitive agents in Chapter 4. Those agents showed that proactive voice prompts could improve design outcomes, but they also exposed a fundamental *delivery* problem. Timely questions can be effective, but repeated questioning can amplify dependence on the agent or disrupt the user's flow. Voice made support easy to receive while working, but it was also too transient: Users could not return to spoken prompts later or decide when to engage with them.

Ultimately, these recurring tensions shifted my focus from purely asking *whether cognitive support is useful* to also asking *how it should be delivered.* Consequently, Chapter 7 further explored this tradeoff through *MindKicks*, which embedded different cognitive proactive "sidekicks" into an existing AI-assisted data analysis workflow via a combination of adaptive UI mechanisms. Studying the system revealed that providing workflow-embedded cognitive support to users through carefully designed adaptive UI can increase analytical AI engagement and help analysts discover more hidden data issues by complementing chat-based coding agents.

Going forward, as AI-based workflows increasingly become part of many people's and organizations' daily work, this dissertation lays an important foundation for future systems and further research on the interaction design of proactive cognitive support, especially how to make such support effective at augmenting human thinking without becoming another source of interruption, overload, or felt loss of agency.

Building effective cognitive AI support requires prototyping methods that allow evaluating AI models within the context of users' unfolding activity in open-ended, non-linear tasks. Another unexpected key insight emerged after I completed the Wizard-of-Oz study in Chapter 4 and wanted to turn the tested support strategies into a functional LLM-based real-time support agent. At first, this seemed straightforward since foundation models (such as GPT-5 at the time) could technically perceive screenshots, user verbalizations, and system logs, while generating fluent questions or suggestions in response. In principle, getting an LLM-based agent to provide *some* support was easy. However, getting it to provide the *right* support at the right moment was much harder. Soon, the central challenge shifted from identifying the best LLM prompt to creating an environment that allowed me to learn how a general-purpose model behaves when placed into a novel, situated support task. This realization led to Chapter 5, where I developed *Counterfactual Replays* and the *Hybrid Wizard-of-Oz* approach for prototyping multimodal GenAI real-time support agents. These methods emerged because novel GenAI applications, such as the cognitive support systems developed in this dissertation, often lack benchmarks, test data sets, or even agreed-upon criteria for what constitutes good model performance. In a way, just as human-centered design has developed methods for engaging with the variability of human behavior and attitudes [227], designing LLM-based applications in a human-centered way requires us to develop new methods for engaging with the variability of model behavior in the context of a user's unfolding activity in open-ended, non-linear tasks. Building SocraBot, therefore, became a mediation process between two moving targets: The user's changing design activity and the model's probabilistic attempts to make sense of it. To evaluate model behavior, I needed to replay real sessions, trigger model responses at specific moments, compare prompt strategies in context, and steer live interactions while learning how the model behaved across situations. From this learning, the broader implications for HCI are to contribute better human-centered prototyping and evaluation methods to AI application development pipelines, which eventually will allow researchers and practitioners to build and study more effective real-time cognitive support systems.

8.1 Future Directions

I would like to highlight two promising broader research directions that this dissertation points toward. Both of these directions follow from the same underlying shift: as GenAI systems become more capable at producing artifacts, a central HCI challenge will become designing workflows that help people specify, explore, evaluate, and learn with these systems in ways that expand their thinking and capabilities.

8.1.1 How Can We Reimagine *Software Tools* as Cognitively Augmented Workflows?

While AI model capabilities have improved rapidly, user interfaces have remained greatly unchanged. Since the breakthrough of LLMs, natural language has become the default AI interaction modality, because it is flexible and easy to implement. But language in chat windows alone is also limiting for many complex tasks. It is serial, often open-ended in the level of abstraction, and frequently separated from the artifact being created. For example, these limitations are evident in current AI coding workflows, where chat-based assistants (such as GitHub Copilot or Claude Code) often make a single conversational thread feel like the primary medium for AI-assisted work [208, 251]. While these tools are extremely powerful, chat windows collapse complex work into the same scrolling exchange. As highlighted throughout this dissertation, as models become more capable, they will also require advanced interaction interfaces beyond chat that best match and support people's cognition, because different cognitive tasks demand different channels, tempos, and spatial arrangements.

Therefore, the contributions of this dissertation point to a broader future research agenda focused on designing *cognitively augmented workflows* and reimagining software tools as AI-supported productivity environments in which cognitive support is embedded into the interaction itself. Intent Tagging offers an early example of this direction. In contrast to chat, which expects users to express an entire slide-generation request in a single prompt, intent tags supported users' cognition by decomposing the task and gradually making emerging goals visible and manipulable through graphical, dynamic UI objects. Similarly, MindKicks explored a related idea in data analysis by complementing a chat coding agent workflow with proactive cognitive support cues attached to code snippets and notebook cells that analysts could inspect while continuing their main workflow. In both cases, the goal was to deliberately design AI interfaces for cognitive support around the user's attention and workflow.

While these systems are early explorations, future work should ask: What should an AI-augmented IDE, CAD tool, notebook, writing environment, or operating system look like if it is specifically designed to support human cognition? What kinds of manipulable representations can best elicit and support users' thinking process as work unfolds? How might AI systems turn exploration itself into a useful process trace, so that trying possibilities, rejecting suggestions, and revising assumptions become part of the working material of a new kind of AI-supported thinking practice? However, these directions also raise new open questions around evaluation. Because if the goal is cognitive augmentation, evaluation needs to encompass more than simply lower task load or faster completion. For example, if we, by default, treat a lower workload as the main goal, we might risk designing systems that remove the cognitive effort that actually often matters for complex work. But for true cognitive augmentation systems, we might actually need to understand whether, for example, the workflow helped users explore a problem more broadly or specify intent more carefully. For such more nuanced assessments, constructs from learning science could offer valuable starting points, such as the differentiation between *extraneous* (unproductive) and *germane* (productive) cognitive load [346]. Ultimately, reimagining software tools as cognitively augmented workflows requires evaluating how their interfaces shape users' thinking as work unfolds.

8.1.2 How Can We Reimagine *Learning* With and Through AI-Augmented Work?

A second direction for future work is learning. If AI gives rise to cognitively augmented forms of work, people will also need new kinds of literacy to effectively thrive. Earlier in this dissertation, I showed that designers' difficulties with AI-based CAD tools did not stem primarily from poor software usability. Instead, they involved learning how to formulate appropriate goals, interpret generated results, and recognize cases in which the tool appeared to succeed even though the underlying problem had been poorly specified. Related concerns are now emerging in AI-supported programming, writing, design, and analysis. Users may receive fluent, seemingly useful assistance without developing the metacognitive awareness required to evaluate whether the system actually supports sound reasoning [109, 289, 349]. From this perspective, AI literacy involves more than understanding what a model is or learning to write better prompts. It also requires people to assess their own understanding, uncertainty, search strategies, and reliance on AI.

However, this raises also a broader question for interface design. Well-designed systems are often assumed to be easy to learn and undemanding. For everyday tools and routine tasks, that is usually a sensible goal. But systems that support complex, open-ended tasks, such as design, analysis, or creative work, may differ. Some ways of interacting with them may take time to learn before their value becomes clear. Therefore, future research should investigate what kinds of interactions and interfaces are worth an increased learning effort because they will substantially expand what people can think and make. While SocraBot, Intent Tagging, and MindKicks provide early examples of this possibility, more capable forms of cognitive augmentation may call for unfamiliar interaction conventions, for example, ways to indicate uncertainty or make hidden assumptions visible.

An important open question is therefore: *What does metacognitive AI literacy look like in practice, and how can interfaces help people develop it while they work?* Besides generating high-quality artifacts, future systems might also help users notice when their provided requirements are underspecified or when they have explored only one framing. Furthermore, such systems could also support a more reflective practice, for example, by making people's thinking strategies visible over time (what they asked, what they ignored, and how their understanding changed). Eventually, this could integrate a layer of learning-oriented augmentation in AI-assisted work.

But this direction also matters from a socio-technical perspective, as GenAI may otherwise reproduce a new version of the "power user" divide. Because people with strong metacognitive skills tend to use AI systems to accelerate their work, while others may struggle to benefit from powerful models [289]. Going further, future research should therefore study how to prevent this divide by designing AI systems that scaffold productive struggle, reflection, and skill growth. Ultimately, the long-term goal is not only to teach people how to use AI tools as they currently exist, but to better understand what people need to learn as computers become more active cognitive partners, and to design interfaces that help users become more capable thinkers through their interactions with these advanced tools.

8.2 Conclusion

This dissertation examined how interaction and system design can better support people's critical thinking and reflection in complex GenAI-assisted work. Across studies in mechanical design, slide presentation authoring, and data analysis, I found that professionals often struggle with misalignment between generated outputs and their intentions, uncertainty about how to guide AI systems, and reduced cognitive engagement when too much of the task is delegated to automation. To address these challenges, I identified support strategies such as reflective questioning and thought externalization, explored voice-based metacognitive support agents, and developed new methods for prototyping multimodal real-time agent behaviors. I also introduced *Intent Tagging*, a graphical interaction technique for more granular and reflective control of large language models, and *MindKicks*, a modular architecture for integrating proactive cognitive support into existing AI-assisted workflows. Together, these studies support the central thesis that more effective and thoughtful use of interactive AI can be achieved by deliberately supporting people's critical thinking, metacognition, and reflection-in-action through novel interfaces and interactions. In sum, this dissertation contributes interaction techniques, systems, prototyping methods, and empirical insights that advance human-AI interaction toward helping people think more deeply, create more thoughtfully, and solve more complex problems with AI.

Appendix A

Additional Materials – Chapter 3

A.1 Task Outcomes

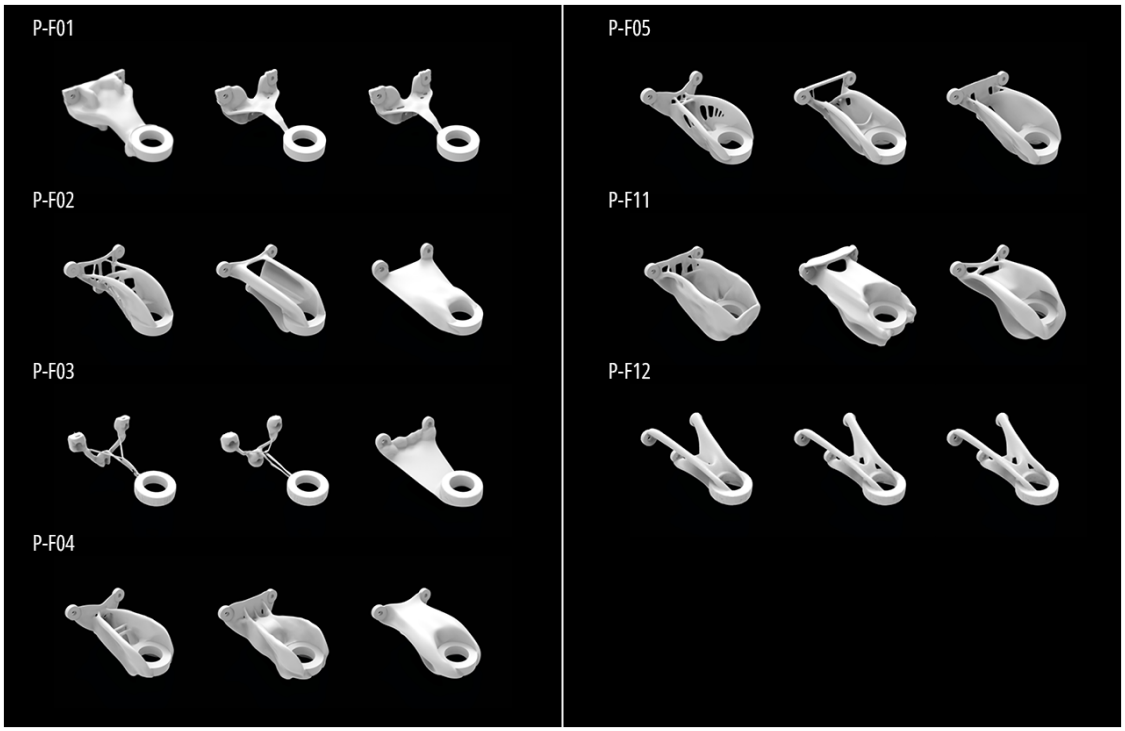


Figure A.1: Overview of all engine mounting brackets created in the Fusion 360 task. Every participant submitted three design options. Since the AI system generated the three options based on the same parameter values, each trio either met or missed the same criteria.

A.2 Interview Protocol

Table A.1: Interview protocol with questions of the semi-structured post-task interview.

Theme	Topic		Question
Collaboration with the tool	*General Experience*	Q3	Could you tell me what it was like to work with the design tool in general?
	General Expectations	Q4	Before starting this project, what role did you expect the design tool to take? And what role did it actually take during the design process? Did the design tool match your expectations?
	Roles – Coordination	Q5	Do you feel that there was a collaborative interaction between you and the design tool? Can you give us some specific examples?
		Q6	Using your own words, what metaphors or analogies would you use to characterize the tool and the collaboration with it?
		Q7	Would you say that the process of working with the design tool is co-creative? Why?
		Q8	How was the workflow with the design tool different from your workflow of working with a human teammate?
	Roles – Expectations	Q9	What role do you want the AI to take during your process?
		Q10	How would you describe an ideal collaborative design tool for that task?
		Q11	What would a design tool need to do so that you would perceive it as “collaborative” and “co-creative”?
	Mental Model	Q12	Do you remember situations in which you found the behavior of the design tool confusing or irritating?
	Shared Mental Models	Q13	Do you think the AI feature successfully understands your design intent and design practice?
		Q14	Do you think the design tool collected enough information to support you most effectively? Why?
	Shared Mental Model – Expectations	Q15	Are there any aspects from your design process where you hope a design tool like this could learn more from you to better support you?
	Communication	Q16	How would you characterize the dialogue between you and the design tool?
		Q17	Can you describe situations in which the communication worked well and not so well?
		Q18	Do you find the interface helpful in guiding you through the whole process while getting started?
	Conflict Resolution	Q19	Do you remember situations of conflict with the design tool? How did you solve the conflict?
	Timing / Flow	Q20	How did the simulation time influence your design process?
Design Process	*Satisfaction with design outcome*	Q21	Are you satisfied with the final design in general? How closely does it match the design brief?
	Surprise	Q22	Do you remember situations in which suggestions from the design tool surprised you? If so, could you give me an example of that? Do you find the surprises productive or not?
	Design Negotiation	Q23	How did you find that the design tool changed the way you design? Did you ever adjust your design goal/strategy according to the tool’s capabilities?
	Exploration	Q24	Did you explore more design options because of the design tool?
	Productivity	Q25	Do you think you can achieve the same output level (quality and quantity) without the help of an AI feature given the time constraints? Why?
	Trust / reassurance	Q26	How much do you trust the results from the design tool?
		Q27	Is there any other sort of analysis you were hoping the software would do that it didn’t? Why?
Learning Process	*Learning Process*	Q28	How would you describe the learning process for using the tool and collaborating with it?
		Q29	Which situations of the design process did require the most guidance?
		Q30	How did you learn about the capabilities of the design tool?
		Q31	Which aspects of the design tool were challenging to understand or learn about?
	Learning Resources	Q32	Which learning resources did you find most useful? Why?
	Learning Expectations	Q33	How could the tool better guide an experienced designer who is starting to use the tool?
		Q34	What would the ideal collaborative learning process look like, in your opinion?
General Expectations		Q35	What would you want the design tool to do more of?
		Q36	What would you want the design tool to do differently?

Appendix B

Additional Materials – Chapter 4

B.1 Study Participants Demographics

Table B.1: Overview of study participants (Chapter 4).

ID	Agent Group	Age	Role	MechDes Exp. Years	Indus. Exp. Years	CAD Exp. Years	FEA Prof.	DFM Prof.
B1	No Support	22	Student, MA MechE	3–5	0	2–4	5	2
B2	No Support	28	Student, PhD MechE	3–5	1–2	5+	6	2
B3	No Support	27	Researcher, MechE	6–10	0	5+	7	5
B4	No Support	23	Student, MA MechE	3–5	3–5	2–4	7	5
B5	No Support	39	Researcher, MechE	3–5	0	5+	1	1
S1	SocratAIs	26	Student, BS MechE	6–10	0	5+	4	1
S2	SocratAIs	23	Student, MS MechE	3–5	0	2–4	5	5
S3	SocratAIs	22	Student, MS MechE	1–2	1–2	2–4	4	1
S4	SocratAIs	26	Student, PhD MechE	6–10	1–2	5+	1	5
S5	SocratAIs	20	Student, BA MechE	3–5	0	2–4	4	6
H1	Hephaistus	30	Student, PhD MechE	3–5	3–5	5+	4	4
H2	Hephaistus	42	Student, PhD MechE	6–10	6–10	5+	4	1
H3	Hephaistus	22	Student, BA MechE	1–2	1–2	2–4	3	2
H4	Hephaistus	26	Mechanical Engineer	3–5	3–5	5+	2	2
H5	Hephaistus	21	Student, BA MechE	1–2	1–2	2–4	3	5
E1	Expert-Freeform	38	Mechanical Engineer	10+	10+	5+	7	7
E2	Expert-Freeform	26	Mechanical Engineer	1–2	3–5	5+	4	6
E3	Expert-Freeform	29	Mechanical Designer	10+	3–5	5+	5	7
E4	Expert-Freeform	29	Mechanical Engineer	6–10	6–10	5+	5	5
E5	Expert-Freeform	23	Student, MS MechE	3–5	0	5+	2	5

TABLE B.2: Overview of demographics of Autodesk Fusion 360 Generative Design experts who acted as wizards (by facilitating agents) in additional Wizard of Oz sessions (Study 2); GD=Generative Design; Fusion 360 Generative Design software proficiency was self-rated on a 1–7 scale.

ID	Age	Role	MechD Exp.Years	F360 GD Prof.	F360 GD Training Exp.	Paired with
Expert 1	31	Sr. Research Engineer	3–5	6/7	Trained customers, students, colleagues	E2, E5
Expert 2	35	Sr. Research & Design Engineer	6–10	7/7	Taught lectures, trained colleagues	E1
Expert 3	47	Principal Research Engineer	15+	7/7	Trained customer support teams	E4
Expert 4	27	Research and Design Engineer	3–5	6/7	Trained customers, colleagues	E3

B.2 Wizard Training and Guides

The agents SocratAIs and HephAIstus were facilitated by myself with some of the sessions (7 in total) being supported by another collaborator with expertise in mechanical engineering and generative design. Both of us had substantial experience with mechanical engineering tasks, Fusion 360, and Generative Design and had also gained comprehensive practical experience in this specific task. We closely followed the designer's verbalizations and screen actions and paid close attention to the task-specific design steps and challenges as outlined in section 4.2.2, such as specifying the bracket's load cases (forces and structural constraints), modeling appropriate geometry for keeping bolts and dampeners free of material (obstacle geometry), defining DFM parameters such as materials and manufacturing options, and also to support users in evaluating the design previews and generated outcomes. We also paid close attention to inconsistencies between the requirements stated in the design brief and the input parameters set by the designer. Such requirements could be explicit in nature (e.g., the force the bracket needs to hold) or implicit features, such as bolt clearances, which were not explicitly mentioned in the design brief. We did not directly tell the participant what to do but rather provide supportive questions, hints, or suggestions. Furthermore, we primarily sent messages in moments when designers transitioned to a different design sub-task (such as from specifying obstacle geometry to specifying loads), but also to watch out for signals when designers would show hesitation or use hedging expressions (e.g., *"I*



FIGURE B.1: Timeline plots visualizing participant and agent interactions throughout the design task; timelines are divided into lanes, each showing (in)correct GenAI input specifications (diamond shapes) for (1) forces, (2) constraints, (3) bolt and (4) dampener clearances, (5) bounding box (from top to bottom); black and orange vertical lines represent exchanged agent and user messages with purple and black triangles indicating an observable impact on the design process.

am unsure if...") or when participants would pause and think silently. Additionally, we freely sent messages in other situations we considered helpful.

The Expert-Freeform agent-facilitators received fewer instructions since I wanted to observe their natural support behavior. However, experts were told to not directly tell the participants what to do but rather to help them in working on the design task and with the GenAI system.

Wizard Guide SocratAIs and HephAIstus

Observing User Behavior:

- Follow designers' verbalizations and screen actions closely.
- Pay special attention to task-specific design steps and challenges, including:
  - Specifying bracket load cases (e.g., forces and structural constraints).
  - Modeling appropriate obstacle geometry (e.g., bolt and dampener clearances).
  - Defining DFM parameters (e.g., materials and manufacturing options).
  - Supporting users in evaluating design previews and generated outcomes.

Monitoring Design Brief Alignment:

- Identify inconsistencies between the design brief requirements and user inputs.
- Consider both explicit requirements (e.g., specified forces) and implicit requirements (e.g., bolt clearances not explicitly mentioned).

Communication Guidelines:

- Avoid directly telling participants what to do; provide supportive questions, hints, or suggestions instead.
- Send messages when:
  - Designers transition to a new design sub-task (e.g., from specifying obstacle geometry to load cases).
  - Designers exhibit hesitation, use hedging expressions (e.g., "I am unsure if…"), or pause to think silently.

– Use discretion to send messages in other situations deemed helpful.

SocratAIs

Agent introduction:

SocratAIs: Hey! I am a voice agent here to support you during the design task. I can hear what you are saying, and I can see your screen and follow along with you while you work on the task. From time to time, I will ask you questions that are supposed to help you to think through the design task. You can also ask me questions at any time.

HephAIstus

Agent introduction:

HephAIstus: Hey! I am a voice agent here to support you during the design task. I can hear what you are saying, and I can see your screen and follow along with you while you work on the task. Feel free to ask me any questions, and I'll do my best to provide you with answers. From time to time, I'll also chime in with tips and guidance to help you along the way.

Project Planning Activity:

The agent provided metacognitive support by suggesting the user to engage in a **project planning activity** by sharing a pre-generated text document outlining critical project-relevant aspects with the user:

HephAIstus: Hey! I noticed you are working on designing a ship engine mounting bracket and that you are planning to use Autodesk Fusion 360 Generative Design to explore different materials and manufacturing options. Before you start working on the task, I suggest having a planning session together to help you walk through the steps and design considerations in designing the bracket. What do you think?

(waiting for confirmation from user)

HephAIstus: I created a shared doc with a high-level breakdown of the steps and design considerations to be discussed. You can find the link in the chat.

Planning Sheet Document:

1) Project Scope and Requirements

- Define the objectives of the bracket design.
- Understand the operational conditions of the ship engine.
- Identify load types (static, dynamic, thermal) and magnitudes.
- Clarify space constraints and installation considerations.

2) Material Selection

- Discuss different material options (metal alloys, composites, etc.).
- Consider material properties such as strength, weight, corrosion resistance, and cost.
- Review the material performance under extreme marine conditions.

3) Manufacturing Method

- Determine feasible manufacturing methods (casting, machining, additive manufacturing, etc.).
- Evaluate the pros and cons of each method concerning the design objectives.
- Discuss generative design constraints for each manufacturing process.

4) Generative Design Parameters

- Set up load cases and boundary conditions in Fusion 360.
- Define the design space and apply necessary constraints and conditions.
- Choose the resolution of the generative design mesh.

5) Design Constraints and Criteria

- Set criteria for minimum safety factors.
- Define requirements for vibration dampening.

- Consider access for maintenance and installation.

6) Simulation and Analysis

- Plan for simulations to predict performance under various loads.
- Analyze stress distribution, deformation, and fatigue life.
- Review thermal and fluid flow analysis if necessary.

7) Optimization Objectives

- Establish the optimization goals such as weight reduction, strength optimization, cost efficiency, etc.
- Discuss trade-offs between different optimization objectives.

8) Compliance and Standards

- Ensure the design meets marine industry standards and regulatory compliance.
- Consider classification society requirements and certifications.

9) Integration with Existing Systems

- Discuss how the bracket will integrate with the ship's engine and surrounding structures.
- Plan for interfaces with other systems and parts.

10) Lifecycle Considerations

- Consider the lifecycle impacts, such as ease of manufacture, sustainability, recyclability, and end-of-life disposal.
- Maintenance.

Free-body Diagram Sketching Activity:

The agent can suggest that the designer sketch out load case-relevant forces and constraints as a free-body diagram by sharing a link to a 2D drawing canvas containing the side and top view of the bracket as a starting point:

```
HephAIstus: Can you walk me through your load cases and constraints by
            sketching a free-body diagram? I shared a link to a board for
```

you to sketch on in the chat (see Fig. B.2).

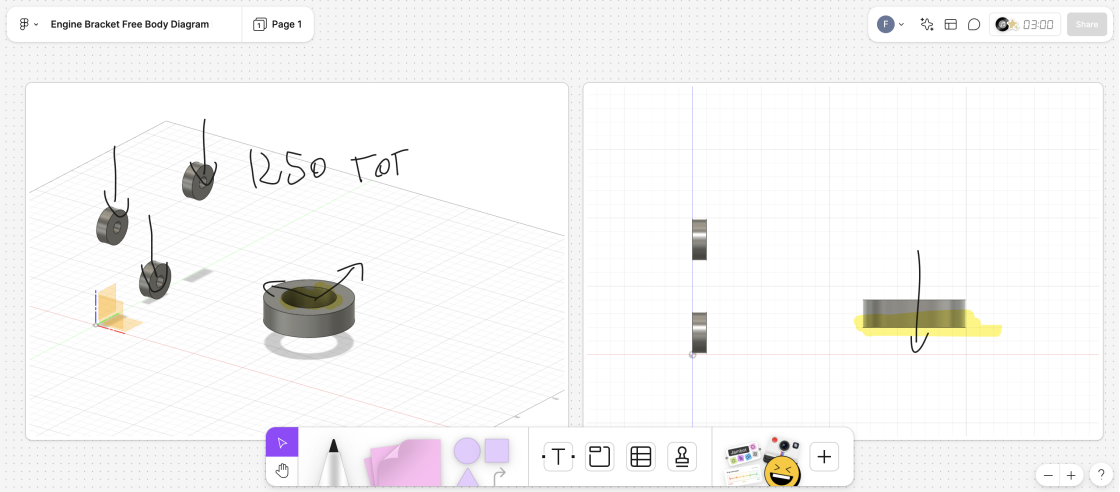


FIGURE B.2: Screenshot of the sketching board HephAIstus sent to a user with scribbles from H3 on it.

Expert-Freeform

Agent introduction:

Expert-Freeform Agent: Hey! I am a voice agent here to support you during the design task. I can hear what you are saying, and I can see your screen and follow along with you while you work on the task. Feel free to ask me any questions, and I'll do my best to provide you with answers. From time to time, I'll also chime in with tips and guidance to help you along the way.

Table B.3: Interview protocol with questions of the semi-structured post-task interview.

Nr	Question
General Feedback on task and thinking aloud	
Q1	Did you encounter any technical difficulties during the design session(s) that limited your ability to work on the task?
Q2	How challenging was the design task of designing an engine bracket for you in general?
Q3	How did it feel to think aloud during the task? Do you think that thinking aloud impacted your ability to complete the task in any way?
Feedback on working with the Generative Design feature	
Q4	Could you tell me what it was like to work with the Fusion 360 Generative Design feature in general?
Q5	Are you satisfied with the final design in general? How closely does it match the design brief?
Q6	Did you encounter any challenges in designing the engine bracket using the generative design feature?
Q7	Could you imagine using this tool in the future?
Q8	How much do you trust the results from the design tool?
Feedback on Support Agent	
Q9	Could you tell me what it was like to work with the design support agent in general?
Q10	Do you remember situations in which you found the agent's support helpful? In which not?
Q11	What would you want the design support agent to do more of?
Q12	What would you want the design support agent to do differently?
Q13	Would you use a tool like the design support agent in your work? Why or why not?
Q14	How did you like the frequency of messages?
Q15	How did you like the planning doc and sketching board?
Q16	In which phases did you find the support more or less helpful?
Q17	How useful did you find the questions that the agent asked you? Do you remember specific questions that you found helpful or unhelpful? Please explain.
Q18	Is there anything else you would like to share with us, or do you think we should know about?

Appendix C

Additional Materials – Chapter 5

C.1 Study Participants Demographics

Table C.1: Overview of study participants. Note: The baseline group (N1–N5) is identical to the baseline group from the study in Chapter 4 since the study design is identical to this work.

ID	Group	Age	Role	MechDes Exp. Years	Indus. Exp. Years	CAD Exp. Years	FEA Prof.	DFM Prof.
A1	HyWoZ Cycle 1	20	Student, BS MechE	3 – 5	1 – 2	5+	3	2
A2	HyWoZ Cycle 1	25	Student, MA MechE	3 – 5	1 – 2	2 – 4	5	6
A3	HyWoZ Cycle 1	21	Student, BS MechE	3 – 5	None	2 – 5	4	5
A4	HyWoZ Cycle 1	22	Student, MS MechE	6 – 10	3 – 5	5+	6	5
A5	HyWoZ Cycle 1	22	Student, MS MechE	1 – 2	0	2 – 4	3	2
B1	HyWoZ Cycle 2	19	Student, BS AerospE	6 – 10	None	5+	4	3
B2	HyWoZ Cycle 2	21	Student, BS MechE	3 – 5	1 – 2	5+	3	5
B3	HyWoZ Cycle 2	20	Student, BS MechE	3 – 5	3 – 5	2 – 4	5	7
B4	HyWoZ Cycle 2	19	Student, BS MechE	1 – 2	None	2 – 4	1	1
B5	HyWoZ Cycle 2	43	Engineer	10+	10+	5+	4	7

C.2 Open-Source Toolkit Details

The toolkit is released as open-source software under the MIT license, with the source code available in a public Git repository[1]. It is implemented as a modular server-client web application that allows components to be adapted or replaced, such as swapping the JSON store for a database or adding custom data processing pipelines. It provides basic extension points that enable connection with hybrid Wizard-of-Oz workflows.

[1] `https://github.com/Augmented-Design-Capability-Studio/replay-prompt-eval`

TABLE C.2: Selected Git commits specifically related to iterative refinement of GPT system prompts (not a record of all commits).

Commit	Message
645a0bf	Initial commit
14a39b0	Prompt engineering & front end fixes
e949a69	Added "reason for denying message" to prompt
9d62ddc	Added additional support strategies to the prompt
bac875d	Refined support messages description
1cb8054	Prompt message generation (time stamp fix)
53b87d0	Software tip type and planning phaser
15d197d	Prompts separated
3943d7e	Changes WOZ to include phase radio group and updated prompt to respect phase
6840f8e	Added task phase classification mechanism
13b9d5a	Task phase classification: More specific explanation of challenges and how to recognize these.

Workflow Scenario

A typical usage scenario unfolds as follows:

1. The user starts by loading a session recording by selecting the corresponding file from the session dropdown menu.
2. They navigate through the replay (video and transcript) to a moment of interest.
3. By clicking the message generation button, an LLM response is requested for the selected context. Once generated, the new message appears in the interface.
4. The user evaluates the message, optionally assigning a five-point rating and leaving a comment.
5. They can iterate by adjusting the system prompt, replaying the same session, or testing prompts across different sessions.

In addition, the application includes a **coding view** that lets users label and annotate generated messages. This view provides an overview of all outputs for a given session, along with associated ratings and comments, and allows jumping back to the corresponding video moment. Updated annotations are stored automatically in the backend database.

C.2.1 Architecture and Components

Frontend (Replay & Interaction).

- **Replay interface:** A web-based video player with synchronized transcripts. Users can load recordings (MP4 + transcript/SRT) through a session dropdown and then play, scrub, or pause the recording.
- **System prompt editor:** The current system prompt can be edited in a dedicated text field.
- **Message generation controls:** A button allows users to trigger LLM responses for the selected context. Generated messages are displayed directly in the interface.
- **Evaluation interface:** Each generated message can be optionally rated with a five-point rating widget and annotated with a free-text comment, which are stored in the backend database.

Backend.

- **Media management:** Users drop video files (MP4) and transcripts (SRT) into a designated backend `media` folder, from which the frontend loads them dynamically.
- **Prompt assembly pipeline:** The backend handles the assembling of prompt and context from the video frames and transcript, and issues LLM API inference calls.
- **Data persistence:** Ratings and comments are stored in a lightweight file-based JSON database.
- **Server implementation:** Implemented as a Node.js service with distinct API endpoints

C.2.2 Details on Hybrid-WoZ User Study Methodology

Video interaction analysis to determine agent impact

I used *video interaction analysis* [33] on the think-aloud recordings to examine how the agent's support affected participants' design process. To assess the effect of each message, I checked whether participants considered new design aspects after a message appeared, either in their responses or in the actions they took.

For more details, see [129] Section 5.5.2.

Design Outcome Analysis

I assessed the feasibility of participants' final designs by comparing their submitted engine brackets with the requirements in the design brief. Each design was scored on five criteria, with one point awarded for each criterion met:

1. The structural soundness was validated using finite element analysis (FEA).
2. The feasible load case setup was checked in their Fusion 360 project file.
3. The optimized mass was not extremely light or heavy.
4. The part had feasible fastener clearance (i.e., clear bolt holes)
5. The part's mass and volume fit within the acceptable bounding dimensions.

The GenAI solver generated approximately 20 designs per round. Because the quality of these outputs could vary, I asked participants to pick three parts they considered feasible. From those three, I chose the highest-scoring part to serve as their final design.

For more details, see [129] Section 5.5.1.

Appendix D

Additional Materials – Chapter 6

D.1 Study Participants Demographics

Table D.1: Overview of study participants (Chapter 6)

ID	Age Range (Years)	Gender	Job Title	Frequency of PowerPoint Usage
P01	18-29	Female	Senior Product Designer	Multiple times per week
P02	30-39	Female	Senior Product Manager	Every day
P03	18-29	Female	Research Intern (PhD)	Multiple times per month
P04	18-29	Male	Senior Human Factors Engineer	Every day
P05	30-39	Female	Senior Designer	Multiple times per week
P06	40-49	Female	Senior User Researcher	Multiple times per week
P07	30-39	Male	Senior Product Designer	Every day
P08	18-29	Female	Content Designer	Multiple times per month
P09	30-39	Male	Research Intern (PhD)	Multiple times per month
P10	18-29	Non-binary	Research Intern (PhD)	Multiple times per year
P11	30-39	Female	Design Program Manager	Multiple times per week
P12	40-49	Female	UX Researcher	Multiple times per month

D.2 Further System Implementation Details

Tag Suggestion Mechanism

To generate intent tag suggestions based on a user's existing tags on the slide deck steering canvas, we construct a call to GPT comprising the static `SYSTEM_PROMPT` and the dynamic `USER_CONTEXT` below. `USER_CONTEXT` is a string dynamically constructed from all active intent tags `[attribute:value]` for each tag group `(Narrative, Visual Style, Content Sources)`.

```
SYSTEM_PROMPT (static):
You are an assistant to help users author slide presentations in PowerPoint.
You are helping the user develop the narrative and visual style and extracting
    content from source documents relevant to the presentation.
You will receive instructions to help with one of these three buckets: Narrative,
    Visual Style, and Content Sources.
The user will already have descriptions, tags, and media elements that describe
    different aspects of the presentation.
Your task is to suggest additional keywords and concepts for each of these buckets.
The user will prompt you in the format:
**Bucket** attribute1:value1, attribute2:value2, ...
A bucket might be empty, in which case you should suggest concepts based on the other
     buckets.
A value might not be associated with an attribute.
For each user prompt, suggest 7 concepts for each of the requested buckets.
A concept is a keyword or phrase that is relevant to the user's presentation. A
    concept should be a single word or a short phrase. Each concept should have an
    attribute and value.
Return your response in JSON format. Only return JSON format without any additional
    Markdown wrapper code blocks.

Here is an example of the JSON return format:
{
    "Narrative": ["attribute1:value1", "attribute1:value1", ...],
    "Visual Style": ["attribute1:value1", "attribute1:value1", ...],
    "Content Sources": ["attribute1:value1", "attribute1:value1", ...]
}

USER_CONTEXT (dynamic):
**Narrative** attribute1:value1, attribute2:value2, ...
**Visual Style** attribute1:value1, attribute2:value2, ...
**Content Sources** attribute1:value1, attribute2:value2, ...
```

Outline Generation From Intent Tags

To generate an outline based on a user's existing tags on the slide deck steering canvas, we construct a call to GPT comprising the static `SYSTEM_PROMPT` and the dynamic `USER_CONTEXT` below. `USER_CONTEXT` is a string dynamically constructed from all active

intent tags `[attribute:value]` for each tag group (`Narrative, Visual Style, Content Sources`). GPT is instructed to generate an outline in Markdown format, including the URLs of image reference tags if present.

```
SYSTEM_PROMPT (static):
You are an assistant to help users author slide presentations in PowerPoint.
You are creating an outline for the user's presentation.
The user will have descriptions and keywords that describe different aspects of the
    presentation.
The user will prompt you in the format: attribute1:value1, attribute2:value2, ...
Ignore all the attributes related to the visual style.
If the content sources include ImageUrl attributes, you should include these images
    in the outline as images in the markdown.
Respond with an outline of the presentation in the format:
section title - section content bullet points.
Return the outline in markdown format without ```markdown at the start and end. Only
     return markdown content for the outline.

USER_CONTEXT (dynamic):
**Narrative** attribute1:value1, attribute2:value2, ...
**Visual Style** attribute1:value1, attribute2:value2, ...
**Content Sources** attribute1:value1, attribute2:value2, ...
```

Slide generation mechanism

To generate the JSON structure for a slide deck, we construct a call to GPT comprising the static `SYSTEM_PROMPT` and the dynamic `USER_CONTEXT`. `USER_CONTEXT` is a string containing the `PRESENTATION OUTLINE` in markdown, `META INFORMATION`, which is a string containing all active intent tags `[attribute:value]`, and optionally the `REFERENCE SLIDE DECK TEMPLATE`, which is a JSON string of a reference slide deck (if provided). The general slide deck template schema and slide generation mechanism are based on the Spectacle[1] library.

```
SYSTEM_PROMPT (static):
Your task is to generate a slide presentation deck from an outline and meta
    information.
You will receive the presentation OUTLINE in markdown format from the user.
```

[1] https://github.com/FormidableLabs/spectacle

```
You will also receive additional META INFORMATION about the desired presentation's
    Narrative, Visual Style, and Content Sources.
The META INFORMATION will describe different aspects of the presentation.
You will receive the META INFORMATION in the format:
**Bucket** attribute1:value1 (optional description), attribute2:value2 (optional
    description), ...
A bucket might be empty, in which case you should suggest concepts based on the other
     buckets.
A value might not be associated with an attribute.
The META INFORMATION might also contain USER DOCUMENTS with content that has to be
    included in the presentation.

GENERATING SLIDES
Your task is to generate a slide deck. Each slide has to consist of a LAYOUT, CONTENT
    , and THEME.

LAYOUT:
The layout determines the arrangement of the content on the slide.
There are five different layouts to choose from:
"title", "listOrParagraph", "verticalImage", "fullImage".

CONTENT:
Each layout takes different content parameters depending on the layout type.
    Optionally, each slide can take an image url as background image in the content.
The background image will replace the background color of the slide.
The "listOrParagraph" and "verticalImage" layouts take either a list or a paragraph
    as content besides the title and image.
For bullet lists, never use more than 3 bullets.

THEME:
The theme determines the fonts, colors, and font sizes used in the slide.

fonts: { header: fontname, text: fontname }
"header" is the font used for titles and "text" is the font used for body, lists,
    paragraphs.
There are six fonts to choose from: "Quicksand", "Playfair Display", "Montserrat", "
    Merriweather", "Roboto" and "Roboto Condensed".

colors: { primary: color, secondary: color, tertiary: color }
```

```
"primary" is the main color used for text, "secondary" is the color used for accents,
     and "tertiary" is the color used for backgrounds.

fontSizes: { h1: size, text: size }
"h1" is the font size used for titles and "text" is the font size used for body text.

REFERENCE SLIDE DECK TEMPLATE
The user might also provide a REFERENCE SLIDE DECK TEMPLATE with a slide deck
    template they want to use for the visual style.
Important: If a REFERENCE SLIDE DECK TEMPLATE is provided, use only this template for
     the visual style (font, backgroundImage, sizes, colors) and IGNORE all visual
    style information from the generic example slide deck below.
ALWAYS use the backgroundImge URLs from the REFERENCE SLIDE DECK TEMPLATE on each
    slide!

REWORK CONTENT FROM THE OUTLINE
You should rework the content from the OUTLINE to fit the slide deck based on the
    desired format specified in the META INFORMATION.
For example, if the META INFORMATION specifies that the presentation should have a
    formal tone, you should rework the content to fit this tone.
If the META INFORMATION specifies that the presentation should have a more verbose
    text, you should rework the content to fit this style.

YOUR TASK:
Your task is to compose a slide deck based on the outline and to adapt the content to
     the layout and theme.
Your task is to define the theme and layout of the slides based on the OUTLINE and
    META INFORMATION provided by the user.
Generate the number of slides based on the outline and meta information provided by
    the user.
Keep the visual style consistent across all slides.
Especially the fonts, colors, background color and font sizes.
You should return the slide deck in JSON format. Only return JSON format without any
    additional Markdown wrapper code blocks.

Here is a generic example of a slide deck in JSON format:
[{
   slideNumber: 1,
   layout: "title",
```

```
    content: {
      title: "A presentation about something",
      subtitle: "by someone",
      backgroundImage: "url(<image url placeholder>)"
    },
    theme: {
      fonts: {
        header: '"Playfair Display", serif',
        text: '"Quicksand", sans-serif',
      },
      colors: {
        primary: "#000",
        secondary: "#000",
        tertiary: "#fff",
      },
      fontSizes: {
        h1: "100px",
        text: "44px",
      },
      space: [16, 24, 32],
    },
  },{
    slideNumber: 2,
    layout: "listOrParagraph",
    ...
    (PARTS OMITTED)
    ...
}]


USER_CONTEXT (dynamic):
PRESENTATION OUTLINE:
$CURRENT_OUTLINE_IN_MARKDOWN_STRING

META INFORMATION:
$CURRENT_ACTIVE_TAGS_AS_ATTRIBUTE:VALUE_PAIRS_CLUSTERED_BY_GROUP

(optional) REFERENCE SLIDE DECK TEMPLATE:
$REFERENCE_SLIDE_DECK_JSON
```

Mechanism to create tags for an existing slide (tag grounding act)

To generate a collection of intent tags based on a provided image of the current slide, we construct a call to GPT comprising the static `SYSTEM_PROMPT` and the dynamic `USER_CONTEXT` below. `USER_CONTEXT` is a string of the current slide in a base64 image format. In addition to the slide image, we experimented with including the active steering board tags in the prompt but refrained from including these in this version of IntentTagger for simplicity. Future explorations outside the scope of this paper should investigate how "global" steering board-level intent tags can meaningfully propagate to individual slides and how changes on single slides can propagate back to the global steering board level while meaningfully resolving conflicts between global and slide-level tags.

```
SYSTEM_PROMPT (static):
You are an assistant to help users author slide presentations in PowerPoint.
You will receive an image of a slide from their slide presentation deck.
Your task is to analyze the slide deck and return descriptive attributes that best
    describe the slide.
You should return these attributes related to three categories: Narrative, Visual
    Style, and Content Sources.
Each attribute should consist of a label and a value, like Tonality:Formal or
    Typography:Modern.
For each bucket, return between 2 to 6 attributes.
Return your response in JSON format.
Only return JSON format without any additional Markdown wrapper code blocks.

Here is an example of the JSON format:
{
    "Narrative": ["attribue1:value1", "attribue1:value1", ...],
    "Visual Style": ["attribue1:value1", "attribue1:value1", ...],
    "Content Sources": ["attribue1:value1", "attribue1:value1", ...]
}

USER_CONTEXT (dynamic):
$IMAGE_OF_CURRENT_SLIDE_AS_BASE64_STRING
```

D.3 Semi-structured Task Outcomes

This section contains screenshots of IntentTagger (deck steering board and slide panel) taken at the end of each participant's semi-structured slide creation task from our user study (see study phase 3 at section 6.5.2).

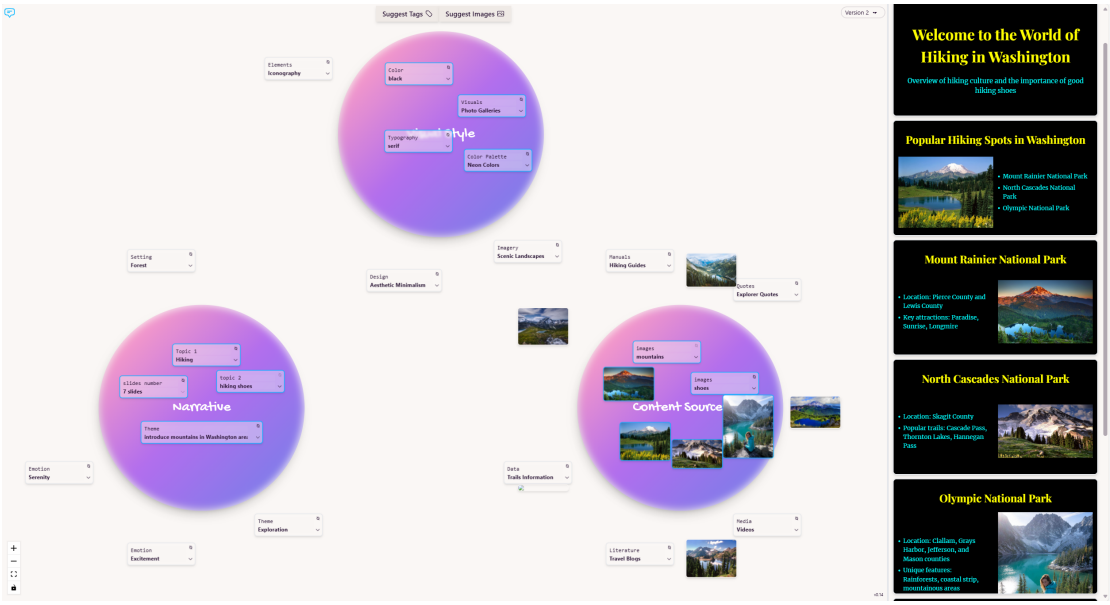


FIGURE D.1: Semi-structured task outcome of P01

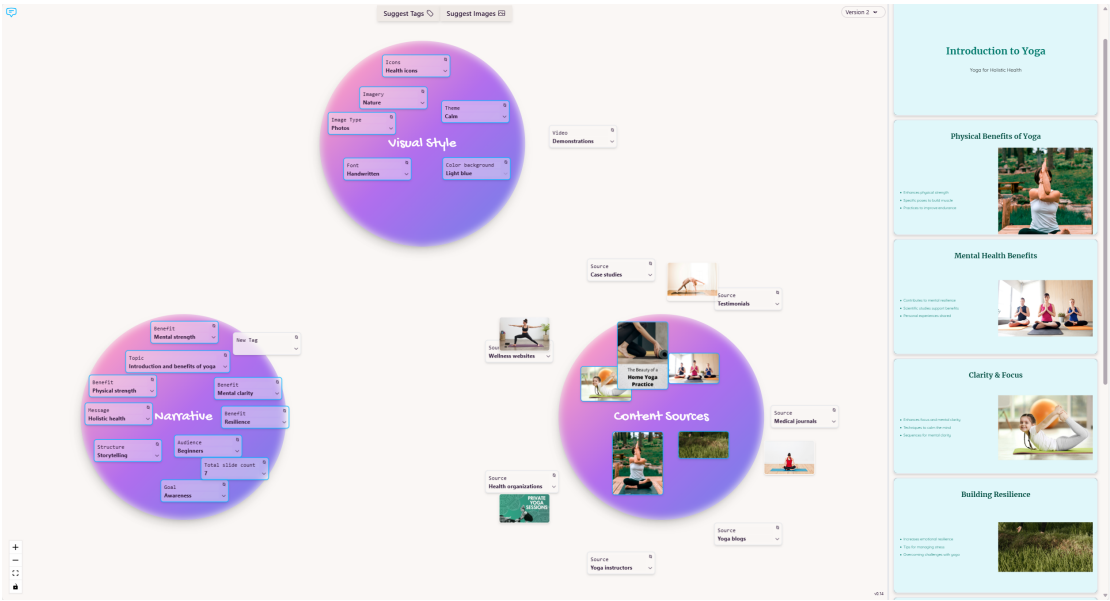


FIGURE D.2: Semi-structured task outcome of P02

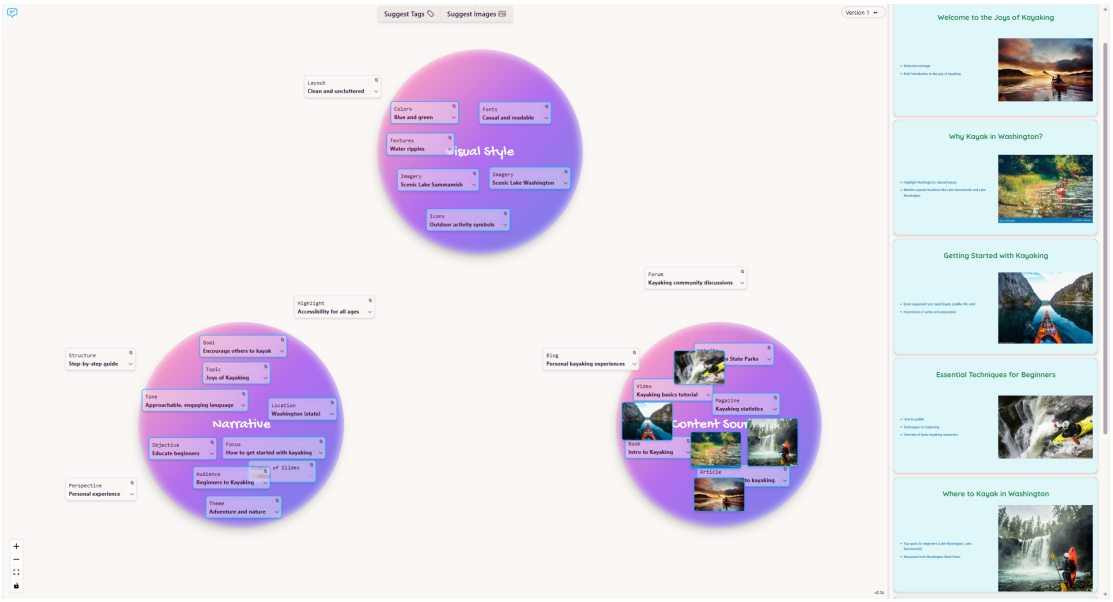


FIGURE D.3: Semi-structured task outcome of P03

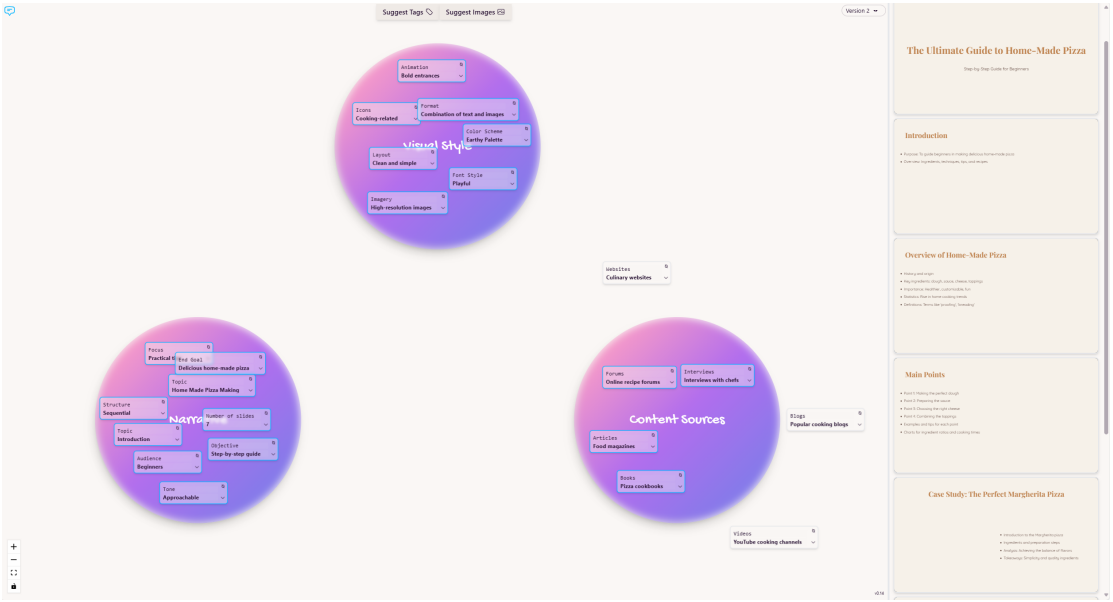


FIGURE D.4: Semi-structured task outcome of P04 (Note: The user was unable to add images to their presentation, due to a temporary outage of the image search API.)

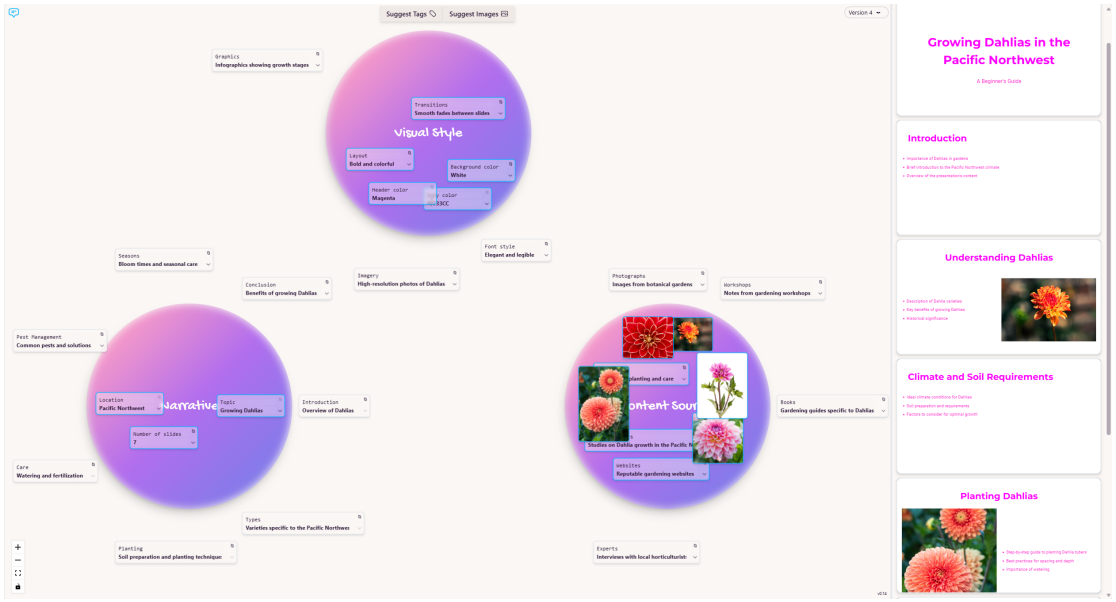


FIGURE D.5: Semi-structured task outcome of P05

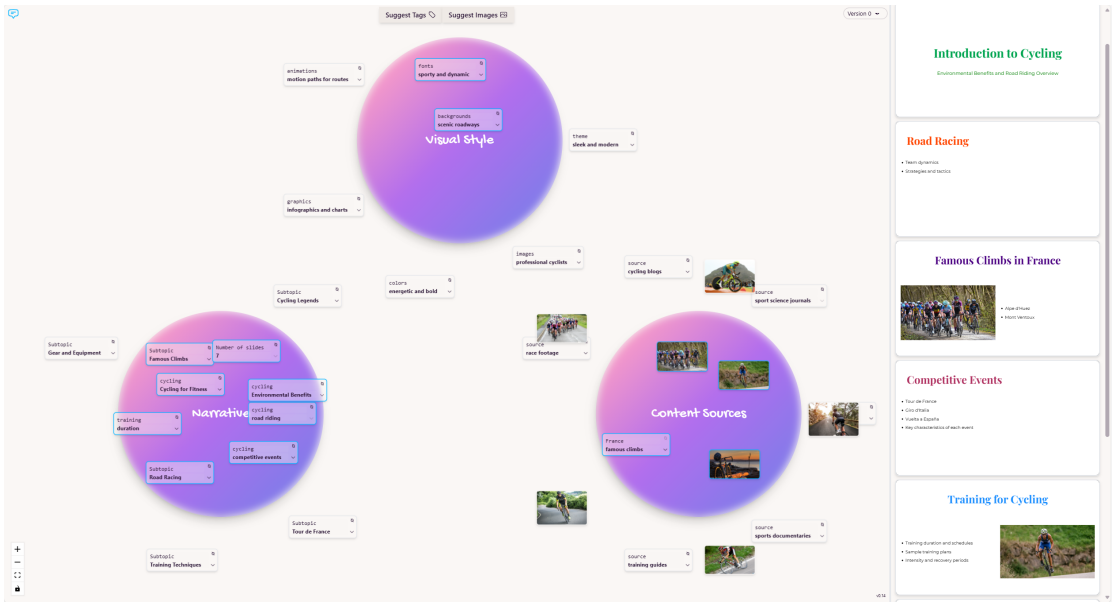


FIGURE D.6: Semi-structured task outcome of P06

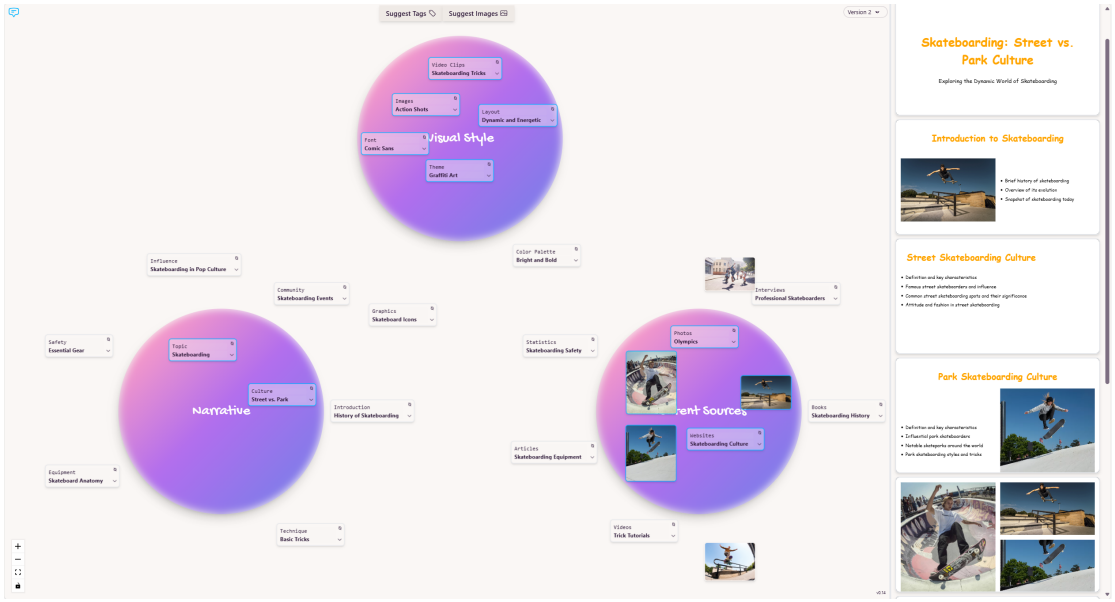


FIGURE D.7: Semi-structured task outcome of P07

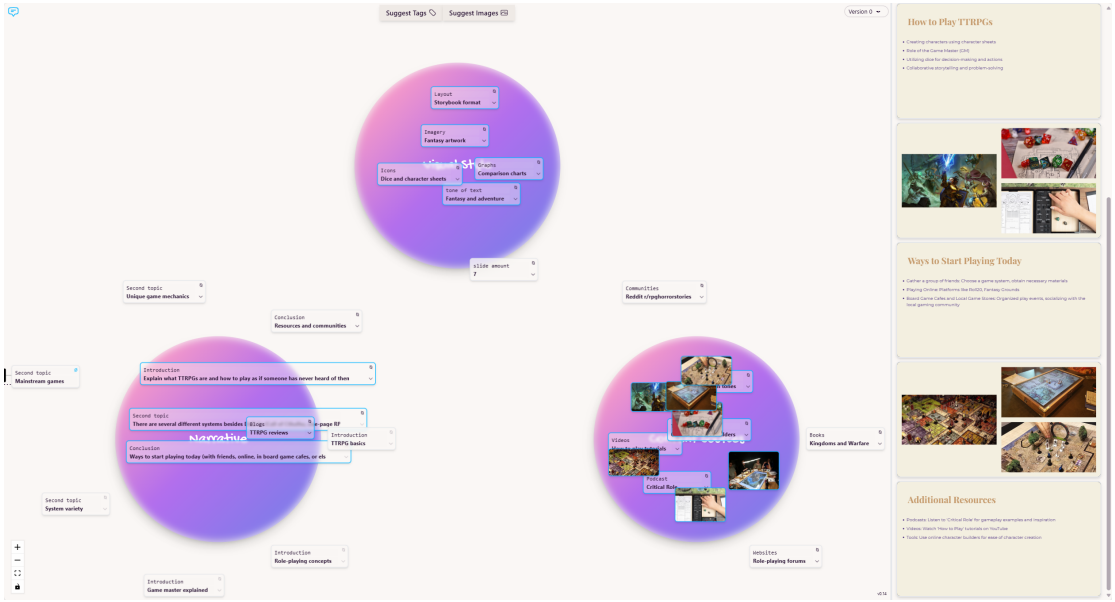


FIGURE D.8: Semi-structured task outcome of P08

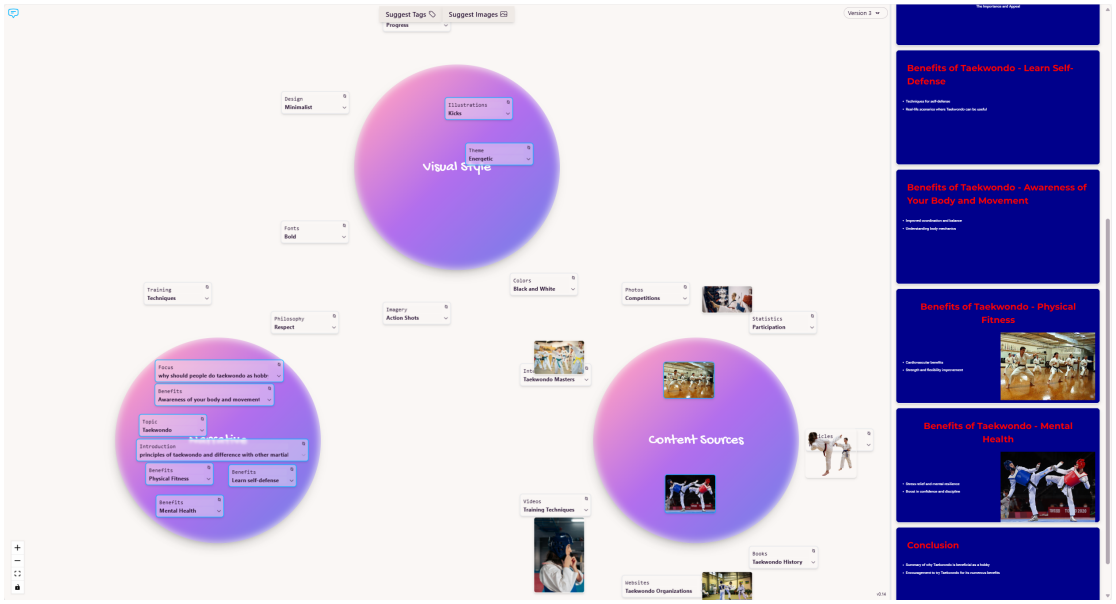


FIGURE D.9: Semi-structured task outcome of P09

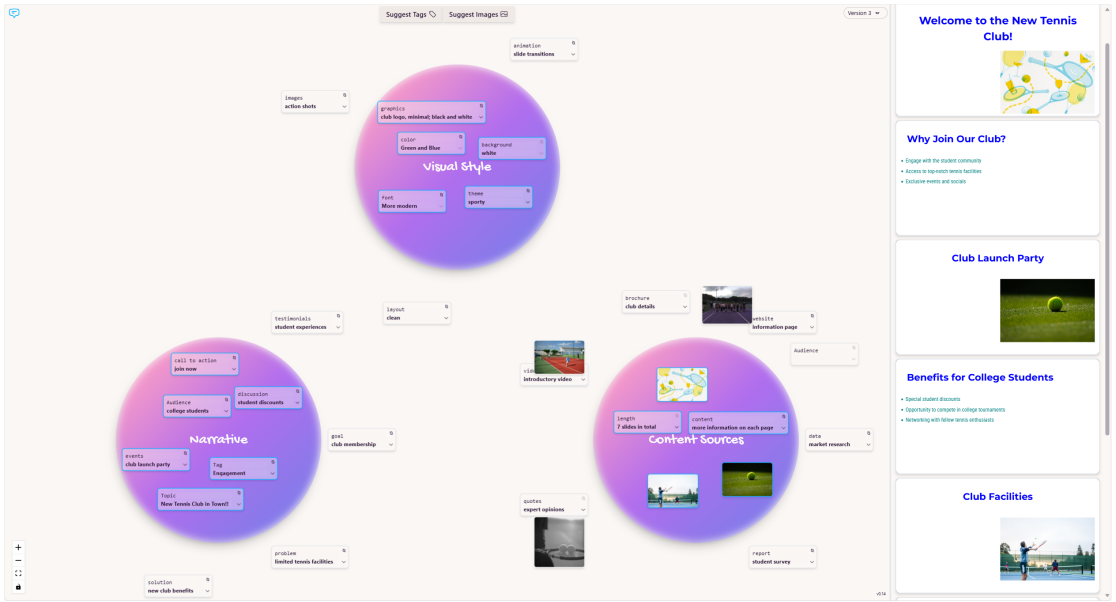


FIGURE D.10: Semi-structured task outcome of P10

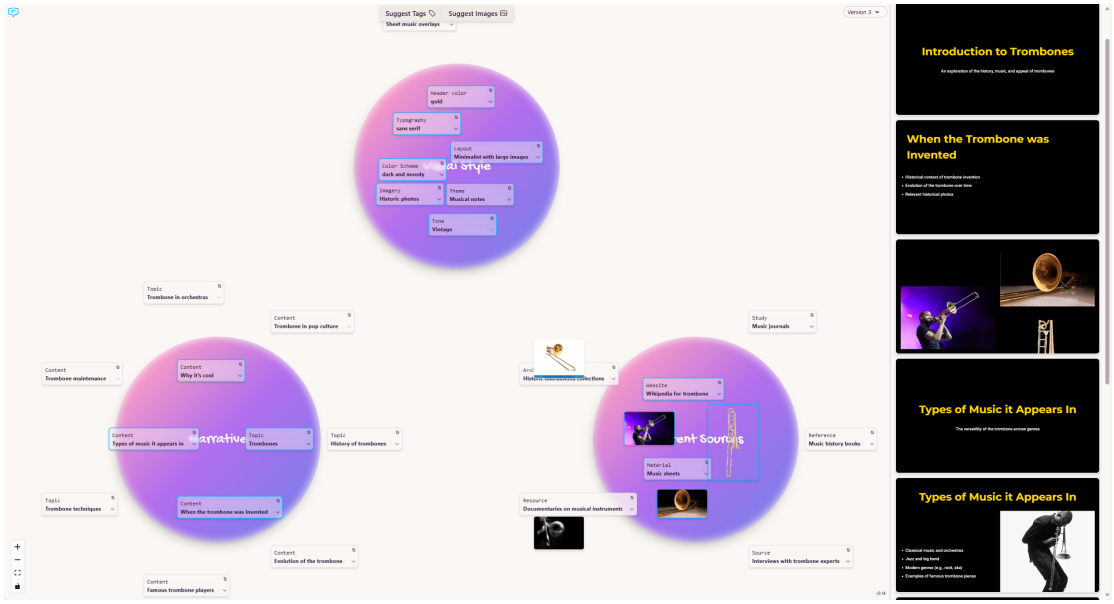


FIGURE D.11: Semi-structured task outcome of P11

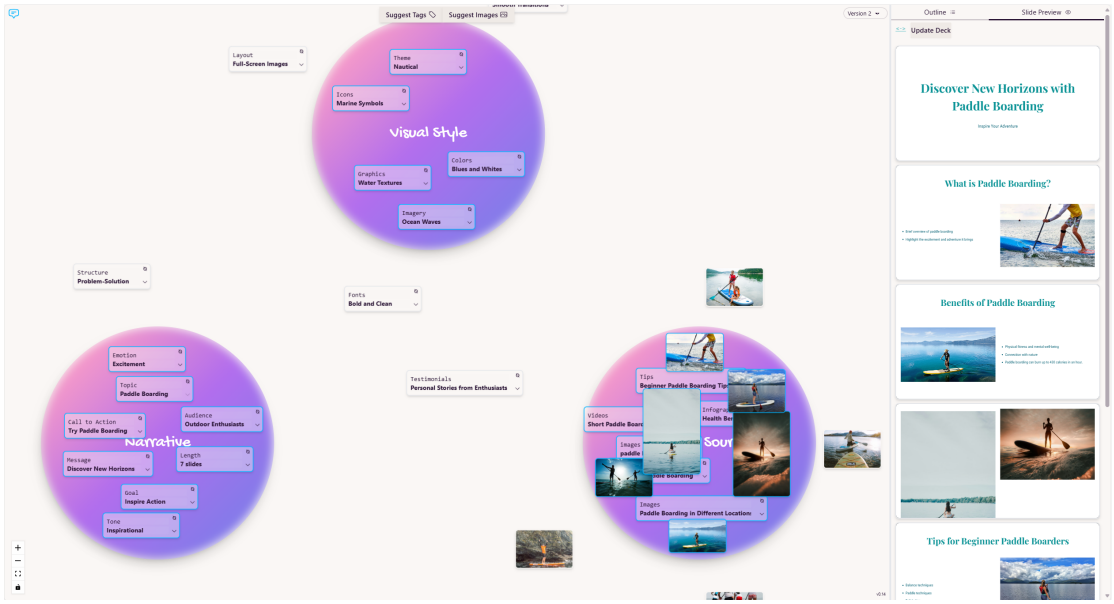


FIGURE D.12: Semi-structured task outcome of P12

Appendix E

Additional Materials – Chapter 7

E.1 Study Participants Demographics

Table E.1: Overview of study participants (Chapter 7)

PID	Cond.	Age	Role / Program	DS Exp.	Python	ML Exp.
P03	MK	25	M.Eng Comp Sci & Eng	3–5	Advanced	Regular
P06	MK	23	MS Data Science & Analytics	3–5	Advanced	Regular
P07	B	21	BS CS & Data Science	3–5	Intermediate	Coursework
P09	B	18	BS Data Science	3–5	Intermediate	Regular
P10	MK	28	Data Engineer	3–5	Intermediate	Coursework
P11	MK	26	PhD Statistics	6–10	Advanced	Regular
P12	B	28	PhD Informatics & Computing	3–5	Intermediate	Regular
P13	B	20	BS Applied Math & Statistics	3–5	Intermediate	Coursework
P14	B	63	MS Data Science	6–10	Advanced	Regular
P16	B	27	AI Quality Analyst	3–5	Advanced	Coursework
P17	MK	21	MS Data Science	3–5	Intermediate	Regular
P18	MK	26	MS Applied Math & Stats	3–5	Intermediate	Coursework
P20	MK	27	MS Computer Science	3–5	Advanced	Regular
P21	MK	22	MS Data Science	3–5	Advanced	Regular
P22	B	31	MS Data Analytics	6–10	Advanced	Regular
P24	B	38	MBA	> 10	Advanced	Regular
P25	B	25	MS Data Science	3–5	Advanced	Regular
P27	MK	27	MS Data Science	3–5	Advanced	Regular
P28	B	22	MS Data Science	3–5	Intermediate	Regular
P29	MK	28	PhD Machine Learning	3–5	Intermediate	Regular

E.2 Study Materials

This section provides the participant-facing materials used for the Riverdale County data-fitness assessment task in Chapter 7. Note: Parts of the Case Brief E.2.1 and the Fitness Assessment Memo Template E.2.3 were developed with LLM support (GPT-5.5) with subsequent reviews and edits by the author.

E.2.1 Case Brief and Task Prompt

Use Case Brief: Riverdale County Property Reassessment Model

Client goal:

Riverdale County wants a predictive model to flag residential properties (homes) that are likely **systematically undervalued** relative to market price, so county appraisers can reassess them.

How the model will be used:

Every quarter, the model will score all approximately 12,000 active residential properties in the county, including properties that have **not sold recently**. This means it cannot rely on features that only exist in the record of a recent sale. Predictions must be **explainable** to property owners who dispute their reassessment. The county's legal team requires that **no protected-class characteristics** (e.g., race, income) appear in the model, even as proxies.

Available data:

The dataset contains residential property sales in Riverdale County **between 2006 and 2010** (approximately 2,930 records, 22 features). A data dictionary is provided in the data notebook.

Important context:

The county is aware that the dataset covers 2006–2010 and is actively collecting more recent records. You may assume the temporal gap will be closed before deployment. Your assessment should focus not on data recency itself, but on which features are likely to **remain stable and meaningful over time** versus which may drift or become inconsistent.

Your deliverable:

A fitness assessment memo for your project lead. She needs to decide whether to recommend this dataset to the client, and if so, with what caveats.

The key question:

Once you identify which variables cannot be used—due to legal constraints, operational unavailability, or data quality problems—does the remaining dataset still contain enough predictive signal to do the job?

It is not enough to say "fix these issues and the data is fine." You must assess whether what is left after removing problematic variables is genuinely useful for predicting undervaluation, or whether the constraints strip out so much signal that the model would not be meaningfully better than a manual process.

E.2.2 Data Dictionary

Table E.2: Data dictionary for the Riverdale County Properties 2006–2010 dataset used in the Chapter 7 study task.

Column	Description
`sale_price`	Recorded sale price at time of transaction ($)
`construction_grade`	Overall material and finish quality (1–10 scale)
`above_ground_sqft`	Above-ground living area (sq ft)
`district`	District identifier (28 categories)
`garage_capacity`	Garage capacity (number of cars)
`basement_sqft`	Total basement area (sq ft)
`construction_year`	Year property was originally built
`renovation_year`	Year of most recent renovation (= construction_year if none)
`total_bathrooms`	Total bathrooms (full + 0.5 × half)
`fireplace_count`	Number of fireplaces
`lot_area`	Lot area (sq ft)
`kitchen_quality`	Kitchen quality rating (Ex / Gd / TA / Fa / Po)
`exterior_quality`	Exterior material quality rating (Ex / Gd / TA / Fa / Po)
`basement_quality`	Basement height/quality rating (Ex / Gd / TA / Fa / Po); some values missing
`sale_year`	Year of sale (2006–2010)
`sale_month`	Month of sale (1–12)
`sale_condition`	Condition of sale (Normal / Abnormal / Partial / etc.)
`zoning_type`	General zoning classification (residential sub-types)
`building_type`	Type of dwelling (single-family / townhouse / duplex / etc.)
`land_contour`	Flatness of property (Lvl / Bnk / HLS / Low)
`functional_rating`	Home functionality rating (Typ / Min1 / Mod / Sev / etc.)
`lot_config`	Lot configuration (Corner / CulDSac / Inside / etc.)

E.2.3 Fitness Assessment Memo Template

Fitness Assessment Memo

To: Project Lead

Re: Riverdale County Reassessment Model

Dataset: `riverdale_properties.csv`

Section 1: Critical Risks (approximately 150 words)

What are the 2–4 (or more) most serious data problems for this use case? For each state:

- What you found (cite a specific evidence/metric/number: row count, percentage, distribution, or cross-tab)
- Why it matters for *this* application
- Severity: **High / Medium / Low**—one sentence why

[Your critical risks here]

Section 2: Tradeoffs (approximately 150 words)

Name at minimum 1–3 (or more) features that are *both useful and problematic*, depending on how the model is built or deployed. For each, explain both sides of the tension and cite at least one quantitative property that makes the tradeoff concrete (e.g., predictive strength, proportion of affected rows).

[Your tradeoffs here]

Section 3: Remaining Signal (approximately 100 words)

After setting aside variables you cannot use—legal exclusions, operationally unavailable features, quality-compromised columns—what usable variables remain? Name them and assess honestly: is their collective predictive signal sufficient to identify undervalued properties, or do the exclusions strip out too much?

Do not just say "fix the issues and the data is usable." Assess what is left.

[Your remaining signal assessment here]

Section 4: Verdict-Changing Unknown (approximately 75 words)

What is the single most important thing about the deployment context that you *cannot* answer from the data—but that would materially change your fitness verdict if you knew it?

[Your key unknown here]

Section 5: Fitness Verdict (approximately 75 words)

Ready / Conditionally Ready / Not Ready

Justify in 2–3 sentences. Reference at least one specific finding by name (a column, statistic, or pattern you observed).
[Your verdict here]

E.3 Chapter 7 Coding Schemes and Measure Details

This appendix documents the coding schemes used for the outcome and process measures in Chapter 7.

TABLE E.3: Finding inventory and evidence rules used in the LLM-assisted audit. T1 were surface-level findings (column-inferable); T2 were deeper-level findings (requiring computational evidence). Injected anomalies were deliberately added or amplified in the study dataset.

ID	Finding	What qualified as passing evidence	Injected anomaly
T1-A	`district` as proxy / fairness risk	Named `district` specifically and linked it to the legal or fairness constraint in the use-case brief.	No
T1-B	`basement_quality` missingness	Cited a missingness rate of roughly 15–20%, or noted the structural pattern that missing values indicated no basement.	No
T1-C	`sale_condition` label contamination	Identified non-Normal sales as contaminating `sale_price`, citing the overall non-Normal proportion (about 18%) or naming categories with reasoning.	No
T1-D	Operational unavailability of sale-time features	Identified features unavailable at quarterly prediction time and explicitly linked them to the deployment constraint.	No
T1-E	General temporal stability concern	Noted that 2006–2010 features may drift over time; generic recency complaints already addressed by the use-case brief did not qualify.	No
T2-A	Assessor grade inflation	Detected an upward trend in `construction_grade`, `kitchen_quality`, or `exterior_quality` over `sale_year`, supported by per-year means, a trend statistic, or a notebook cell with year-grouped output.	Yes
T2-B	`sale_condition` district concentration	Identified non-random clustering of Partial or Abnormal sales in specific districts, supported by district-level proportions or notebook cross-tabulation.	Yes
T2-C	`lot_area` unit outliers	Identified anomalously low `lot_area` values in the 300–900 range, with an accepted count of about 35–45 rows, a minimum value, histogram observation, or notebook output showing the cluster.	Yes
T2-D	`renovation_year` redundancy	Showed that `renovation_year` equals `construction_year` for roughly 50–65% of properties, using a memo proportion or notebook comparison.	No
T2-E	Quality-block multicollinearity	Reported correlation between two or more quality columns, or between quality columns and `above_ground_sqft`, using a correlation coefficient or notebook correlation matrix.	No
T2-F	`district` × `construction_year` confound	Showed that `district` correlates with building age, using a memo correlation or a notebook cell analyzing the cross-relationship.	No
T2-G	Abnormal-sale price discount quantified	Quantified that Abnormal sales transact below Normal sales at equivalent square footage, using an approximate discount magnitude (about 18%) or notebook grouped comparison.	No
T2-H	Predictive variance estimate	Provided a quantitative estimate of predictive signal, such as variance explained, feature correlation with `sale_price`, or partial R^2.	No

E.3.1 Finding Inventory Coding and LLM-Assisted Audit

The outcome analysis used an LLM-as-judge pipeline to code each participant's memo and notebook against the pre-identified finding inventory (Opus 4.7). The judge received a finding inventory, the participant's memo, and relevant notebook evidence, then returned structured JSON outputs with binary codes and supporting evidence references.

The coding followed two evidence gates:

- **Tier 1 gate:** The memo or notebook had to name the finding or clearly describe the underlying issue and connect it to the reassessment use case.
- **Tier 2 gate:** The participant had to state the finding and provide computational support in the notebook, such as a table, plot, statistic, model output, or code cell that directly substantiated the claim.

Table E.3 lists the full finding inventory and the operational rule used to determine whether each finding passed the evidence gate. The judge prompt was calibrated against five human-audited anchor participants. After calibration, I manually audited the resulting codes, with special attention to the Tier 2 evidence gates, injected anomalies, novel findings, and notebook-only findings.

E.3.2 Process Coding Scheme

Coding agent request types

For each coding agent event, I coded the participant's request according to the main purpose of the request:

- **Execute:** The participant asked the coding agent to implement a concrete notebook action, such as loading data, generating a plot, computing a statistic, fitting a model, or editing code.
- **Verify:** The participant asked the agent to check, interpret, sanity-check, or validate an existing result, code cell, or analytical claim.
- **Challenge:** The participant pushed back on an agent response, questioned its correctness, asked it to revise a flawed answer, or pointed out an inconsistency.
- **Explore:** The participant asked for open-ended analytical directions, possible next steps, or additional checks to consider.

- **Synthesize:** The participant asked the agent to summarize findings, connect multiple results, draft memo text, or pull several analytical threads together.

Note: In the findings, Execute, Verify, and Challenge are grouped as execution-oriented requests, while Explore and Synthesize are grouped as exploratory requests.

MindKicks interaction intensity

For each MindKicks card, I coded the highest observed level of participant engagement. The scale is ordinal: higher levels subsume lower levels.

- **Ignored:** The card appeared, but the participant did not visibly orient to it or interact with it.
- **Glanced:** The participant briefly looked at or hovered over the card but did not appear to read it in detail (under 3 seconds).
- **Read:** The participant opened, expanded, pinned, or otherwise spent enough time with the card to plausibly read the suggestion (more than 3 seconds).
- **CodeGen:** The participant requested code generation (pressed the card's "Try Me" button).
- **CodeAdded:** The participant added card-generated code into the notebook workflow.

The main quantitative findings report Glanced, Read, CodeGen, and CodeAdded counts and use the MK Interaction Score defined in Section 7.5.8.

Analytical reaction codes

For both coding agent and MindKicks events, I coded analytical reactions when the participant's verbalization or observable action responded to a prior AI output. Reactions were coded from the think-aloud transcript and video recording.

- **Surface:** A minimal acknowledgment, such as noticing that the AI produced output, without adding analytical content.
- **Observation:** A descriptive statement about a result, variable, plot, or suggestion without interpreting its implication for the dataset-fitness task.
- **Interpretation:** A participant explained what a result or suggestion meant for the analysis, the data, or the downstream reassessment use case.

- **Critical:** A participant questioned, challenged, qualified, or identified a limitation in an AI output, result, or emerging conclusion.
- **Synthesis:** A participant connected multiple findings, evidence sources, or task considerations into a broader judgment or memo-level conclusion.

The Analytical AI-Engagement Score (AIES) defined in Section 7.5.8 includes Interpretation, Critical, and Synthesis reactions. Surface and Observation reactions were retained in the codebook but excluded from AIES because they indicated lower analytical depth.

E.4 Coding Agent Request Events

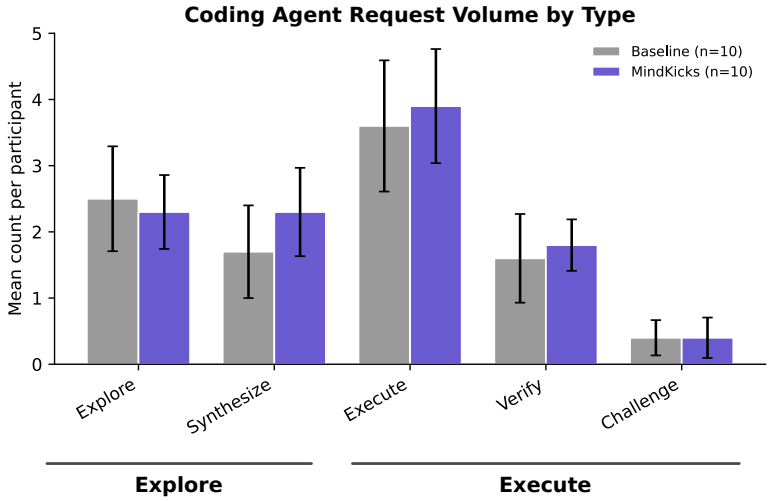


FIGURE E.1: Mean coding-agent request volume by request type and condition. Both groups used the coding agent in similar ways. Error bars show ±1 SEM.

TABLE E.4: Coding-agent request events from Chapter 7 process analysis. Request text is lightly cleaned for appendix readability: blank lines are removed, and long requests are shortened with ellipses while preserving the opening context and the participant's final ask where possible.

Session	Time (sec)	Coding Agent Request	Request Type
P03-MK	5.7	For starters, I want to visualize some of the important parameters for the task in hand (refer cell 1). What can be the potential data analysis techniques i can use	Explore
P03-MK	13.6	i further want to analyse which features are not relevant or might posses a risk based on the requiremnets we have	Explore
P03-MK	17.0	can you add all the analyses that you mentioned. it would be good to have some written conclusions too	Execute
P03-MK	28.3	Median Price Premium for properties WITH recorded renovations: -7.9% can you explain this further	Verify
P03-MK	32.6	To see the real effect of renovation, we should compare homes of similar ages.	Execute
P03-MK	36.5	In Cell 30 and 34, you calculate a 'Renovation Premium' by comparing sale prices to the … … can you consider this and edit	Execute
P03-MK	43.9	based on all the plots can you prepare an analysis for what paraters are important and what not. refer to cell 1 for the task	Synthesize

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P03-MK	48.8	what do you think were the most problematic and risky parameters and factors based on the analysis so far	Synthesize
P03-MK	58.3	any other features you think if we knew would have been helpful that can be collected	Verify
P06-MK	0.8	Can you read the descriptions of the task and give a full guidance to me if this dataset is good to use?	Explore
P06-MK	3.2	Can you analyze the feature correlations with sale price output and tell me what to do	Explore
P06-MK	9.8	can you evaluate the missing data and feature stability	Execute
P06-MK	11.7	Can you compare and summarize the output of missingness findings and feature correlations, then give me a brief summary	Synthesize
P06-MK	17.4	Can you read the ambient agents and give me suggestions that if i need to do those tasks, if it has overlap with your previous findinds please skip it.	Explore
P06-MK	20.4	Are the analysis I did enough for me to finish some of the section questions? if not, what should I do?	Explore
P06-MK	29.4	Now I have added some suggestions from the agents and please summarize the findings for … … answer some the sections questions. and also compare your previous findings to see if […]	Synthesize
P06-MK	31.9	what is Grade 7 and 1.1 variation,	Verify
P06-MK	33.9	Can you explain your findings to me in a more easy understand way? It seems like you used some grading criteria that is hard for me to understand?	Verify
P06-MK	35.4	I mean the summarization that can answer the section question part.	Verify
P06-MK	41.5	Your response has too many bullet points, can you make the summarise more like paragraphs	Execute
P06-MK	43.3	I think your output is too detailed like you include some numbers or prices, it is not general	Execute
P07-B	0.4	Tell me about this data sets, summarize the important columns	Explore
P07-B	2.0	In the dataset, is there any data/column that adhere to this or violate this This means … … characteristics (e.g., race, income) appear in the model, even as proxies.	Explore
P07-B	4.6	what could be serious data problems for this use case? and i can check this btw just let … … guide me step by step for data explotaion for the use case brief	Explore
P07-B	7.8	use pandas instead of polars	Execute
P07-B	14.3	for step 4, avg_basement_sqft shows some number, what do this suggest, and how can i start answering the memo	Explore
P07-B	17.4	# Check price signal from kitchen quality quality_impact_df = (… … for this one as well and guide me how can i answer the notebook memo based on this findings	Synthesize

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P07-B	20.8	Section 1: Critical Risks (The Crash) What you found: Look at the Market Trend plot. You … … what if its only $3 like for avg here	Challenge
P07-B	25.9	explain to me like im five SECTION 1 — CRITICAL RISKS Risk: Operational Blindness (Sale … … use this feature in production, it will significantly over-predict the value of homes…	Explore
P07-B	27.7	Median Price by Sale Condition: sale_condition AdjLand 110000.0 Abnorml 126500.0 Family … … 261329.0 Name: sale_price, dtype: float64 Price Gap (Normal vs Abnormal): $159,000	Verify
P07-B	30.3	okay slow down what other thigns can be consider for section 1 answer, like one thign and … … situations by $30k+, leading to unfairly high tax assessments for those homeowners.	Explore
P07-B	31.7	Number of homes with a 'Ghost' Basement: 16 Percentage of the whole dataset: 0.5% nope wrong	Challenge
P07-B	32.5	Homes with NO renovation recorded: 1569 Percentage of the dataset: 53.5%	Verify
P07-B	36.4	I truly think that sale condition is not strong enough, do something else? and lets do section 2 first	Synthesize
P07-B	37.5	Cheapest District Median: $88,250 Most Expensive District Median: $319,000 Location … … 24.846416 7 20.853242 8 14.027304 9 4.573379 10 1.501706 Name: proportion, dtype: float64	Verify
P07-B	40.0	### **SECTION 3 — REMAINING SIGNAL (\~100 words)** After setting aside variables you … … identify undervalued properties, or do the exclusions strip out too much? *Don't just […]	Synthesize
P07-B	41.8	After setting aside legally restricted variables (district, zoning_type), operationally … … is debatable, what is the proof for this code wise? i can give it to you to analyze	Explore
P07-B	44.1	sale_year 2006 159500.0 2007 165125.0 2008 161000.0 2009 160850.0 2010 155000.0 Name: … … its not bad idk like ?	Verify
P07-B	56.5	"what does is mean: Appraisal Appeal Threshold"	Verify
P07-B	57.7	why does this a problem? Median Price by Sale Condition: sale_condition AdjLand 110000.0 … … 261329.0 Name: sale_price, dtype: float64 Price Gap (Normal vs Abnormal): $159,000	Verify
P09-B	5.2	design a heat map of the correlation between each feature and sale price	Execute
P09-B	8.5	what is the variability over time of each feature?	Execute
P09-B	12.7	what was your threshold or metric for calculating stability vs hih drift risk for each numerical value?	Verify

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P09-B	31.7	using your previous metrics, analyze variability over time per month instead of per year. because we only have 2006-2010 data, an n of 5 to analyze variability over time may be skewed	Execute
P09-B	35.7	refactor this code to include all numerical features in the dataset, not just the 6 "key" features you selected - we need the full picture	Execute
P09-B	40.5	which features vary most significantly monthly (i.e. that would impact the efficacy of our later ML model), and what is your opinion on excluding them from the dataset?	Synthesize
P10-MK	7.8	give me the code to find the missing garage_capacity basement_sqft records	Execute
P10-MK	45.0	After setting aside variables you cannot use — legal exclusions, operationally … … identify undervalued properties, or do the exclusions strip out too much?	Synthesize
P10-MK	48.1	[1. AUDIT REDUNDANCY IN QUALITY GRADES - High 2. Silent row-dropping during modeling - … … count, percentage, distribution, or cross-tab) * Why it matters for *this* application	Synthesize
P10-MK	52.6	oth useful and problematic,* depending on how the model is built or deployed. - Quarterly timing v/s market seasonality	Verify
P10-MK	54.4	What is the single most important thing about the deployment context that you *cannot* answer from the data — but that would materially change your fitness verdict if you knew it	Synthesize
P11-MK	5.3	write a linear regression model code, using df dataset, sale_price ~ all other variables.	Execute
P11-MK	9.0	categorical_transformer = Pipeline(steps=[("imputer", SimpleImputer(strategy="constant", … … sparse_output=False))]) how does it handle missingness for categorical variables	Verify
P11-MK	12.7	help me to identify which variables cannot be used — due to legal constraints, operational unavailability, or data quality problems	Explore
P11-MK	22.2	build another linear regression model, remove sale year, sale month, sale condition, district. make missing basement_quality to be 0, and keep renovation year.	Execute
P11-MK	24.6	do the f test for the reduced model	Execute
P11-MK	26.0	you can install statsmodels	Execute
P11-MK	30.6	for the reduced model, do t test for each feature.	Execute
P11-MK	33.9	report the full table with all features	Execute
P11-MK	39.8	check how many samples(proportion) have the same renovation_year and construction year?	Execute
P11-MK	43.3	do the model diagnosis for the reduced model.	Execute

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P11-MK	47.0	What are the 2–4 (or more) most serious data problems for this use case? For each state: … … add more if i missed, 1. missingness. 2. outliers.	Synthesize
P11-MK	47.6	### **SECTION 1 — CRITICAL RISKS (\~150 words)** What are the 2–4 (or more) most serious … … add your answers here	Synthesize
P11-MK	48.9	### **SECTION 2 — TRADEOFFS (\~150 words)** Name at minimum 1–3 (or more) features that … … (e.g., predictive strength, proportion of affected rows). [Your tradeoffs here]	Synthesize
P11-MK	49.6	### **SECTION 3 — REMAINING SIGNAL (\~100 words)** After setting aside variables you … … identify undervalued properties, or do the exclusions strip out too much? *Don't just…	Synthesize
P11-MK	51.2	### **SECTION 4 — VERDICT-CHANGING UNKNOWN (\~75 words)** What is the single most … … answer from the data — but that would materially change your fitness verdict if you…	Synthesize
P11-MK	52.3	for section 4, how about the missingness of basement feature	Challenge
P11-MK	52.9	for section 4, how about the missingness of basement info	Challenge
P11-MK	53.3	### **SECTION 5 — FITNESS VERDICT (\~75 words)** **Ready / Conditionally Ready / Not … … at least one specific finding by name (a column, statistic, or pattern you observed).	Synthesize
P11-MK	55.4	i think its intersting that both construction year and renovatrion are significant because more than half of renovation year are the same as construction year. use this for section 5	Challenge
P11-MK	56.5	— ## FITNESS ASSESSMENT MEMO **To:** Project Lead **Re:** Riverdale County Reassessment … … concrete (e.g., predictive strength, proportion of affected rows). * **Renovation Year:**	Synthesize
P12-B	6.6	any na's in my dataset?	Execute
P12-B	7.0	write a script to check for na's	Execute
P12-B	12.9	build a correlation without these three variables, district variable and zoning_type basement quality	Execute
P12-B	26.3	list the EDA steps	Verify
P12-B	28.2	can you show the p values for the correlation matrix?	Execute
P12-B	33.8	how to identify undervalued properties?	Explore
P12-B	36.8	what literature is on how to identify undervalued properties?	Explore
P12-B	42.5	what are caveats?	Explore
P12-B	55.1	list numeric_df_2 varibles only	Execute
P13-B	9.4	create a histogram for frequency of all of the variables we have in df	Execute

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P13-B	14.5	Can you compare the districts to eaach other to see if there are any that are unrepresented by the data and would fall under a critical risk for creating a prediction	Execute
P13-B	23.1	Can we see if there are any features that are uncorrelated to sale price? Can we find the features that are most correlated to sale price?	Execute
P13-B	26.8	can you interpret the pearsons correlation coefficient.	Verify
P13-B	29.1	can you find the pearsons coef for the categorical variables too? since you didnt include them tjere	Execute
P13-B	34.5	are there any protected class characreristics present in our dataset that we wouldn't be able to use	Explore
P13-B	41.3	Can you identify features with large outliers or with values that would greatly impact a … … feature? Would those features be unsuitable for predicting sale price like in the prompt	Execute
P13-B	54.5	IS there a larg ecorrelation between sqf and sme of the other predictors?	Execute
P14-B	5.0	It looks like basement quality has 18.12% missing, can we add a section to colinearity and look variables we need to remove due to colinearity before we start looking at building models.	Execute
P14-B	11.8	okay, we need to come up with a clean database to analyze. WE need to remove … … remove the colinear variable and fill in the missing data in the the remaining variables.	Execute
P14-B	15.5	sale_price is the variable we want to build a model from the remaining cleanup variable, … … compare it to the true sale price. so for the next step, build a sale_price model,…	Execute
P14-B	19.9	As a Last step, can you find all properties that have a sale_price that is more than 10% … … underassesed properties. Put this extract in a new database for analysis and reporting.	Execute
P14-B	21.0	could you also write a second report, labeling those properties by sale year.	Execute
P14-B	23.7	could you looke at the orginal data and find the top 5 varaibles on predictive strength in a model that uses the cleaned up data to predict_sale_price.	Execute
P14-B	25.0	Since the top 4 are all Districts, the 5th is kitchen_quality, could you change that to the top 10 variables on Predictive strenght.	Execute
P14-B	26.7	Since the top 10 are all Districts or kitchenquality, could you change that to the top 20 variables on Predictive strenght.	Execute
P14-B	34.0	could you list the variables used, and a Strenght of Prediction measure in the model to predict projected_sale_price.	Execute
P16-B	0.0	Do you see any potential legal repercussions from using any of the features in this dataset? Be brief	Explore

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P16-B	7.2	In the dataset, what does the feature partial sale mean then or why would a partial sale take place? Brief	Verify
P16-B	15.2	Get rid of the features that we can't use. Namely the protected class predictors, such as … … or kitchen quality, because i'm not sure if these assessments are ongoing and up to date.	Execute
P16-B	19.4	Look, the construction grade feature seems less subjective than other quality-based … … wrong with the sale year and month, and the sale condition	Challenge
P16-B	27.0	What is the most explainable potential model for this task?	Explore
P16-B	29.1	Would smth like XGBoost not be considered explainable enough? Why not?	Challenge
P16-B	33.9	Use the cleaned dataset and make a multiple linear regression model, and compare it to an XGBoost model. Don't overdo it, make a basic predictive ability comparison.	Execute
P16-B	38.7	Make a multiple linear regression model, and include some stats about its performance as relevant for this task.	Execute
P16-B	40.9	Using the model performance data, fill in the fitness assessment memo, strictly following the guidelines.	Synthesize
P17-MK	0.8	ok summarize what we have talked about: what should we do, steps that we should focus on, and how I can best utilize this tools ine ach step	Explore
P17-MK	6.0	from the data dictionary, which feature is definitely risky or unuseful here?	Explore
P17-MK	7.8	summarize these to prep for the memo part (fit in the memo requirements)	Synthesize
P17-MK	12.1	from the heatmap are these data the only data that we should use and why?	Verify
P17-MK	14.0	from the heatmap, are these the only features taht we should've kept?	Verify
P17-MK	16.1	ok discard then.	Execute
P17-MK	22.2	currently The county-wide average grade is 6.23. Analysis shows 12 districts have an … … Should I or shoiuld I not include hte?	Verify
P17-MK	25.2	Ok so what are the current findings„ and what have we answered clearly, and what sections in the report need more investigation? IS teh data alreaady cleaned by now?	Synthesize
P17-MK	29.4	Final data contains only 2950 rows out of 12000? Is that too little data? Like I don't mind but I just want to jutify whether thisdataset should be used or not.	Verify
P17-MK	30.8	Can we look more into the oage of the records?	Explore
P17-MK	33.2	currently what are the stories for my memo for all sections?	Synthesize
P17-MK	49.8	ok what are the risky ones and waht are the remaining predictors for section 3. And lowkey I don't think that this is conditionally ready. Why so >	Challenge
P18-MK	11.2	give me code to determine correlation between basement_quality and sale price	Execute
P18-MK	16.9	give me a correlation of all the factors vs sale price.	Execute

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P18-MK	25.7	give me the distribution of all factors vs sale price having a correlation of more than 0.5	Execute
P18-MK	50.6	find features that are *both useful and problematic,* in this dataset. tell me how are you doing this and what are the shortcomings	Explore
P20-MK	0.9	What are features in the dataset that may be protected-class characteristics? Even as proxies?	Explore
P20-MK	2.8	Generate code to find out how many rows of data we have per feature	Execute
P20-MK	4.3	Can you think of scenarios where basement_quality could be a useful feature for this use case? Is it even useful	Explore
P20-MK	9.6	Filter out data with a sale_year older than 1990	Execute
P20-MK	11.7	Filter out data with missing basement_quality value	Execute
P20-MK	26.8	Explain the values in bsaement_quality feature	Execute
P20-MK	27.2	Do we have granular data on why basement_quality is missing?	Execute
P20-MK	29.4	Find features from cell 21. These must be avoided for legal reasons.	Execute
P20-MK	31.3	district (High Risk) zoning_type (Medium Risk) construction_year / renovation_year … … Exclude these features	Execute
P20-MK	32.0	Identify more biases in the data	Explore
P20-MK	34.4	Identify key missing data point	Execute
P20-MK	37.6	Apart from recency, find another critical missing information	Explore
P20-MK	40.2	Find another piece of critically important missing data	Explore
P20-MK	42.7	Is the data (after exclusions of features) enough for a correct model predicition	Verify
P20-MK	47.4	Any other critical missing data EXCEPT recency, bedroom_count?	Explore
P20-MK	48.8	Good reason why these 4 could also be valuable?	Verify
P21-MK	6.8	Go over this notebook, I have to fill out a memo and firstly I am trying to figure out … … to perform some more EDA to get a better idea of some of the critical risks over here.	Explore
P21-MK	17.0	I would like you to also go through some of the suggestions by ambient agents	Execute
P21-MK	23.1	so for section 2, I am thinking of one os features which is beoth useful and problematic … … give me 2 features for the section 2 in the memo	Verify
P21-MK	45.1	print out the variables which do not have legal exclusions, operationally unavailable … … what are some of the usable variables. we can run a quick analysis check to print them…	Execute

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P22-B	3.6	Summarize the asks and identify the key things I need to check for. What are the key … … What do you think are the possible issues I need to look for in each variable? What […]	Explore
P22-B	8.2	For each variable find if they are categorical or continuous. if categorical how many distrinct values are there. give a concise code block	Execute
P22-B	15.6	For each categorical variables calculate the average sales price for each category. For continuous variables, Plot the price vs variable for each of the variables.	Execute
P22-B	18.9	Create a correlation plot between the independent continuous variables to identify variables that are highly correlated	Execute
P22-B	22.7	Are there any variables that may have a legal challenge?	Explore
P22-B	26.5	Should we have separate models for different build type? will that remove the bias?	Explore
P22-B	30.4	I think we can remove district. We can keep rest of the variables as they are not a … … Which all of the remaining variables are useful in the model and the data will be …	Explore
P22-B	33.1	Are we only left with continuous variables? would none of the categorical variables be available for all buildings?	Verify
P22-B	35.2	Lets build a linear regression model with the usable variables to see if we can use these to make a reliable model to predict value of the property.	Execute
P22-B	38.1	What are the key variables? What are the top continuous variables? are all top variables categorical?	Execute
P22-B	39.6	What are the top variables in the model we used? We did not district rigth	Verify
P22-B	40.4	Analyize the regression model to see what are the useful variables	Execute
P22-B	46.2	Based on the analysis and the type of data what are the top critical risks? Few are: The … … List all risks.	Synthesize
P22-B	48.6	Update only section 1 of notebook with the exact risks	Synthesize
P22-B	51.6	What useful and problematic features? Include sales pos data, District, And Building type. Incldue that we can use building type since it may not be a proxy	Synthesize
P22-B	53.3	For section 3 just list all remaining categorical and continuos variables we used in the linear model. Give the usefullness of each variable in the linear model.	Synthesize
P22-B	54.3	In section also give context of the approach. What did we do to see if these signals are useful	Synthesize
P22-B	56.8	For section 4 what could be the biggest unknown?	Explore
P22-B	57.8	Any other unknown that can change verdict?	Explore
P22-B	59.0	Lets go with demographic overlay	Execute

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P25-B	5.5	Take a look at the Fitness Assessment Memo since that is what i need to ultimately … … Is my understanding wrong? What is your suggestion	Explore
P25-B	7.6	Yea lets first remove proxy features/race related columns that should not be used for training. Tell me your thought process while suggesting the column names	Explore
P25-B	9.7	Can we do it using pandas	Execute
P25-B	13.2	sale_condition, sale_month, sale_year (Operational Risk): The county wants to score all … … Can you explain this a little? Can we train a model on N features and then when […]	Verify
P25-B	17.2	Based on the few points that we just discussed. Can you help create a short draft for me to review for section 1	Synthesize
P25-B	22.9	What about sale price. Is that a feature we keep?	Verify
P25-B	26.9	Lets now progress to section 2 - tradeoffs. Same strategy, we will still go through the thought process before finalising the answer	Explore
P25-B	30.0	Can you run and evaluate the quantitive results shown above	Verify
P25-B	37.5	SECTION 4 — VERDICT-CHANGING UNKNOWN i didnt understand your explanation	Explore
P27-MK	9.5	USE CASE BRIEF — Riverdale County Property Reassessment Model Client goal: Riverdale … … You must assess whether what is left after removing problematic variables is genuinely […]	Explore
P27-MK	12.0	Yes we can start implementing	Execute
P27-MK	18.7	i think we need to dig deeper into the data to assess its quality for predictiveness.	Explore
P27-MK	21.7	analyse the output to understand what this implies for the data predictiveness	Verify
P27-MK	23.6	yes	Execute
P27-MK	34.2	lets look at what we have done till now and assess how prepared are we to write the memo	Synthesize
P27-MK	37.7	lets look at the temporal drift first	Execute
P27-MK	49.3	Here is some context from our previous chat: I've analyzed the statistical distribution … … answer from the data — but that would materially change	Synthesize
P27-MK	52.0	based on the curent cell outputs lets build the memo, follow the template	Synthesize
P28-B	1.5	SECTION 1 — CRITICAL RISKS (~150 words) What are the 2–4 (or more) most serious data … … CAN YOU ELABORATE THIS AND PROVIDE THE APPROPRIATE CODE TO FIND THE RESULTS	Explore
P28-B	6.9	What are the 2–4 (or more) most serious data problems for this use case? For each state: … … USING THE OUTPUTS I GOT, ANSWER THESE QUESTIONS IN 150 WORDS APPROX	Synthesize

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P28-B	13.8	### **SECTION 2 — TRADEOFFS (\~150 words)** Name at minimum 1–3 (or more) features that … … PROVIDE ME SUITABLE CODE TO ANSWER THESE QUESTIONS	Explore
P28-B	18.3	### SECTION 3 — REMAINING SIGNAL (~100 words) After setting aside variables you cannot … … ANSWER THESE QUESTIONS USING PREVIOUS CODE YOU PROVIDED AND IF NEEDED GIVE ME A…	Synthesize
P28-B	22.3	### SECTION 4 — VERDICT-CHANGING UNKNOWN (~75 words) What is the single most important … … ANSWER THESE QUESTIONS USING THE CODE YOU PROVIDED EARLIER THREE SECTIONS IN 75 WORDS…	Synthesize
P28-B	24.8	ANSWER THE QUESTIONS IN SECTIONS 4 WHICH IS GIVEN THAT THE WORD LIMIT IS APPROMIATELY 75 WORDS	Synthesize
P28-B	27.6	### SECTION 5 — FITNESS VERDICT (~75 words) Ready / Conditionally Ready / Not Ready … … ANSWER THE QUESTION IN SECTION 5	Synthesize
P29-MK	0.2	Explain the context. What should my model predict? fromt he cell 1	Explore
P29-MK	2.6	I have the data from 2006 to 2010 will i used the all data to train? or suppose I should use 2006 to 2009 and let the 2010 for the testing? is that?	Verify
P29-MK	9.7	Okay Fromt he missing counnt in cell 9 what i can see is only basement quality has large … … fit the model removing the mising count and without the basement quality first and […]	Execute
P29-MK	14.8	Which machine learning model did you use here?	Verify
P29-MK	15.7	use RF, XGboost and do the prediction and asses the undervalued task	Execute
P29-MK	17.7	Since Xgboost predicts well I will move forward with XGboost, I want to see the feature importance chart to know which features are import here for the prediction	Execute
P29-MK	20.0	Now based on the prediction for the year 2010 All I need to know the how many housed were soled undervalued in the year 2010 and how many were overvalued, do the analysis	Execute
P29-MK	22.3	Now I have few basis querry about the prediction you made. 1. Did you removed the entire row where the variable had missing value?	Verify
P29-MK	25.1	is the sell price was skewed? Do i need to do log transformation?	Execute
P29-MK	26.4	Okay I am done with the analysis, Now i need to write the fitness assement memo: lets go each sectio at a time ,	Synthesize
P29-MK	33.5	I dont think, so what i think is Varible realted to the location should be there to predict the price	Execute
P29-MK	37.7	what for section 2?	Explore
P29-MK	40.4	i think you are okay, write it in that secton	Synthesize
P29-MK	42.0	What did the section 3 mean?	Explore

Continued on next page

Table E.4: Coding-agent request events from Chapter 7 process analysis (continued).

Session	Time (s)	Coding Agent Request	Request Type
P29-MK	44.3	okay write the section3 for me	Synthesize
P29-MK	48.3	fix the sell 2 markdown why its now shwing the writing for the section 3 and next	Execute

E.5 MindKicks Sessions Timelines



FIGURE E.2: MindKicks condition timelines, clustered by MK Integration Pattern and annotated with CA Usage Style. Each row corresponds to a participant session, ordered according to the MK Integration Pattern groups in Table 7.5; participant labels indicate the corresponding CA style. Blue diamonds indicate MindKicks card interactions, blue circles indicate analytical reactions to MindKicks cards, light blue ticks indicate system-generated MindKicks cards/merges, orange squares indicate coding agent requests, and orange circles indicate reactions to coding agent responses. Outlined markers indicate lighter engagement events (e.g., glanced cards, surface/observation reactions, or Execute/Verify coding agent requests), while filled markers indicate deeper uptake or more analytically engaged events (e.g., Read/CodeGen/CodeAdded cards, Interpretation/Critical/Synthesis reactions, or Explore/Synthesize/Challenge coding agent requests).

E.6 Analytical AI-Engagement Scores (AIES)

TABLE E.5: Per-participant Analytical AI-Engagement Scores. *CA Analytical Reaction* columns count surface (Sur), observation (Obs), interpretation (Inter), critical (Crit), and synthesis (Synth) responses to CA outputs. *MK Analytical Reaction* columns count the same response classes for MindKicks cards. *AIES* columns sum interpretation, critical, and synthesis responses (the analytically engaged subset).

PID	Cond.	CA Analytical Reaction					AIES-CA	MK Analytical Reaction					AIES-MK	AIES-TOT
		Sur	Obs	Inter	Crit	Synth		Sur	Obs	Inter	Crit	Synth		
		Low		*High*				*Low*		*High*				
P07	B	0	10	10	1	0	11	0	0	0	0	0	0	11
P09	B	0	3	3	0	0	3	0	0	0	0	0	0	3
P12	B	1	2	5	0	1	6	0	0	0	0	0	0	6
P13	B	0	1	5	0	2	7	0	0	0	0	0	0	7
P14	B	0	6	3	0	0	3	0	0	0	0	0	0	3
P16	B	1	5	2	1	0	3	0	0	0	0	0	0	3
P22	B	0	7	10	2	0	12	0	0	0	0	0	0	12
P24	B	0	0	0	0	0	0	0	0	0	0	0	0	0
P25	B	0	1	7	1	0	8	0	0	0	0	0	0	8
P28	B	1	4	2	0	0	2	0	0	0	0	0	0	2
P03	MK	0	4	4	0	0	4	0	5	4	8	2	14	18
P06	MK	0	10	2	0	0	2	0	1	1	0	0	1	3
P10	MK	0	2	3	0	0	3	0	1	1	3	2	6	9
P11	MK	0	10	9	0	0	9	0	2	1	0	0	1	10
P17	MK	0	7	4	1	0	5	0	3	3	1	0	4	9
P18	MK	0	0	4	0	0	4	0	0	2	0	0	2	6
P20	MK	0	6	6	2	0	8	0	2	6	5	1	12	20
P21	MK	0	0	2	0	1	3	0	2	0	1	2	3	6
P27	MK	0	3	4	0	0	4	0	2	4	3	0	7	11
P29	MK	0	8	7	0	1	8	0	1	0	0	0	0	8
Mean (B)		*0.3*	*3.9*	*4.7*	*0.5*	*0.3*	*5.5*	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*	*0.0*	*5.5*
Mean (MK)		*0.0*	*5.0*	*4.5*	*0.3*	*0.2*	*5.0*	*0.0*	*1.9*	*2.2*	*2.1*	*0.7*	*5.0*	*10.0*

Bibliography

[1] Mark Abdelshiheed, John Wesley Hostetter, Tiffany Barnes, and Min Chi. 2023. Leveraging Deep Reinforcement Learning for Metacognitive Interventions Across Intelligent Tutoring Systems. In *Artificial Intelligence in Education*, Ning Wang, Genaro Rebolledo-Mendez, Noboru Matsuda, Olga C. Santos, and Vania Dimitrova (Eds.). Vol. 13916. Springer Nature Switzerland, Cham, 291–303. doi:10.1007/978-3-031-36272-9_24

[2] Piotr D. Adamczyk and Brian P. Bailey. 2004. If not now, when? the effects of interruption at different moments within task execution. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. Association for Computing Machinery, New York, NY, USA, 271–278. doi:10.1145/985692.985727

[3] Adobe. 2024. Adobe Firefly. https://www.adobe.com/products/firefly.html.

[4] Adobe. 2025. XD Platform. https://adobexdplatform.com/.

[5] Akshay Agrawal and Myles Scolnick. 2023. marimo: An Open-Source Reactive Notebook for Python. doi:10.5281/zenodo.12735329

[6] Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. 2025. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning. arXiv:2507.19457 [cs] doi:10.48550/arXiv.2507.19457

[7] Yongsu Ahn, Nam Wook Kim, and Benjamin Bach. 2026. Disrupting Cognitive Passivity: Rethinking AI-Assisted Data Literacy through Cognitive Alignment. arXiv:2604.02783 [cs.HC] doi:10.48550/arXiv.2604.02783

[8] Erfan Al-Hossami, Razvan Bunescu, Ryan Teehan, Laurel Powell, Khyati Mahajan, and Mohsen Dorodchi. 2023. Socratic Questioning of Novice Debuggers: A Benchmark Dataset and Preliminary Evaluations. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*. Association for Computational Linguistics, Toronto, Canada, 709–726. doi:10.18653/v1/2023.bea-1.57

[9] Grégoire Allaire, François Jouve, and Anca-Maria Toader. 2002. A Level-Set Method for Shape Optimization. *Comptes Rendus Mathematique* 334, 12 (Jan. 2002), 1125–1130. doi:10.1016/S1631-073X(02)02412-3

[10] J.E. Allen, Curry Guinn, and E. Horvtz. 1999. Mixed-initiative interaction. *IEEE Intelligent Systems* 14 (Oct. 1999), 14–23. doi:10.1109/5254.796083

[11] Zeyad Alshaikh, L. Tamang, and V. Rus. 2020. Experiments with a Socratic Intelligent Tutoring System for Source Code Understanding. In *Proceedings of the Thirty-Third International FLAIRS Conference*. AAAI Press, Palo Alto, CA, USA, 4 pages.

[12] Saleema Amershi, Dan Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. 2019. Guidelines for Human-AI Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow Scotland Uk, 1–13. doi:10.1145/3290605.3300233

[13] Lorin W. Anderson and David R. Krathwohl. 2001. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Longman, New York, NY, USA.

[14] Robert Rene Henri Anholt. 2006. *Dazzle 'em with Style: The Art of Oral Scientific Presentation* (2nd ed.). Elsevier Academic Press, Philadelphia.

[15] Anthropic. 2024. Introducing the next Generation of Claude. https://www.anthropic.com/news/claude-3-family.

[16] Ian Arawjo, Chelse Swoopes, Priyan Vaithilingam, Martin Wattenberg, and Elena L. Glassman. 2024. ChainForge: A Visual Toolkit for Prompt Engineering and LLM Hypothesis Testing. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–18. doi:10.1145/3613904.3642016

[17] Gantugs Atarsaikhan, Brian Kenji Iwana, and Seiichi Uchida. 2020. Guided Neural Style Transfer for Shape Stylization. *PLOS ONE* 15, 6 (June 2020), e0233489. doi:10.1371/journal.pone.0233489

[18] ATLAS.ti. 2024. ATLAS.Ti. https://atlasti.com.

[19] Marco Aurisicchio, Rob H Bracewell, Ken M Wallace, et al. 2007. Characterising Design Questions That Involve Reasoning. In *DS 42: Proceedings of ICED 2007, the 16th International Conference on Engineering Design, Paris, France, 28.-31.07. 2007*. The Design Society, Glasgow, United Kingdom, 11 pages.

[20] Autodesk. 2020. Fusion 360 Generative Design Extension. https://www.autodesk.com/products/fusion-360/generative-design-extension.

[21] Autodesk. 2021. Generative Design - Structural Load - Force per Entity? https://forums.autodesk.com/t5/fusion-360-design-validate/generative-design-structural-load-force-per-entity/td-p/10380547 Section: Fusion 360 Design, Validate & Document.

[22] Paul Ayres and John Sweller. 2005. The Split-Attention Principle in Multimedia Learning. In *The Cambridge Handbook of Multimedia Learning* (1 ed.), Richard Mayer (Ed.). Cambridge University Press, 135–146. doi:10.1017/CBO9780511816819.009

[23] Azrah Azhar, Erica L. Gralla, Connor Tobias, and Jeffrey W. Herrmann. 2016. Identification of Subproblems in Complex Design Problems: A Study of Facility Design. In *Volume 7: 28th International Conference on Design Theory and Methodology*. American Society of Mechanical Engineers, New York, NY, USA, Article V007T06A018, 9 pages. doi:10.1115/detc2016-60397

[24] Maral Babapour. 2015. Roles of Externalisation Activities in the Design Process. *Swedish Design Research Journal* 11 (May 2015), 34–46. doi:10.3384/svid.2000-964x.14134

[25] Lisanne Bainbridge. 1983. Ironies of Automation. *Automatica* 19, 6 (Nov. 1983), 775–779. doi:10.1016/0005-1098(83)90046-8

[26] Alex Baker and André van der Hoek. 2010. Ideas, Subjects, and Cycles as Lenses for Understanding the Software Design Process. *Design Studies* 31, 6 (Nov. 2010), 590–613. doi:10.1016/j.destud.2010.09.008

[27] Linden J. Ball and Bo T. Christensen. 2009. Analogical Reasoning and Mental Simulation in Design: Two Strategies Linked to Uncertainty Resolution. *Design Studies* 30, 2 (March 2009), 169–186. doi:10.1016/j.destud.2008.12.005

[28] Linden J. Ball and Bo T. Christensen. 2019. Advancing an Understanding of Design Cognition and Design Metacognition: Progress and Prospects. *Design Studies* 65 (Nov. 2019), 35–59. doi:10.1016/j.destud.2019.10.003

[29] Maria Bannert, Peter Reimann, and Christoph Sonnenberg. 2014. Process Mining Techniques for Analysing Patterns and Strategies in Students' Self-Regulated Learning. *Metacognition and Learning* 9, 2 (Aug. 2014), 161–185. doi:10.1007/s11409-013-9107-6

[30] Gagan Bansal, Besmira Nushi, Ece Kamar, Walter S. Lasecki, Daniel S. Weld, and Eric Horvitz. 2019. Beyond Accuracy: The Role of Mental Models in Human-AI Team Performance. *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing* 7, 1 (Oct. 2019), 2–11. https://ojs.aaai.org/index.php/HCOMP/article/view/5285

[31] Gagan Bansal, Besmira Nushi, Ece Kamar, Daniel S. Weld, Walter S. Lasecki, and Eric Horvitz. 2019. Updates in Human-AI Teams: Understanding and Addressing the Performance/Compatibility Tradeoff. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (July 2019), 2429–2437. doi:10.1609/aaai.v33i01.33012429

[32] Shraddha Barke, Michael B. James, and Nadia Polikarpova. 2023. Grounded Copilot: How Programmers Interact with Code-Generating Models. *Proceedings of the ACM on Programming Languages* 7, OOPSLA1 (April 2023), 78:85–78:111. doi:10.1145/3586030

[33] Eric P.S. Baumer and Bill Tomlinson. 2011. Comparing Activity Theory with Distributed Cognition for Video Analysis: Beyond "Kicking the Tires". In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 133–142. doi:10.1145/1978942.1978962

[34] Warren Berger. 2014. *A More Beautiful Question: The Power of Inquiry to Spark Breakthrough Ideas.* Bloomsbury USA, New York, NY.

[35] Dianne C. Berry and Donald E. Broadbent. 1984. On the Relationship between Task Performance and Associated Verbalizable Knowledge. *The Quarterly Journal of Experimental Psychology Section A* 36, 2 (May 1984), 209–231. doi:10.1080/14640748408402156

[36] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. 2023. Improving Image Generation with Better Captions. OpenAI technical report, 19 pages. https://cdn.openai.com/papers/dall-e-3.pdf

[37] BlockNote. 2024. BlockNote - Javascript Block-Based React Rich Text Editor. https://www.blocknotejs.org.

[38] Benjamin S. Bloom, Max D. Engelhart, Edward J. Furst, Walker H. Hill, and David R. Krathwohl. 1956. *Taxonomy of Educational Objectives: The Classification of Educational Goals. Handbook I: Cognitive Domain.* Longmans, Green, New York, NY, USA.

[39] Kirsten Boehner, Janet Vertesi, Phoebe Sengers, and Paul Dourish. 2007. How HCI Interprets the Probes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07).* Association for Computing Machinery, New York, NY, USA, 1077–1086. doi:10.1145/1240624.1240789

[40] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak

Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2022. On the Opportunities and Risks of Foundation Models. arXiv:2108.07258 [cs] doi:10.48550/arXiv.2108.07258

[41] Conrad Borchers, Jiayi Zhang, Ryan S. Baker, and Vincent Aleven. 2024. Using Think-Aloud Data to Understand Relations between Self-Regulation Cycle Characteristics and Student Performance in Intelligent Tutoring Systems. In *Proceedings of the 14th Learning Analytics and Knowledge Conference*. 529–539. arXiv:2312.05675 [cs] doi:10.1145/3636555.3636911

[42] Jeremy Boy, Louis Eveillard, Françoise Detienne, and Jean-Daniel Fekete. 2015. Suggested interactivity: Seeking perceived affordances for information visualization. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 639–648.

[43] Stephen Brade, Bryan Wang, Mauricio Sousa, Sageev Oore, and Tovi Grossman. 2023. Promptify: Text-to-Image Generation through Interactive Prompt Exploration with Large Language Models. arXiv:2304.09337 [cs]

[44] Virginia Braun and Victoria Clarke. 2019. Reflecting on Reflexive Thematic Analysis. *Qualitative Research in Sport, Exercise and Health* 11, 4 (Aug. 2019), 589–597. doi:10.1080/2159676X.2019.1628806

[45] Margot Brereton. 2004. Distributed Cognition in Engineering Design: Negotiating between Abstr Act and Material Representations. In *Design Representation*. Springer London, 83–103. doi:10.1007/978-1-85233-863-3_4

[46] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan,

Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models Are Few-Shot Learners. https://arxiv.org/abs/2005.14165v4.

[47] Zana Buçinca, Maja Barbara Malaya, and Krzysztof Z. Gajos. 2021. To Trust or to Think: Cognitive Forcing Functions Can Reduce Overreliance on AI in AI-assisted Decision-making. *Proc. ACM Hum.-Comput. Interact.* 5, CSCW1 (April 2021), 188:1–188:21. doi:10.1145/3449287

[48] Sahan Bulathwela, Hamze Muse, and Emine Yilmaz. 2023. Scalable Educational Question Generation with Pre-trained Language Models. In *Artificial Intelligence in Education*, Ning Wang, Genaro Rebolledo-Mendez, Noboru Matsuda, Olga C. Santos, and Vania Dimitrova (Eds.). Vol. 13916. Springer Nature Switzerland, Cham, 327–339. doi:10.1007/978-3-031-36272-9_27

[49] Jean-Marie Burkhardt, Françoise Détienne, Anne-Marie Hébert, Laurence Perron, Stéphane safin, and Pierre Leclercq. 2009. An Approach to Assess the Quality of Collaboration in Technology-Mediated Design Situations. In *European Conference on Cognitive Ergonomics: Designing beyond the Product — Understanding Activity and User Experience in Ubiquitous Environments (ECCE '09)*. VTT Technical Research Centre of Finland, FI-02044 VTT, FIN, Article 30, 9 pages. event-place: Helsinki, Finland.

[50] Bill Buxton. 2007. *Sketching User Experiences: Getting the Design Right and the Right Design.* Morgan Kaufmann, Burlington. doi:10.1016/B978-0-12-374037-3.X5043-3

[51] Bill Buxton. 2011. *Sketching User Experiences: Getting the Design Right and the Right Design.* Morgan Kaufmann, Amsterdam.

[52] Çetin Tünger, Çetin Tünger, Şule Taşlı Pektaş, and Sule Tasli Pektas. 2020. A Comparison of the Cognitive Actions of Designers in Geometry-Based and Parametric Design Environments. *Open House International* 45 (June 2020), 87–101. doi:10.1108/ohi-04-2020-0008

[53] Jj Cadiz, Gina Venolia, Gavin Jancke, and Anoop Gupta. 2001. *Sideshow: Providing Peripheral Awareness of Important Information.* Technical Report MSR-TR-2001-83. Microsoft Research, Redmond, WA, USA. https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr-2001-83.pdf

[54] Carrie J. Cai, Samantha Winter, David Steiner, Lauren Wilcox, and Michael Terry. 2019. "Hello AI": Uncovering the Onboarding Needs of Medical Practitioners for Human-AI Collaborative Decision-Making. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (Nov. 2019), 1–24. doi:10.1145/3359206

[55] Wanling Cai and Li Chen. 2020. Predicting User Intents and Satisfaction with Dialogue-based Conversational Recommendations. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '20)*. Association for Computing Machinery, New York, NY, USA, 33–42. doi:10.1145/3340631.3394856

[56] Carlos Cardoso, Ozgur Eris, Petra Badke-Schaub, and Marco Aurisicchio. 2014. Question Asking in Design Reviews: How Does Inquiry Facilitate the Learning Interaction?. In *Proceedings of the 10th Design Thinking Research Symposium (DTRS)*. Purdue University, West Lafayette, IN, USA, 1–18. https://docs.lib.purdue.edu/dtrs/2014/Impact/1/

[57] Juan C. Castro-Alonso and John Sweller. 2020. The Modality Effect of Cognitive Load Theory. In *Advances in Human Factors in Training, Education, and Learning Sciences*, Waldemar Karwowski, Tareq Ahram, and Salman Nazir (Eds.). Springer International Publishing, Cham, 75–84.

[58] Carlos Rafael Catalan, Lheane Marie Dizon, Patricia Nicole Monderin, and Emily Kuang. 2026. "I'm Not Reading All of That": Understanding Software Engineers' Level of Cognitive Engagement with Agentic Coding Assistants. arXiv:2603.14225 [cs.HC] doi:10.48550/arXiv.2603.14225

[59] Arjun Chandrasekaran, Deshraj Yadav, Prithvijit Chattopadhyay, Viraj Prabhu, and Devi Parikh. 2017. It Takes Two to Tango: Towards Theory of AI's Mind. http://arxiv.org/abs/1704.00717 arXiv:1704.00717 [cs].

[60] Catherine C. Chase, Jenna Marks, Laura J. Malkiewich, and Helena Connolly. 2019. How Teacher Talk Guidance during Invention Activities Shapes Students'

Cognitive Engagement and Transfer. *International Journal of STEM Education* 6, 1 (Dec. 2019), 14. doi:10.1186/s40594-019-0170-7

[61] Rimika Chaudhury, Philip J. Guo, and Parmit K. Chilana. 2022. "There's no way to keep up!": Diverse Motivations and Challenges Faced by Informal Learners of ML. In *2022 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 1–11. doi:10.1109/VL/HCC53370.2022.9833100 ISSN: 1943-6106.

[62] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating Large Language Models Trained on Code. arXiv:2107.03374 [cs]

[63] Michelene T. H. Chi. 2006. Laboratory Methods for Assessing Experts' and Novices' Knowledge. In *The Cambridge Handbook of Expertise and Expert Performance*, K. Anders Ericsson, Neil Charness, Paul J. Feltovich, and Robert R. Hoffman (Eds.). Cambridge University Press, Cambridge, 167–184. doi:10.1017/CBO9780511816796.010

[64] Parmit K. Chilana, Nathaniel Hudson, Srinjita Bhaduri, Prashant Shashikumar, and Shaun K. Kane. 2018. Supporting Remote Real-Time Expert Help: Opportunities and Challenges for Novice 3D Modelers. In *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, Piscataway, NJ, USA, 157–166. doi:10.1109/vlhcc.2018.8506568

[65] Leonardo Christino, Taylor Hill, and Fernando Paulovich. 2022. KnowledgeDecks: Automatically Generating Presentation Slide Decks of Visual Analytics Knowledge Discovery Applications. arXiv:2212.01469 [cs] doi:10.48550/arXiv.2212.01469

[66] John Joon Young Chung and Eytan Adar. 2023. PromptPaint: Steering Text-to-Image Generation Through Paint Medium-like Interactions. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–17. doi:10.1145/3586183.3606777

[67] John Joon Young Chung, Wooseok Kim, Kang Min Yoo, Hwaran Lee, Eytan Adar, and Minsuk Chang. 2022. TaleBrush: Sketching Stories with Generative Pre-trained Language Models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3491102.3501819

[68] Herbert H. Clark and Susan E. Brennan. 1991. Grounding in communication. In *Perspectives on socially shared cognition.*, Lauren B. Resnick, John M. Levine, and Stephanie D. Teasley (Eds.). American Psychological Association, Washington, 127–149. doi:10.1037/10096-006

[69] Carlos Coimbra Cardoso, Petra Badke-Schaub, and Ozgur Eris. 2016. Inflection Moments in Design Discourse: How Questions Drive Problem Framing during Idea Generation. *Design Studies* 46 (Sept. 2016), 59–78. doi:10.1016/j.destud.2016.07.002

[70] Comfy-Org. 2025. ComfyUI. GitHub repository. https://github.com/Comfy-Org/ComfyUI

[71] Alan Cooper, Robert Reimann, Dave Cronin, and Alan Cooper. 2014. *About Face: The Essentials of Interaction Design* (fourth edition ed.). John Wiley and Sons, Indianapolis, IN.

[72] Carlos G. Correa, Mark K. Ho, Fred Callaway, and Thomas L. Griffiths. 2020. Resource-Rational Task Decomposition to Minimize Planning Costs. arXiv:2007.13862 [cs] doi:10.48550/arXiv.2007.13862

[73] Maria João Costa, Hugo Amaro, and Hugo Gonçalo Oliveira. 2023. SmartEDU: Accelerating Slide Deck Production with Natural Language Processing. In *Lecture Notes in Computer Science*. Springer Nature Switzerland, 109–123.

[74] Scotty D. Craig, Jeremiah Sullins, Amy Witherspoon, and Barry Gholson. 2006. The Deep-Level-Reasoning-Question Effect: The Role of Dialogue and Deep-Level-Reasoning Questions During Vicarious Learning. *Cognition and Instruction* 24, 4 (2006), 565–591. arXiv:27739846

[75] Nigel Cross. 2006. *Designerly Ways of Knowing*. Springer, Berlin London.

[76] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz Studies — Why and How. *Knowledge-Based Systems* 6, 4 (Dec. 1993), 258–266. doi:10.1016/0950-7051(93)90017-N

[77] Valdemar Danry, Pat Pataranutaporn, Yaoli Mao, and Pattie Maes. 2023. Don't Just Tell Me, Ask Me: AI Systems That Intelligently Frame Explanations as Questions Improve Human Logical Discernment Accuracy over Causal AI Explanations. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–13. doi:10.1145/3544548.3580672

[78] Dassault Systèmes. 2021. Solidworks | Enhance Your Design Workflow with Generative Design. https://www.solidworks.com/media/enhance-your-design-workflow-generative-design.

[79] Nicholas Davis, Chih-Pin Hsiao, Yanna Popova, and Brian Magerko. 2015. An Enactive Model of Creativity for Computational Collaboration and Co-creation. In *Creativity in the Digital Age*, Nelson Zagalo and Pedro Branco (Eds.). Springer London, London, 109–133. doi:10.1007/978-1-4471-6681-8_7

[80] Nicholas Davis, Chih-PIn Hsiao, Kunwar Yashraj Singh, Lisa Li, Sanat Moningi, and Brian Magerko. 2015. Drawing Apprentice: An Enactive Co-Creative Agent for Artistic Collaboration. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*. ACM, Glasgow United Kingdom, 185–186. doi:10.1145/2757226.2764555

[81] Dean De Cock. 2011. Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. *Journal of Statistics Education* 19, 3 (2011), 1–14. doi:10.1080/10691898.2011.11889627

[82] Leslie A. DeChurch and Jessica R. Mesmer-Magnus. 2010. Measuring shared team mental models: A meta-analysis. *Group Dynamics: Theory, Research, and Practice* 14, 1 (2010), 1–14. doi:10.1037/a0017455

[83] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference On*. IEEE, 248–255.

[84] Michael Desmond and Michelle Brachman. 2024. Exploring Prompt Engineering Practices in the Enterprise. arXiv:2403.08950 [cs] doi:10.48550/arXiv.2403.08950

[85] John Dewey. 1933. *How We Think: A Restatement of the Relation of Reflective Thinking to the Educative Process*. D.C. Heath and Company.

[86] J. T. Dillon. 1984. The Classification of Research Questions. *Review of Educational Research* 54, 3 (Sept. 1984), 327–361. doi:10.3102/00346543054003327

[87] Alan Dix and Layda Gongora. 2011. Externalisation and Design. In *Procedings of the Second Conference on Creativity and Innovation in De Sign*. ACM, 31–42. doi:10.1145/2079216.2079220

[88] Kemal Mert Dogan, Hiromasa Suzuki, Erkan Gunpinar, and Myung-Soo Kim. 2019. A Generative Sampling System for Profile Designs with Shape Constraints and User Evaluation. *Computer-Aided Design* 111 (June 2019), 93–112. doi:10.1016/j.cad.2019.02.002

[89] Kees Dorst and Nigel Cross. 2001. Creativity in the Design Process: Co-Evolution of Problem–Solution. *Design Studies* 22, 5 (Sept. 2001), 425–437. doi:10.1016/S0142-694X(01)00009-6

[90] Kees Dorst and Judith Dijkhuis. 1995. Comparing Paradigms for Describing Design Activity. *Design Studies* 16, 2 (April 1995), 261–274. doi:10.1016/0142-694X(94)00012-3

[91] Tomas Dorta, Annemarie Lesage, Edgar Pérez, and J. Bastien. 2011. Signs of Collaborative Ideation and the Hybrid Ideation Space. In *Design Creativity 2010*. Springer London, London, United Kingdom, 199–206. doi:10.1007/978-0-85729-224-7_26

[92] Graham Dove, Kim Halskov, Jodi Forlizzi, and John Zimmerman. 2017. UX Design Innovation: Challenges for Working with Machine Learning as a Design Material. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, Denver Colorado USA, 278–288. doi:10.1145/3025453.3025739

[93] Graham Dove, Nicolai Brodersen Hansen, and Kim Halskov. 2016. An Argument For Design Space Reflection. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction (NordiCHI '16)*. Association for Computing Machinery, New York, NY, USA, 1–10. doi:10.1145/2971485.2971528

[94] Steven Dow, Julie Fortuna, Dan Schwartz, Beth Altringer, Daniel Schwartz, and Scott Klemmer. 2011. Prototyping dynamics: sharing multiple designs improves exploration, group rapport, and results. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2807–2816. doi:10.1145/1978942.1979359

[95] Steven P. Dow, Alana Glassco, Jonathan Kass, Melissa Schwarz, Daniel L. Schwartz, and Scott R. Klemmer. 2010. Parallel Prototyping Leads to Better Design Results, More Divergence, and Increased Self-Efficacy. *ACM Transactions on Computer-Human Interaction* 17, 4 (Dec. 2010), 1–24. doi:10.1145/1879831.1879836

[96] Ian Drosos, Advait Sarkar, Xiaotong, Xu, and Neil Toronto. 2025. "It Makes You Think": Provocations Help Restore Critical Thinking to AI-Assisted Knowledge Work. arXiv:2501.17247 [cs.HC] doi:10.48550/arXiv.2501.17247

[97] Ian Drosos, Advait Sarkar, Xiaotong Xu, Carina Negreanu, Sean Rintel, and Lev Tankelevitch. 2024. "It's like a Rubber Duck That Talks Back": Understanding Generative AI-Assisted Data Analysis Workflows through a Participatory Prompting Study. In *Proceedings of the 3rd Annual Meeting of the Symposium on Human-Computer Interaction for Work (CHIWORK '24)*. Association for Computing Machinery, New York, NY, USA, 1–21. doi:10.1145/3663384.3663389

[98] Steven M. Drucker, Georg Petschnigg, and Maneesh Agrawala. 2006. Comparing and Managing Multiple Versions of Slide Presentations. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. ACM, Montreux Switzerland, 47–56. doi:10.1145/1166253.1166263

[99] Aline Duelen, Iris Jennes, and Wendy Van den Broeck. 2024. Socratic AI Against Disinformation: Improving Critical Thinking to Recognize Disinformation Using Socratic AI. In *Proceedings of the 2024 ACM International Conference on Interactive Media Experiences (IMX '24)*. Association for Computing Machinery, New York, NY, USA, 375–381. doi:10.1145/3639701.3663640

[100] Jane L. E, Yu-Chun Grace Yen, Isabelle Yan Pan, Grace Lin, Mingyi Li, Hyoungwook Jin, Mengyi Chen, Haijun Xia, and Steven P. Dow. 2024. When to Give Feedback: Exploring Tradeoffs in the Timing of Design Feedback. In *Proceedings of the 16th Conference on Creativity & Cognition (C&C '24)*. Association for Computing Machinery, New York, NY, USA, 292–310. doi:10.1145/3635636.3656183

[101] Darren Edge, Joan Savage, and Koji Yatani. 2013. HyperSlides: Dynamic Presentation Prototyping. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Paris France, 671–680. doi:10.1145/2470654.2470749

[102] Linda Elder and Richard Paul. 2016. *The Thinker's Guide to The Art of Socratic Questioning*. Foundation for Critical Thinking Press.

[103] David Engel, Anita Woolley, Lisa Jing, Christopher Chabris, and Thomas Malone. 2014. Reading the Mind in the Eyes or Reading between the Lines? Theory of Mind Predicts Collective Intelligence Equally Well Online and Face-To-Face. *PloS one* 9 (Dec. 2014), e115212. doi:10.1371/journal.pone.0115212

[104] Douglas C. Engelbart. 1962. *Augmenting Human Intellect: A Conceptual Framework*. Summary Report AFOSR-3223. Stanford Research Institute, Menlo Park, CA. https://dougengelbart.org/pubs/papers/scanned-original/1962-augment-3906-Augmenting-Human-Intellect-a-Conceptual-Framework.pdf

[105] Ozgur Eris. 2003. Asking generative design questions: a fundamental cognitive mechanism in design thinking. In *DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design*. The Design Society, Stockholm, Sweden, 587–588.

[106] Ozgur Eris. 2004. *Effective Inquiry for Innovative Engineering Design*. Springer US, Boston, MA. doi:10.1007/978-1-4419-8943-7

[107] Lucile Favero, Juan Antonio Pérez-Ortiz, Tanja Käser, and Nuria Oliver. 2025. Enhancing Critical Thinking in Education by Means of a Socratic Chatbot. In *AI in Education and Educational Research*, Francisco Bellas and Oscar Fontenla-Romero (Eds.). Springer Nature Switzerland, Cham, 17–32. doi:10.1007/978-3-031-93409-4_2

[108] K. J. Kevin Feng and David W. Mcdonald. 2023. Addressing UX Practitioners' Challenges in Designing ML Applications: An Interactive Machine Learning Approach. In *Proceedings of the 28th International Conference on Intelligent User Interfaces (IUI '23)*. Association for Computing Machinery, New York, NY, USA, 337–352. doi:10.1145/3581641.3584064

[109] Daniela Fernandes, Steeven Villa, Salla Nicholls, Otso Haavisto, Daniel Buschek, Albrecht Schmidt, Thomas Kosch, Chenxinran Shen, and Robin Welsch. 2025. AI Makes You Smarter but None the Wiser: The Disconnect between Performance and Metacognition. *Computers in Human Behavior* 175 (2025), 108779. doi:10.1016/j.chb.2025.108779

[110] Jennifer Fernquist, Tovi Grossman, and George Fitzmaurice. 2011. Sketch-Sketch Revolution: An Engaging Tutorial System for Guided Sketching and Application Learning. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 373–382. doi:10.1145/2047196.2047245 event-place: Santa Barbara, California, USA.

[111] Figma. 2025. Figma: The Collaborative Interface Design Tool. https://www.figma.com/.

[112] Leah Findlater and Joanna McGrenere. 2004. A comparison of static, adaptive, and adaptable menus. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. Association for Computing Machinery, New York, NY, USA, 89–96. doi:10.1145/985692.985704

[113] Leah Findlater, Karyn Moffatt, Joanna McGrenere, and Jessica Dawson. 2009. Ephemeral adaptation: the use of gradual onset to improve menu selection performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1655–1664. doi:10.1145/1518701.1518956

[114] Stephen M. Fiore and Travis J. Wiltshire. 2016. Technology as Teammate: Examining the Role of External Cognition in Support of Team Cognitive Processes. *Frontiers in Psychology* 7 (Oct. 2016), 1531. doi:10.3389/fpsyg.2016.01531

[115] John H. Flavell. 1979. Metacognition and Cognitive Monitoring: A New Area of Cognitive–Developmental Inquiry. *American Psychologist* 34, 10 (Oct. 1979), 906–911. doi:10.1037/0003-066X.34.10.906

[116] React Flow. 2024. React Flow. https://reactflow.dev/.

[117] Kate Forbes-Riley and Diane J. Litman. 2009. Adapting to Student Uncertainty Improves Tutoring Dialogues. In *Artificial Intelligence in Education*. IOS Press, Amsterdam, Netherlands, 33–40. doi:10.3233/978-1-60750-028-5-33 MAG ID: 1568422853.

[118] FormidableLabs. 2024. Spectacle. GitHub repository. https://github.com/FormidableLabs/spectacle

[119] formlabs. 2020. Generative Design 101. https://formlabs.com/blog/generative-design/.

[120] Tsu-Jui Fu, William Yang Wang, Daniel McDuff, and Yale Song. 2022. DOC2PPT: Automatic Presentation Slides Generation from Scientific Documents. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 1 (June 2022), 634–642.

[121] Kazuma Fukumura and Takayuki Ito. 2025. Can LLM-Powered Multi-Agent Systems Augment Human Creativity? Evidence from Brainstorming Tasks. In *Proceedings of the ACM Collective Intelligence Conference (CI '25)*. Association for Computing Machinery, New York, NY, USA, 20–29. doi:10.1145/3715928.3737479

[122] Krzysztof Z. Gajos, Mary Czerwinski, Desney S. Tan, and Daniel S. Weld. 2006. Exploring the design space for adaptive graphical user interfaces. In *Proceedings of the working conference on Advanced visual interfaces (AVI '06)*. Association for Computing Machinery, New York, NY, USA, 201–208. doi:10.1145/1133265.1133306

[123] Krzysztof Z. Gajos and Lena Mamykina. 2022. Do People Engage Cognitively with AI? Impact of AI Assistance on Incidental Learning. In *Proceedings of the 27th International Conference on Intelligent User Interfaces (IUI '22)*. Association for Computing Machinery, New York, NY, USA, 794–806. doi:10.1145/3490099.3511138

[124] Steve Garner and Deana McDonagh-Philp. 2001. Problem Interpretation and Resolution via Visual Stimuli: The Use of 'Mood Boards' in Design Education. *Journal of Art & Design Education* 20, 1 (Feb. 2001), 57–64. doi:10.1111/1468-5949.00250

[125] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2016. Image Style Transfer Using Convolutional Neural Networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Las Vegas, NV, USA, 2414–2423. doi:10.1109/CVPR.2016.265

[126] William Gaver. 2012. What Should We Expect from Research through Design?. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Austin Texas USA, 937–946. doi:10.1145/2207676.2208538

[127] Michael Gerlich. 2025. AI Tools in Society: Impacts on Cognitive Offloading and the Future of Critical Thinking. *Societies* 15, 1 (Jan. 2025), 6. doi:10.3390/soc15010006

[128] Pratik Ghosh and Sean Rintel. 2025. YES AND: A Generative AI Multi-Agent Framework for Enhancing Diversity of Thought in Individual Ideation for Problem-Solving Through Confidence-Based Agent Turn-Taking. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3706599.3720142

[129] Frederic Gmeiner, Kaitao Luo, Ye Wang, Kenneth Holstein, and Nikolas Martelaro. 2025. Exploring the Potential of Metacognitive Support Agents for Human-AI Co-Creation. In *Proceedings of the 2025 ACM Designing Interactive Systems Conference (DIS '25)*. Association for Computing Machinery, New York, NY, USA, 1244–1269. doi:10.1145/3715336.3735785

[130] Frederic Gmeiner, Humphrey Yang, Lining Yao, Kenneth Holstein, and Nikolas Martelaro. 2023. Exploring Challenges and Opportunities to Support Designers in Learning to Co-create with AI-based Manufacturing Design Tools. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–20. doi:10.1145/3544548.3580999

[131] Gabriela Goldschmidt, Hagay Hochman, and Itay Dafni. 2010. The Design Studio "Crit": Teacher–Student Communication. *Artificial Intelligence for Engineering*

Design, Analysis and Manufacturing 24, 3 (Aug. 2010), 285–302. doi:10.1017/S089006041000020X

[132] Elizabeth Goodman, Erik Stolterman, and Ron Wakkary. 2011. Understanding Interaction Design Practices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1061–1070. doi:10.1145/1978942.1979100

[133] Google. 2025. Stax - The Complete Toolkit for AI Evaluation. https://stax.withgoogle.com/landing.

[134] Andrew D. Gordon, Carina Negreanu, José Cambronero, Rasika Chakravarthy, Ian Drosos, Hao Fang, Bhaskar Mitra, Hannah Richardson, Advait Sarkar, Stephanie Simmons, Jack Williams, and Ben Zorn. 2023. Co-Audit: Tools to Help Humans Double-Check AI-generated Content. arXiv:2310.01297 [cs] doi:10.48550/arXiv.2310.01297

[135] Arthur C. Graesser and Natalie K. Person. 1994. Question Asking During Tutoring. *American Educational Research Journal* 31, 1 (March 1994), 104–137. doi:10.3102/00028312031001104

[136] Ken Gu, Madeleine Grunde-McLaughlin, Andrew M. McNutt, Jeffrey Heer, and Tim Althoff. 2023. How Do Data Analysts Respond to AI Assistance? A Wizard-of-Oz Study.

[137] Ken Gu, Ruoxi Shang, Tim Althoff, Chenglong Wang, and Steven M. Drucker. 2024. How Do Analysts Understand and Verify AI-Assisted Data Analyses? arXiv:2309.10947 [cs.HC] doi:10.48550/arXiv.2309.10947

[138] Ning Gu and Peiman Amini Behbahani. 2018. Shape Grammars: A Key Generative Design Algorithm. In *Handbook of the Mathematics of the Arts and Sciences*, Bharath Sriraman (Ed.). Springer International Publishing, Cham, 1–21. doi:10.1007/978-3-319-70658-0_7-1

[139] Erkan Gunpinar, Umut Can Coskun, Mustafa Ozsipahi, and Serkan Gunpinar. 2019. A Generative Design and Drag Coefficient Prediction System for Sedan Car Side Silhouettes Based on Computational Fluid Dynamics. *Computer-Aided Design* 111 (June 2019), 65–79. doi:10.1016/j.cad.2019.02.003

[140] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. 2020. Deep Learning for 3D Point Clouds: A Survey. arXiv:1912.12033

[141] Douglas J. Hacker (Ed.). 1998. *Metacognition in Educational Theory and Practice*. Erlbaum, Mahwah, NJ.

[142] Roger R Hall. 2001. Prototyping for Usability of New Technology. *International Journal of Human-Computer Studies* 55, 4 (Oct. 2001), 485–501. doi:10.1006/ijhc.2001.0478

[143] Hari Subramonyam, Divy Thakkar, Jurgen Dieber, and Anoop Sinha. 2024. Content-Centric Prototyping of Generative AI Applications: Emerging Approaches and Challenges in Collaborative Software Teams. arXiv preprint arXiv:2402.17721. arXiv:2402.17721 [cs.HC] doi:10.48550/arxiv.2402.17721

[144] Galen Harrison, Kevin Bryson, Ahmad Emmanuel Balla Bamba, Luca Dovichi, Aleksander Herrmann Binion, Arthur Borem, and Blase Ur. 2024. JupyterLab in Retrograde: Contextual Notifications That Highlight Fairness and Bias Issues for Data Scientists. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3613904.3642755

[145] Björn Hartmann, Loren Yu, Abel Allison, Yeonsoo Yang, and Scott R. Klemmer. 2008. Design as exploration: creating interface alternatives through parallel authoring and runtime tuning. In *Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 91–100. doi:10.1145/1449715.1449732

[146] Robert Hausmann and Kurt Vanlehn. 2010. The Effect of Self-Explaining on Robust Learning. *I. J. Artificial Intelligence in Education* 20 (Jan. 2010), 303–332. doi:10.3233/JAI-2010-010

[147] Gaole He, Lucie Kuiper, and Ujwal Gadiraju. 2023. Knowing About Knowing: An Illusion of Human Competence Can Hinder Appropriate Reliance on AI Systems. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–18. doi:10.1145/3544548.3581025

[148] George Head. 2003. Effective collaboration: Deep collaboration as an essential element of the learning process. *The Journal of Educational Enquiry* 4, 2 (2003), 47–62.

[149] Sofie Heirweg, Mona De Smul, Emmelien Merchie, Geert Devos, and Hilde Keer. 2020. Mine the Process: Investigating the Cyclical Nature of Upper Primary School Students' Self-Regulated Learning. *Instructional Science* 48, 4 (Aug. 2020), 337–369. doi:10.1007/s11251-020-09519-0

[150] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring Massive Multitask Language Understanding. arXiv:2009.03300 [cs] doi:10.48550/arXiv.2009.03300

[151] Yueh-Ren Ho, Bao-Yu Chen, and Chien-Ming Li. 2023. Thinking More Wisely: Using the Socratic Method to Develop Critical Thinking Skills amongst Healthcare Students. *BMC Medical Education* 23, 1 (March 2023), 173. doi:10.1186/s12909-023-04134-2

[152] Kenneth Holstein, Vincent Aleven, and Nikol Rummel. 2020. A Conceptual Framework for Human–AI Hybrid Adaptivity in Education. In *Artificial Intelligence in Education*, Ig Ibert Bittencourt, Mutlu Cukurova, Kasia Muldner, Rose Luckin, and Eva Millán (Eds.). Vol. 12163. Springer International Publishing, Cham, 240–254. doi:10.1007/978-3-030-52237-7_20 Series Title: Lecture Notes in Computer Science.

[153] Kenneth Holstein, Erik Harpstead, Rebecca Gulotta, and Jodi Forlizzi. 2020. Replay Enactments: Exploring Possible Futures through Historical Data. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference*. ACM, Eindhoven Netherlands, 1607–1618. doi:10.1145/3357236.3395427

[154] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. 2023. CogAgent: A Visual Language Model for GUI Agents. arXiv:2312.08914

[155] Eric Horvitz. 1999. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems the CHI is the limit - CHI '99*. ACM Press, Pittsburgh, Pennsylvania, United States, 159–166. doi:10.1145/302979.303030

[156] Stephanie Houde and Charles Hill. 1997. What Do Prototypes Prototype? In *Handbook of Human-Computer Interaction*. Elsevier, 367–381. doi:10.1016/B978-044481862-1.50082-0

[157] Philipp Hugenroth, Valdemar Danry, and Pattie Maes. 2026. Critical Inker: Scaffolding Critical Thinking in AI-Assisted Writing Through Socratic Questioning. arXiv preprint arXiv:2604.07167. arXiv:2604.07167 [cs.HC] doi:10.48550/arXiv.2604.07167

[158] Ada Hurst, Shirley Lin, Claire Treacy, Oscar G. Nespoli, and John S. Gero. 2023. Comparing Academics and Practitioners Q & A Tutoring in the Engineering Design Studio. *Proceedings of the Design Society* 3 (July 2023), 997–1006. doi:10.1017/pds.2023.100

[159] Ivan Hybs and John S. Gero. 1992. An evolutionary process model of design. *Design Studies* 13, 3 (July 1992), 273–290. doi:10.1016/0142-694X(92)90216-W

[160] K. Höök. 2000. Steps to take before intelligent user interfaces become real. *Interacting with Computers* 12, 4 (Feb. 2000), 409–426. doi:10.1016/S0953-5438(99)00006-5

[161] Shamsi T. Iqbal and Brian P. Bailey. 2006. Leveraging characteristics of task structure to predict the cost of interruption. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. Association for Computing Machinery, New York, NY, USA, 741–750. doi:10.1145/1124772.1124882

[162] Shamsi T. Iqbal and Brian P. Bailey. 2007. Understanding and developing models for detecting and differentiating breakpoints during interactive tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. Association for Computing Machinery, New York, NY, USA, 697–706. doi:10.1145/1240624.1240732

[163] Shamsi T. Iqbal and Brian P. Bailey. 2008. Effects of intelligent notification management on users and their tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. Association for Computing Machinery, New York, NY, USA, 93–102. doi:10.1145/1357054.1357070

[164] Amir Jahanlou, William Odom, and Parmit Chilana. 2021. Challenges in Getting Started in Motion Graphic Design: Perspectives from Casual and Professional Motion Designers. In *Proceedings of Graphics Interface 2021 (GI 2021)*. Canadian Information Processing Society, 35 – 45. doi:10.20380/GI2021.06 ISSN: 0713-5424 event-place: Virtual Event.

[165] Heisawn Jeong and Michelene T. H. Chi. 2007. Knowledge convergence and collaborative learning. *Instructional Science* 35, 4 (May 2007), 287–315. doi:10.1007/s11251-006-9008-z

[166] Peiling Jiang, Jude Rayan, Steven P. Dow, and Haijun Xia. 2023. Graphologue: Exploring Large Language Model Responses with Interactive Diagrams. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–20. doi:10.1145/3586183.3606737

[167] Nikhita Joshi, Justin Matejka, Fraser Anderson, Tovi Grossman, and George Fitzmaurice. 2020. MicroMentor: Peer-to-Peer Software Help Sessions in Three Minutes or Less. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3313831.3376230

[168] Harmanpreet Kaur, Alex C Williams, and Walter S Lasecki. 2019. Building Shared Mental Models between Humans and AI for Effective Collaboration. In *CHI 2019 Workshop on Where Is the Human? Bridging the Gap Between AI and HCI*. Association for Computing Machinery, Glasgow, Scotland, 7 pages.

[169] Shabnam Kavousi, Patrick A. Miller, and Patricia A. Alexander. 2020. The Role of Metacognition in the First-Year Design Lab. *Educational Technology Research and Development* 68, 6 (Dec. 2020), 3471–3494. doi:10.1007/s11423-020-09848-4

[170] Anna Kawakami, Nikos Arechiga, Luke Guerdan, Matthew Lee, Yang Cheng, Scott Carter, Anita Sun, Haiyi Zhu, and Alison Hu. 2022. Towards a Learner-Centered Explainable AI. *Workshop on Human-Centered Explainable AI (HCXAI) at the ACM Conference on Human Factors in Computing Systems, CHI 2022* (2022), 7.

[171] Majeed Kazemitabaar, Oliver Huang, Sangho Suh, Austin Z. Henley, and Tovi Grossman. 2025. Exploring the Design Space of Cognitive Engagement Techniques with AI-Generated Code for Enhanced Learning. In *Proceedings of the 30th International Conference on Intelligent User Interfaces (IUI '25)*. Association for Computing Machinery, New York, NY, USA, 695–714. doi:10.1145/3708359.3712104

[172] Majeed Kazemitabaar, Jack Williams, Ian Drosos, Tovi Grossman, Austin Zachary Henley, Carina Negreanu, and Advait Sarkar. 2024. Improving Steering and Verification in AI-Assisted Data Analysis with Interactive Task Decomposition. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3654777.3676345

[173] Shahroz Khan, Erkan Gunpinar, and Bekir Sener. 2019. GenYacht: An Interactive Generative Design System for Computer-Aided Yacht Hull Design. *Ocean Engineering* 191 (Nov. 2019), 106462. doi:10.1016/j.oceaneng.2019.106462

[174] Sumbul Khan and Bige Tunçer. 2019. Gesture and Speech Elicitation for 3D CAD Modeling in Conceptual Design. *Automation in Construction* 106 (Oct. 2019), 102847. doi:10.1016/j.autcon.2019.102847

[175] Abdullah Khanfor. 2023. Tasks Decomposition Approaches in Crowdsourcing Software Development. In *Human Interface and the Management of Information*, Hirohiko Mori and Yumi Asahi (Eds.). Springer Nature Switzerland, Cham, 488–498. doi:10.1007/978-3-031-35129-7_35

[176] Kimia Kiani, Parmit K. Chilana, Andrea Bunt, Tovi Grossman, and George Fitzmaurice. 2020. "I Would Just Ask Someone": Learning Feature-Rich Design Software in the Modern Workplace. In *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 1–10. doi:10.1109/VL/HCC50065.2020.9127288 ISSN: 1943-6106.

[177] Tae Soo Kim, Yoonjoo Lee, Jamin Shin, Young-Ho Kim, and Juho Kim. 2024. EvalLM: Interactive Evaluation of Large Language Model Prompts on User-Defined Criteria. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–21. doi:10.1145/3613904.3642216

[178] Gary Klein, Brian Moon, and Robert Hoffman. 2006. Making Sense of Sensemaking 1: Alternative Perspectives. *IEEE Intelligent Systems* 21 (08 2006), 70–73. doi:10.1109/MIS.2006.75

[179] Scott R. Klemmer, Anoop K. Sinha, Jack Chen, James A. Landay, Nadeem Aboobaker, and Annie Wang. 2000. Suede: A Wizard of Oz Prototyping Tool for Speech User Interfaces. In *Proceedings of the 13th Annual ACM Symposium*

on User Interface Software and Technology (UIST '00). Association for Computing Machinery, New York, NY, USA, 1–10. doi:10.1145/354401.354406

[180] Artur Klingbeil, Cassandra Grützner, and Philipp Schreck. 2024. Trust and Reliance on AI — An Experimental Study on the Extent and Costs of Overreliance on AI. *Computers in Human Behavior* 160 (Nov. 2024), 108352. doi:10.1016/j.chb.2024.108352

[181] Agnes Mercedes Kloft, Robin Welsch, Thomas Kosch, and Steeven Villa. 2024. "AI Enhances Our Performance, I Have No Doubt This One Will Do the Same": The Placebo Effect Is Robust to Negative Descriptions of AI. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–24. doi:10.1145/3613904.3642633

[182] Amy J. Ko and Brad A. Myers. 2004. Designing the Whyline: A Debugging Interface for Asking Questions about Program Behavior. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Vienna Austria, 151–158. doi:10.1145/985692.985712

[183] Janin Koch and Antti Oulasvirta. 2018. Group Cognition and Collaborative AI. In *Human and Machine Learning*, Jianlong Zhou and Fang Chen (Eds.). Springer International Publishing, Cham, 293–312. doi:10.1007/978-3-319-90403-0_15 Series Title: Human–Computer Interaction Series.

[184] Rafal Kocielnik, Saleema Amershi, and Paul N. Bennett. 2019. Will You Accept an Imperfect AI?: Exploring Designs for Adjusting End-user Expectations of AI Systems. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow Scotland Uk, 1–14. doi:10.1145/3290605.3300641

[185] Kenneth R. Koedinger, Albert T. Corbett, and Charles Perfetti. 2012. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science* 36, 5 (2012), 757–798. doi:10.1111/j.1551-6709.2012.01245.x _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1551-6709.2012.01245.x.

[186] Jon Kolko. 2010. Abductive Thinking and Sensemaking: The Drivers of Design Synthesis. *Design Issues* 26, 1 (Jan. 2010), 15–28. doi:10.1162/desi.2010.26.1.15

[187] Jon Kolko. 2010. Sensemaking and Framing: A Theoretical Reflection on Perspective in Design Synthesis. In *Design and Complexity - DRS International Conference 2010*. Design Research Society, Montreal, Canada, 10 pages. https://dl.designresearchsociety.org/drs-conference-papers/drs2010/researchpapers/67

[188] Philip J. Koopman. 2021. A Taxonomy of Decomposition Strategies Based on Structures, Behaviors, and Goals. In *ASME 1995 Design Engineering Technical Conferences Collocated with the ASME 1995 15th International Computers in Engineering Conference and the ASME 1995 9th Annual Engineering Database Symposium*. American Society of Mechanical Engineers Digital Collection, 611–618. doi:10.1115/DETC1995-0195

[189] Thomas Kosch, Jakob Karolus, Johannes Zagermann, Harald Reiterer, Albrecht Schmidt, and Paweł W. Woźniak. 2023. A Survey on Measuring Cognitive Workload in Human-Computer Interaction. *Comput. Surveys* 55, 13s (Dec. 2023), 1–39. doi:10.1145/3582272

[190] Thomas Kosch, Robin Welsch, Lewis Chuang, and Albrecht Schmidt. 2023. The Placebo Effect of Artificial Intelligence in Human–Computer Interaction. *ACM Trans. Comput.-Hum. Interact.* 29, 6 (Jan. 2023), 56:1–56:32. doi:10.1145/3529225

[191] Max Kreminski and John Joon Young Chung. 2024. Intent Elicitation in Mixed-Initiative Co-Creativity. In *Joint Proceedings of the ACM IUI Workshops 2024*. CEUR-WS.org, Greenville, SC, USA, 10 pages.

[192] Max Kreminski, Melanie Dickinson, Joseph Osborn, Adam Summerville, Michael Mateas, and Noah Wardrip-Fruin. 2020. Germinate: A Mixed-Initiative Casual Creator for Rhetorical Games. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 16, 1 (Oct. 2020), 102–108. doi:10.1609/aiide.v16i1.7417

[193] Rebecca Krosnick, Fraser Anderson, Justin Matejka, Steve Oney, Walter S. Lasecki, Tovi Grossman, and George Fitzmaurice. 2021. Think-Aloud Computing: Supporting Rich and Low-Effort Knowledge Capture. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3411764.3445066

[194] Kelly Y. L. Ku and Irene T. Ho. 2010. Metacognitive Strategies That Enhance Critical Thinking. *Metacognition and Learning* 5, 3 (Dec. 2010), 251–267. doi:10.1007/s11409-010-9060-6

[195] Tugba Kulahcioglu, Dmitriy Fradkin, and Sridharan Palanivelu. 2017. Incorporating Task Analysis in the Design of a Tool for a Complex and Exploratory Search Task. In *Proceedings of the 2017 Conference on Conference Human Information Int Eraction and Retrieval*. ACM, 373–376. doi:10.1145/3020165.3022156

[196] Mustafa Kurt and Sevinc Kurt. 2017. Improving Design Understandings and Skills through Enhanced Metacognition: Reflective Design Journals. *International Journal of Art & Design Education* 36, 2 (2017), 226–238. doi:10.1111/jade.12094

[197] Hannu Kuusela and Pallab Paul. 2000. A Comparison of Concurrent and Retrospective Verbal Protocol Analysis. *The American Journal of Psychology* 113, 3 (2000), 387. arXiv:1423365 doi:10.2307/1423365

[198] Benjamin Lafreniere, Parmit K. Chilana, Adam Fourney, and Michael A. Terry. 2015. These Aren't the Commands You're Looking For: Addressing False Feedforward in Feature-Rich Software. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, Charlotte NC USA, 619–628. doi:10.1145/2807442.2807482

[199] Matthew V. Law, Nikhil Dhawan, Hyunseung Bang, So-Yeon Yoon, Daniel Selva, and Guy Hoffman. 2019. Side-by-Side Human–Computer Design Using a Tangible User Interface. In *Design Computing and Cognition '18*, John S. Gero (Ed.). Springer International Publishing, Cham, 155–173. doi:10.1007/978-3-030-05363-5_9

[200] Matthew V. Law, Amritansh Kwatra, Nikhil Dhawan, Matthew Einhorn, Amit Rajesh, and Guy Hoffman. 2020. Design Intention Inference for Virtual Co-Design Agents. In *Proceedings of the 20th ACM International Conference on Intelligent Virtual Agents*. ACM, Virtual Event Scotland UK, 1–8. doi:10.1145/3383652.3423861

[201] Bryan Lawson and Shee Ming Loke. 1997. Computers, words and pictures. *Design Studies* 18, 2 (April 1997), 171–183. doi:10.1016/S0142-694X(97)85459-2

[202] Hao-Ping Lee, Advait Sarkar, Lev Tankelevitch, Ian Drosos, Sean Rintel, Richard Banks, and Nicholas Wilson. 2025. The Impact of Generative AI on Critical

Thinking: Self-Reported Reductions in Cognitive Effort and Confidence Effects From a Survey of Knowledge Workers. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CH '25)*. Association for Computing Machinery, New York, NY, USA, 23. doi:10.1145/3706598.3713778

[203] John. D. Lee and Katrina. A. See. 2004. Trust in Automation: Designing for Appropriate Reliance. *Human Factors: The Journal of the Human Factors and Ergonomics Society* 46, 1 (Jan. 2004), 50–80. doi:10.1518/hfes.46.1.50_30392

[204] Michael Jing Long Lee, Advait Sarkar, and Alan F. Blackwell. 2024. Predictability of Identifier Naming with Copilot: A Case Study for Mixed-Initiative Programming Tools. In *Proceedings of the 35th Annual Conference of the Psychology of Programming Interest Group (PPIG 2024)*. Psychology of Programming Interest Group, Cambridge, United Kingdom, 47–69.

[205] Wendy G. Lehnert. 1978. *The Process of Question Answering: A Computer Simulation of Cognition* (1 ed.). Routledge, London. doi:10.4324/9781003316817

[206] Mosh Levy, Alon Jacoby, and Yoav Goldberg. 2024. Same Task, More Tokens: The Impact of Input Length on the Reasoning Performance of Large Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Bangkok, Thailand, 15339–15353. doi:10.18653/v1/2024.acl-long.818

[207] Jiahao Li, Jeeeun Kim, and Xiang 'Anthony' Chen. 2019. Robiot: A Design Tool for Actuating Everyday Objects with Automatically Generated 3D Printable Mechanisms. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 673–685. doi:10.1145/3332165.3347894

[208] Jenny T. Liang, Chenyang Yang, and Brad A. Myers. 2024. A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges. *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering* (Feb. 2024), 1–13. doi:10.1145/3597503.3608128

[209] Q. Vera Liao, Hariharan Subramonyam, Jennifer Wang, and Jennifer Wortman Vaughan. 2023. Designerly Understanding: Information Needs for Model Transparency to Support Design Ideation for AI-Powered User Experience. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*

(CHI '23). Association for Computing Machinery, New York, NY, USA, 1–21. doi:10.1145/3544548.3580652

[210] Leonhard Lichtschlag, Thorsten Karrer, and Jan Borchers. 2009. Fly: A Tool to Author Planar Presentations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Boston MA USA, 547–556. doi:10.1145/1518701.1518786

[211] Hyunseung Lim, Dasom Choi, Sooyohn Nam, Bogoan Kim, and Hwajung Hong. 2026. Understanding Human–Multi-Agent Team Formation for Creative Work. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*. Association for Computing Machinery, New York, NY, USA, 1–21. doi:10.1145/3772318.3791166

[212] Youn-Kyung Lim, Erik Stolterman, and Josh Tenenberg. 2008. The Anatomy of Prototypes: Prototypes as Filters, Prototypes as Manifestations of Design Ideas. *ACM Transactions on Computer-Human Interaction* 15, 2 (July 2008), 1–27. doi:10.1145/1375761.1375762

[213] David Chuan-En Lin and Nikolas Martelaro. 2024. Jigsaw: Supporting Designers to Prototype Multimodal Applications by Chaining AI Foundation Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–15. doi:10.1145/3613904.3641920

[214] Zhiyu Lin, Upol Ehsan, Rohan Agarwal, Samihan Dani, Vidushi Vashishth, and Mark Riedl. 2023. Beyond Prompts: Exploring the Design Space of Mixed-Initiative Co-Creativity Systems. arXiv:2305.07465 doi:10.48550/arXiv.2305.07465

[215] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–31. doi:10.1145/3544548.3580817

[216] Michael Xieyang Liu, Tongshuang Wu, Tianying Chen, Franklin Mingzhe Li, Aniket Kittur, and Brad A. Myers. 2024. Selenite: Scaffolding Online Sensemaking with Comprehensive Overviews Elicited from Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–26. arXiv:2310.02161 [cs] doi:10.1145/3613904.3642149

[217] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the Middle: How Language Models Use Long Contexts. arXiv:2307.03172 [cs] doi:10.48550/arXiv.2307.03172

[218] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs] doi:10.48550/arXiv.2107.13586

[219] Vivian Liu, Han Qiao, and Lydia Chilton. 2022. Opal: Multimodal Image Generation for News Illustration. arXiv:2204.09007 [cs]

[220] Yefeng Liu, Darren Edge, and Koji Yatani. 2013. SidePoint: A Peripheral Knowledge Panel for Presentation Slide Authoring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. Association for Computing Machinery, New York, NY, USA, 681–684. doi:10.1145/2470654.2470750

[221] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J. Cai. 2020. Novice-AI Music Co-Creation via AI-Steering Tools for Deep Generative Models. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–13. doi:10.1145/3313831.3376739

[222] M. Sravanthi, C. R. Chowdary, and P. S. Kumar. 2009. SlidesGen: Automatic Generation of Presentation Slides for a Technical Paper Using Summarization. In *Proceedings of the Twenty-Second International FLAIRS Conference*. AAAI Press, Palo Alto, CA, USA, 284–289.

[223] Atefeh Mahdavi Goloujeh, Anne Sullivan, and Brian Magerko. 2024. Is It AI or Is It Me? Understanding Users' Prompt Journey with Text-to-Image Generative AI Tools. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3613904.3642861

[224] Shareen Mahmud, Jessalyn Alvina, Parmit K. Chilana, Andrea Bunt, and Joanna McGrenere. 2020. Learning Through Exploration: How Children, Adults, and Older Adults Interact with a New Feature-Rich Application. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–14. doi:10.1145/3313831.3376414

[225] Lauren E. Margulieux, James Prather, Brent N. Reeves, Brett A. Becker, Gozde Cetin Uzun, Dastyni Loksa, Juho Leinonen, and Paul Denny. 2024. Self-Regulation, Self-Efficacy, and Fear of Failure Interactions with How Novices Use LLMs to Solve Programming Problems. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*. ACM, Milan Italy, 276–282. doi:10.1145/3649217.3653621

[226] Chris Martens and Matthew A. Hammer. 2017. Languages of Play: Towards Semantic Foundations for Game Interfaces. In *Proceedings of the 12th International Conference on the Foundations of Digital Games (FDG '17)*. Association for Computing Machinery, New York, NY, USA, 1–10. doi:10.1145/3102071.3102096

[227] Bella Martin and Bruce M. Hanington. 2012. *Universal Methods of Design: 100 Ways to Research Complex Problems, Develop Innovative Ideas, and Design Effective Solutions* (digital ed ed.). Rockport Publishers, Beverly, MA.

[228] Marvin. 2024. Marvin. https://heymarvin.com/.

[229] Mary Lou Maher, Brian Magerko, Dan Venura, Douglas Fisher, Rogelio E. Cardona-Rivera, Nancy Fulda, John Gero, Minwoo Lee, David Wilson, James C. Kaufman, Maithilee Kunda, Michael Muller, Rachel K.E. Bellamy, Maya Ackerman, and Evangelia Chrysikou. 2022. A Research Plan for Integrating Generative and Cognitive AI for Human Centered, Explainable Co-Creative AI. In *GenAICHI: CHI 2022 Workshop on Generative AI and HCI*. GenAICHI, Online, 5 pages.

[230] Mary Lou Maher, Brian Magerko, Dan Venura, Douglas Fisher, Rogelio E. Cardona-Rivera, Nancy Fulda, John Gero, Minwoo Lee, David Wilson, James C. Kaufman, Maithilee Kunda, Michael Muller, Rachel K.E. Bellamy, Maya Ackerman, and Evangelia Chrysikou. 2022. A Research Plan for Integrating Generative and Cognitive AI for Human Centered, Explainable Co-Creative AI. In *GenAICHI: CHI 2022 Workshop on Generative AI and HCI*. GenAICHI, Online, 5 pages. https://generativeaiandhci.github.io/

[231] Utiyama Masao and Hasida Kôiti. 1999. Automatic Slide Presentation from Semantically Annotated Documents. In *Proceedings of the Workshop on Coreference and Its Applications - CorefApp '99*. Association for Computational Linguistics, College Park, Maryland, 25. doi:10.3115/1608810.1608816

[232] Damien Masson, Sylvain Malacria, Géry Casiez, and Daniel Vogel. 2023. Statslator: Interactive Translation of NHST and Estimation Statistics Reporting Styles in Scientific Documents. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, San Francisco CA USA, 1–14. doi:10.1145/3586183.3606762

[233] Damien Masson, Sylvain Malacria, Géry Casiez, and Daniel Vogel. 2024. DirectGPT: A Direct Manipulation Interface to Interact with Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–16. doi:10.1145/3613904.3642462

[234] Damien Masson, Jo Vermeulen, George Fitzmaurice, and Justin Matejka. 2022. Supercharging Trial-and-Error for Learning Complex Software Applications. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3491102.3501895 event-place: New Orleans, LA, USA.

[235] Justin Matejka, Michael Glueck, Erin Bradner, Ali Hashemi, Tovi Grossman, and George Fitzmaurice. 2018. Dream Lens: Exploration and Visualization of Large-Scale Generative Design Datasets. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, Montreal QC Canada, 1–12. doi:10.1145/3173574.3173943

[236] Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. Ambient Help. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2751–2760. doi:10.1145/1978942.1979349

[237] Tara Matthews, Anind K. Dey, Jennifer Mankoff, Scott Carter, and Tye Rattenbury. 2004. A toolkit for managing user attention in peripheral displays. In *Proceedings of the 17th annual ACM symposium on User interface software and technology (UIST '04)*. Association for Computing Machinery, New York, NY, USA, 247–256. doi:10.1145/1029632.1029676

[238] D.Scott McCrickard, Richard Catrambone, C.M Chewar, and John T Stasko. 2003. Establishing tradeoffs that leverage attention for utility: empirically evaluating information display in notification systems. *International Journal of Human-Computer Studies* 58, 5 (May 2003), 547–582. doi:10.1016/S1071-5819(03)00022-3

[239] Ambuj Mehrish, Navonil Majumder, Rishabh Bhardwaj, Rada Mihalcea, and Soujanya Poria. 2023. A Review of Deep Learning Techniques for Speech Processing. arXiv:2305.00359

[240] Anne Meier, Hans Spada, and Nikol Rummel. 2007. A rating scheme for assessing the quality of computer-supported collaboration processes. *International Journal of Computer-Supported Collaborative Learning* 2, 1 (April 2007), 63–86. doi:10.1007/s11412-006-9005-x

[241] Achim Menges and Sean Ahlquist (Eds.). 2011. *Computational Design Thinking* (1. publ ed.). Wiley, Chichester.

[242] Microsoft. 2024. Bing Search API. https://www.microsoft.com/en-us/bing/apis/bing-web-search-api.

[243] Microsoft. 2024. PowerPoint. https://www.microsoft.com/en-us/microsoft-365/powerpoint.

[244] Microsoft. 2024. PowerPoint Copilot. https://support.microsoft.com/en-us/office/create-a-new-presentation-with-copilot-in-powerpoint-3222ee03-f5a4-4d27-8642-9c387ab4854d.

[245] Microsoft. 2024. PowerPoint Designer. https://support.microsoft.com/en-us/office/create-professional-slide-layouts-with-designer-53c77d7b-dc40-45c2-b684-81415eac0617.

[246] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (Feb. 2019), 1–38. doi:10.1016/j.artint.2018.07.007

[247] Miro. 2024. Miro. https://miro.com/.

[248] N. Miyake. 1997. Making Internal Processes External for Constructive Collaboration. In *Proceedings Second International Conference on Cognitive Technology Hu Manizing the Information Age*. IEEE Comput. Soc, 119–123. doi:10.1109/ct.1997.617690

[249] Susan Mohammed and Brad C. Dumville. 2001. Team mental models in a team knowledge framework: expanding theory and measurement across disciplinary boundaries. *Journal of Organizational Behavior* 22, 2 (March 2001), 89–106. doi:10.1002/job.86

[250] Susan Mohammed and Erika Ringseis. 2001. Cognitive Diversity and Consensus in Group Decision Making: The Role of Inputs, Processes, and Outcomes. *Organizational Behavior and Human Decision Processes* 85, 2 (July 2001), 310–335. doi:10.1006/obhd.2000.2943

[251] Hussein Mozannar, Gagan Bansal, Adam Fourney, and Eric Horvitz. 2024. Reading Between the Lines: Modeling User Behavior and Costs in AI-Assisted Programming. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–16. doi:10.1145/3613904.3641936

[252] Hussein Mozannar, Arvind Satyanarayan, and David Sontag. 2022. Teaching Humans When To Defer to a Classifier via Exemplars. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 5 (June 2022), 5323–5331. doi:10.1609/aaai.v36i5.20469

[253] Danil Nagy, Dale Zhao, and David Benjamin. 2018. Nature-Based Hybrid Computational Geometry System for Optimizing Component Structure. In *Humanizing Digital Reality*, Klaas De Rycke, Christoph Gengnagel, Olivier Baverel, Jane Burry, Caitlin Mueller, Minh Man Nguyen, Philippe Rahm, and Mette Ramsgaard Thomsen (Eds.). Springer Singapore, Singapore, 167–176. doi:10.1007/978-981-10-6611-5_15

[254] Suchismita Naik, Austin L. Toombs, Amanda Snellinger, Scott Saponas, and Amanda K Hall. 2025. Designing with Multi-Agent Generative AI: Insights from Industry Early Adopters. In *Proceedings of the 2025 ACM Designing Interactive Systems Conference (DIS '25)*. Association for Computing Machinery, New York, NY, USA, 1961–1972. doi:10.1145/3715336.3735823

[255] K. Nakakoji and Yasuhiro Yamamoto. 2003. Toward A Taxonomy of Interaction Design Techniques for Externalizing in Creative Work. In *Proceedings of the Tenth International Conference on Human-Computer Interaction*. Lawrence Erlbaum Associates, Mahwah, NJ, USA, 1258–1262.

[256] Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. 2020. House-GAN: Relational Generative Adversarial Networks for Graph-Constrained House Layout Generation. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Vol. 12346. Springer International Publishing, Cham, 162–177. doi:10.1007/978-3-030-58452-8_10

[257] Allen Newell and Herbert A. Simon. 1972. *Human Problem Solving*. Prentice-Hall, Oxford, England. xiv, 920 pages.

[258] Mark W. Newman, Mark S. Ackerman, Jungwoo Kim, Atul Prakash, Zhenan Hong, Jacob Mandel, and Tao Dong. 2010. Bringing the Field into the Lab: Supporting Capture and Replay of Contextual Data for the Design of Context-Aware Applications. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. Association for Computing Machinery, New York, NY, USA, 105–108. doi:10.1145/1866029.1866048

[259] Hongwei Niu, Cees Van Leeuwen, Jia Hao, Guoxin Wang, and Thomas Lachmann. 2022. Multimodal Natural Human–Computer Interfaces for Computer-Aided Design: A Review Paper. *Applied Sciences* 12, 13 (June 2022), 6510. doi:10.3390/app12136510

[260] Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. 2024. ScreenAgent: A Vision Language Model-driven Computer Control Agent. arXiv preprint arXiv:2402.07945. arXiv:2402.07945 [cs.HC] doi:10.48550/ARXIV.2402.07945

[261] Ahmed K. Noor. 2017. AI and the Future of the Machine Design. *Mechanical Engineering* 139, 10 (Oct. 2017), 38–43. doi:10.1115/1.2017-Oct-2

[262] nTopology. 2021. Generative Design Software for Multi-Objective Optimization. https://ntopology.com/generative-design-software/.

[263] William Odom, John Zimmerman, Scott Davidoff, Jodi Forlizzi, Anind K. Dey, and Min Kyung Lee. 2012. A Fieldwork of the Future with User Enactments. In *Proceedings of the Designing Interactive Systems Conference*. ACM, Newcastle Upon Tyne United Kingdom, 338–347. doi:10.1145/2317956.2318008

[264] Changhoon Oh, Jungwoo Song, Jinhan Choi, Seonghyeon Kim, Sungwoo Lee, and Bongwon Suh. 2018. I Lead, You Help but Only with Enough Details: Understanding User Experience of Co-Creation with Artificial Intelligence. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3173574.3174223

[265] Yeonjoo Oh, Suguru Ishizaki, Mark D. Gross, and Ellen Yi-Luen Do. 2013. A Theoretical Framework of Design Critiquing in Architecture Studios. *Design Studies* 34, 3 (May 2013), 302–325. doi:10.1016/j.destud.2012.08.004

[266] OpenAI. 2024. GPT-4o. https://openai.com/index/hello-gpt-4o/.

[267] OpenAI. 2024. OpenAI API. https://platform.openai.com.

[268] OpenAI. 2025. Evals. GitHub repository. https://github.com/openai/evals

[269] Antti Oulasvirta and Kasper Hornbæk. 2022. Counterfactual Thinking: What Theories *Do* in Design. *International Journal of Human–Computer Interaction* 38, 1 (Jan. 2022), 78–92. doi:10.1080/10447318.2021.1925436

[270] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training Language Models to Follow Instructions with Human Feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*. Curran Associates Inc., Red Hook, NY, USA, 27730–27744.

[271] Fred Paas, Juhani E. Tuovinen, Huib Tabbers, and Pascal W. M. Van Gerven. 2003. Cognitive Load Measurement as a Means to Advance Cognitive Load Theory. *Educational Psychologist* 38, 1 (Jan. 2003), 63–71. doi:10.1207/S15326985EP3801_8

[272] Srishti Palani, Yingyi Zhou, Sheldon Zhu, and Steven P. Dow. 2022. InterWeave: Presenting Search Suggestions in Context Scaffolds Information Search and Synthesis. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. ACM, Bend OR USA, 1–16. doi:10.1145/3526113.3545696

[273] Ernesto Panadero. 2017. A Review of Self-regulated Learning: Six Models and Four Directions for Research. *Frontiers in Psychology* 8 (April 2017), 422. doi:10.3389/fpsyg.2017.00422

[274] Wamiq Reyaz Para, Shariq Farooq Bhat, Paul Guerrero, Tom Kelly, Niloy Mitra, Leonidas Guibas, and Peter Wonka. 2021. SketchGen: Generating Constrained CAD Sketches. arXiv:2106.02711 doi:10.48550/arXiv.2106.02711

[275] Jieun Park, Sangyeon Kim, and Sangwon Lee. 2023. A User Preference and Intent Extraction Framework for Explainable Conversational Recommender Systems. In *Companion Proceedings of the 2023 ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '23 Companion)*. Association for Computing Machinery, New York, NY, USA, 16–23. doi:10.1145/3596454.3597178

[276] Soya Park, Hari Subramonyam, and Chinmay Kulkarni. 2024. Thinking Assistants: LLM-Based Conversational Assistants That Help Users Think By Asking Rather than Answering. arXiv:2312.06024 [cs] doi:10.48550/arXiv.2312.06024

[277] Debleena Paul, Gaurav Sanap, Snehal Shenoy, Dnyaneshwar Kalyane, Kiran Kalia, and Rakesh K. Tekade. 2021. Artificial Intelligence in Drug Discovery and Development. *Drug Discovery Today* 26, 1 (Jan. 2021), 80–93. doi:10.1016/j.drudis.2020.10.010

[278] Owain Pedgley. 2007. Capturing and Analysing Own Design Activity. *Design Studies* 28, 5 (Sept. 2007), 463–483. doi:10.1016/j.destud.2007.02.004

[279] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. arXiv:2302.06590 [cs.SE] doi:10.48550/arXiv.2302.06590

[280] Savvas Petridis, Michael Xieyang Liu, Alexander J. Fiannaca, Vivian Tsai, Michael Terry, and Carrie J. Cai. 2024. In Situ AI Prototyping: Infusing Multimodal Prompts into Mobile Settings with MobileMaker. In *2024 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, Liverpool, United Kingdom, 121–133. doi:10.1109/VL/HCC60511.2024.00023

[281] Savvas Petridis, Hijung Valentina Shin, and Lydia B Chilton. 2021. SymbolFinder: Brainstorming Diverse Symbols Using Local Semantic Networks. In *The 34th Annual ACM Symposium on User Interface Software and Technology*

(UIST '21). Association for Computing Machinery, New York, NY, USA, 385–399. doi:10.1145/3472749.3474757

[282] Savvas Petridis, Michael Terry, and Carrie Jun Cai. 2023. PromptInfuser: Bringing User Interface Mock-ups to Life with Large Language Models. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems (CHI EA '23)*. Association for Computing Machinery, New York, NY, USA, 1–6. doi:10.1145/3544549.3585628

[283] Peter Pirolli and Stuart Card. 2005. The Sensemaking Process and Leverage Points for Analyst Technology as Identified through Cognitive Task Analysis. In *Proceedings of the International Conference on Intelligence Analysis*. Office of the Assistant Director of Central Intelligence for Analysis and Production, McLean, VA, USA, 2–4.

[284] Carolyn Plumb, Rose Marra, Douglas Hacker, and John Dunlosky. 2018. Measuring Engineering Students' Metacognition with a Think-Aloud Protocol. In *2018 ASEE Annual Conference & Exposition Proceedings*. ASEE Conferences, Salt Lake City, Utah, 30796. doi:10.18260/1-2--30796

[285] Raquel Plumed, Carmen González-Lluch, David Pérez-López, Manuel Contero, and Jorge D Camba. 2021. A Voice-Based Annotation System for Collaborative Computer-Aided Design. *Journal of Computational Design and Engineering* 8, 2 (April 2021), 536–546. doi:10.1093/jcde/qwaa092

[286] Heather Pon-barry, Karl Schultz, Elizabeth Owen Bratt, Brady Clark, and Stanley Peters. 2006. Responding to Student Uncertainty in Spoken Tutorial Dialogue Systems. *International Journal of Artificial Intelligence in Education* 16 (2006), 171–194.

[287] Pontificia Universidad Javeriana, Juanita Tobón, Fabio Tellez, and Oscar Alzate. 2019. Metacognition in the Wild: Metacognitive Studies in Design Education. In *Insider Knowledge - Proceedings of the Design Research Society Learn X Design Conference, 2019*. Design Research Society, Ankara, Turkey, 13 pages. doi:10.21606/learnxdesign.2019.09128

[288] Zachary Pousman and John Stasko. 2006. A Taxonomy of Ambient Information Systems: Four Patterns of Design. In *Proceedings of the Working Conference on*

Advanced Visual Interfaces (AVI '06). Association for Computing Machinery, New York, NY, USA, 67–74. doi:10.1145/1133265.1133277

[289] James Prather, Brent Reeves, Juho Leinonen, Stephen MacNeil, Arisoa S. Randrianasolo, Brett Becker, Bailey Kimmel, Jared Wright, and Ben Briggs. 2024. The Widening Gap: The Benefits and Harms of Generative AI for Novice Programmers. doi:10.48550/arXiv.2405.17739 arXiv:2405.17739 [cs].

[290] Rebecca Anne Price and Peter Lloyd. 2022. Asking Effective Questions: Awareness of Bias in Designerly Thinking. In *Handbook of Engineering Systems Design*, Anja Maier, Josef Oehmen, and Pieter E. Vermaas (Eds.). Springer International Publishing, Cham, 1–16. doi:10.1007/978-3-030-46054-9_24-3

[291] Chen Qu, Liu Yang, W. Bruce Croft, Yongfeng Zhang, Johanne R. Trippas, and Minghui Qiu. 2019. User Intent Prediction in Information-seeking Conversations. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval (CHIIR '19)*. Association for Computing Machinery, New York, NY, USA, 25–33. doi:10.1145/3295750.3298924

[292] Kexin Quan, Dina Albassam, Mengke Wu, Zijian Ding, and Jessie Chin. 2026. Towards AI as Colleagues: Multi-Agent System Improves Structured Ideation Processes. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*. Association for Computing Machinery, New York, NY, USA, 1–26. doi:10.1145/3772318.3790375

[293] R Core Team. 2024. *R: A Language and Environment for Statistical Computing*. Vienna, Austria.

[294] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. https://arxiv.org/abs/2204.06125v1.

[295] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-Shot Text-to-Image Generation. http://arxiv.org/abs/2102.12092 arXiv:2102.12092 [cs].

[296] Jude Rayan, Dhruv Kanetkar, Yifan Gong, Yuewen Yang, Srishti Palani, Haijun Xia, and Steven P. Dow. 2024. Exploring the Potential for Generative AI-based Conversational Cues for Real-Time Collaborative Ideation. In *Creativity and Cognition*. ACM, Chicago IL USA, 117–131. doi:10.1145/3635636.3656184

[297] React.js. 2024. React.Js. https://react.dev/.

[298] reddit.com. 2022. r/Fusion360. https://www.reddit.com/r/Fusion360/

[299] Xiangshi Ren, Gao Zhang, and Guozhong Dai. 2000. An Experimental Study of Input Modes for Multimodal Human-Computer Interaction. In *Advances in Multimodal Interfaces — ICMI 2000 (Lecture Notes in Computer Science)*, Tieniu Tan, Yuanchun Shi, and Wen Gao (Eds.). Springer, Berlin, Heidelberg, 49–56. doi:10.1007/3-540-40063-X_7

[300] Garr Reynolds and Guy Kawasaki. 2020. *Presentation Zen: Simple Ideas on Presentation Design and Delivery* (3rd edition ed.). New Riders, San Francisco, CA.

[301] Jeba Rezwana and Mary Lou Maher. 2023. Designing Creative AI Partners with COFI: A Framework for Modeling Interaction in Human-AI Co-Creative Systems. *ACM Transactions on Computer-Human Interaction* 30, 5 (Sept. 2023), 67:1–67:28. doi:10.1145/3519026

[302] Daiana Rinja, Eduardo Araujo Oliveira, Sonsoles López-Pernas, Mohammed Saqr, Marcus Specht, and Kamila Misiejuk. 2026. Unpacking Vibe Coding: Help-Seeking Processes in Student-AI Interactions While Programming. arXiv:2604.27134 [cs.AI] doi:10.48550/arXiv.2604.27134

[303] Evan F. Risko and Sam J. Gilbert. 2016. Cognitive Offloading. *Trends in Cognitive Sciences* 20, 9 (Sept. 2016), 676–688. doi:10.1016/j.tics.2016.07.002

[304] Horst W. J. Rittel and Melvin M. Webber. 1973. Dilemmas in a General Theory of Planning. *Policy Sciences* 4, 2 (June 1973), 155–169. doi:10.1007/BF01405730

[305] Yvonne Rogers and Harry Brignull. 2003. Computational Offloading: Supporting Distributed Team Working through Visually Augmenting Verbal Communication. In *Proceedings of the 25th Annual Cognitive Science Society* (1st edition ed.). Psychology Press, 6.

[306] Yvonne Rogers, Kay Connelly, Lenore Tedesco, William Hazlewood, Andrew Kurtz, Robert E. Hall, Josh Hursey, and Tammy Toscos. 2007. Why It's Worth the Hassle: The Value of in-Situ Studies When Designing Ubicomp. In *Proceedings of the 9th International Conference on Ubiquitous Computing (UbiComp '07)*. Springer-Verlag, Berlin, Heidelberg, 336–353.

[307] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs] doi:10.48550/arXiv.2112.10752

[308] Debrina Roy, Nicole Calpin, Kathy Cheng, Alison Olechowski, Andrea P. Argüelles, Nicolás F. Soria Zurita, and Jessica Menold. 2024. Designing Together: Exploring Collaborative Dynamics of Multi-Objective Design Problems in Virtual Environments. *Journal of Mechanical Design* 146, 3 (March 2024), 031702. doi:10.1115/1.4063658

[309] Marta Royo, Elena Mulet, Vicente Chulvi, and Francisco Felip. 2021. Guiding Questions for Increasing the Generation of Product Ideas to Meet Changing Needs (QuChaNe). *Research in Engineering Design* 32, 3 (July 2021), 411–430. doi:10.1007/s00163-021-00364-x

[310] Jim Rudd, Ken Stern, and Scott Isensee. 1996. Low vs. High-Fidelity Prototyping Debate. *Interactions* 3, 1 (Jan. 1996), 76–85. doi:10.1145/223500.223514

[311] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. 2022. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. https://arxiv.org/abs/2205.11487v1.

[312] Harshita Sahijwani. 2022. Adaptive Dialogue Management for Conversational Information Elicitation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 3495. doi:10.1145/3477495.3531684

[313] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. doi:10.48550/ARXIV.2402.07927

[314] Advait Sarkar. 2024. AI Should Challenge, Not Obey. *Commun. ACM* 67, 10 (Sept. 2024), 18–21. doi:10.1145/3649404

[315] Advait Sarkar. 2024. Intention Is All You Need. arXiv:2410.18851 [cs] doi:10.48550/arXiv.2410.18851

[316] Advait Sarkar, Xiaotong, Xu, Neil Toronto, Ian Drosos, and Christian Poelitz. 2024. When Copilot Becomes Autopilot: Generative AI's Critical Risk to Knowledge Work and a Critical Solution. arXiv:2412.15030 [cs] doi:10.48550/arXiv.2412.15030

[317] Wout Schellaert, Fernando Martínez-Plumed, Karina Vold, John Burden, Pablo A. M. Casares, Bao Sheng Loe, Roi Reichart, Sean Ó hÉigeartaigh, Anna Korhonen, and José Hernández-Orallo. 2023. Your Prompt Is My Command: On Assessing the Human-Centred Generality of Multimodal Models. *Journal of Artificial Intelligence Research* 77 (June 2023), 377–394. doi:10.1613/jair.1.14157

[318] Matthias Scheutz, Scott A. DeLoach, and Julie A. Adams. 2017. A Framework for Developing and Using Shared Mental Models in Human-Agent Teams. *Journal of Cognitive Engineering and Decision Making* 11, 3 (Sept. 2017), 203–224. doi:10.1177/1555343416682891 Publisher: SAGE Publications.

[319] Johannes Schneider. 2022. Foundation Models in Brief: A Historical, Socio-Technical Focus. arXiv preprint arXiv:2212.08967. arXiv:2212.08967 [cs.AI] doi:10.48550/ARXIV.2212.08967

[320] Donald A. Schön. 1983. *The Reflective Practitioner: How Professionals Think in Action.* Basic Books, New York.

[321] Donald A. Schön. 1992. Designing as reflective conversation with the materials of a design situation. *Know.-Based Syst.* 5, 1 (mar 1992), 3–14. doi:10.1016/0950-7051(92)90020-G

[322] Moushumi Sharmin and Brian P. Bailey. 2011. "I Reflect to Improve My Design": Investigating the Role and Process of Reflection in Creative Design. In *Proceedings of the 8th ACM Conference on Creativity and Cognition (C&C '11).* Association for Computing Machinery, New York, NY, USA, 389–390. doi:10.1145/2069618.2069710

[323] Hua Shen, Tiffany Knearem, Reshmi Ghosh, Kenan Alkiek, Kundan Krishna, Yachuan Liu, Ziqiao Ma, Savvas Petridis, Yi-Hao Peng, Li Qiwei, Sushrita Rakshit, Chenglei Si, Yutong Xie, Jeffrey P. Bigham, Frank Bentley, Joyce Chai, Zachary Lipton, Qiaozhu Mei, Rada Mihalcea, Michael Terry, Diyi Yang, Meredith Ringel

Morris, Paul Resnick, and David Jurgens. 2024. Towards Bidirectional Human-AI Alignment: A Systematic Review for Clarifications, Framework, and Future Directions. arXiv:2406.09264 [cs] doi:10.48550/arXiv.2406.09264

[324] Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 4222–4235. doi:10.18653/v1/2020.emnlp-main.346

[325] Ben Shneiderman. 1981. Direct manipulation: A step beyond programming languages. In *Proceedings of the Joint Conference on Easier and More Productive Use of Computer Systems. (Part - II): Human Interface and the User Interface - Volume 1981 (CHI '81)*. Association for Computing Machinery, New York, NY, USA, 143. doi:10.1145/800276.810991

[326] Ben Shneiderman. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages (VL '96)*. IEEE Computer Society, USA, 336.

[327] Ben Shneiderman. 2007. Creativity support tools: accelerating discovery and innovation. *Commun. ACM* 50, 12 (Dec. 2007), 20–32. doi:10.1145/1323688.1323689

[328] Kumar Shridhar, Jakub Macina, Mennatallah El-Assady, Tanmay Sinha, Manu Kapur, and Mrinmaya Sachan. 2022. Automatic Generation of Socratic Subquestions for Teaching Math Word Problems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 4136–4149. doi:10.18653/v1/2022.emnlp-main.277

[329] Pao Siangliulue, Kenneth C. Arnold, Krzysztof Z. Gajos, and Steven P. Dow. 2015. Toward Collaborative Ideation at Scale: Leveraging Ideas from Others to Generate More Creative and Diverse Ideas. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. Association for Computing Machinery, New York, NY, USA, 937–945. doi:10.1145/2675133.2675239

[330] Sidra Sidra and Claire Mason. 2024. Reconceptualizing AI Literacy: The Importance of Metacognitive Thinking in an Artificial Intelligence (AI)-Enabled Workforce. In *2024 IEEE Conference on Artificial Intelligence (CAI)*. 1181–1186. doi:10.1109/CAI59869.2024.00211

[331] Siemens. 2024. Generative Design | Siemens Software. https://www.sw.siemens.com/en-US/technology/generative-design/.

[332] Auste Simkute, Lev Tankelevitch, Viktor Kewenig, Ava Elizabeth Scott, Abigail Sellen, and Sean Rintel. 2025. Ironies of Generative AI: Understanding and Mitigating Productivity Loss in Human-AI Interaction. *International Journal of Human–Computer Interaction* 41, 5 (March 2025), 2898–2919. doi:10.1080/10447318.2024.2405782

[333] H.A. Simon. 1969. *The Sciences of the Artificial.* M.I.T. Press.

[334] Anjali Singh, Karan Taneja, Zhitong Guan, and Soo Young Rieh. 2026. MetaCues: Enabling Critical Engagement with Generative AI for Information Seeking and Sensemaking. arXiv:2603.19634 [cs.HC] doi:10.48550/arXiv.2603.19634

[335] Nikhil Singh, Guillermo Bernal, Daria Savchenko, and Elena L. Glassman. 2023. Where to Hide a Stolen Elephant: Leaps in Creative Writing with Multimodal Machine Intelligence. *ACM Trans. Comput.-Hum. Interact.* 30, 5 (Sept. 2023), 68:1–68:57. doi:10.1145/3511599

[336] Patrick Stadler and Vanshika Bawa. 2023. Fake It and Let Them Make It: Combining Wizard-of-Oz and Rapid Prototyping Tools for a Holistic Co-Design of Conversational User Interfaces. In *Proceedings of Mensch Und Computer 2023 (MuC '23)*. Association for Computing Machinery, New York, NY, USA, 482–486. doi:10.1145/3603555.3608549

[337] Keith Stenning, Alexander Schmoelz, Heather Wren, Elias Stouraitis, Theodore Scaltsas, Constantine Alexopoulos, and Amelie Aichhorn. 2016. Socratic Dialogue as a Teaching and Research Method for Co-Creativity? *Digital Culture and Education* 8, 2 (2016), 13.

[338] Jonathan M. Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M. Donghia, Craig R. MacNair, Shawn French, Lindsey A. Carfrae, Zohar Bloom-Ackermann, Victoria M. Tran, Anush Chiappino-Pepe, Ahmed H.

Badran, Ian W. Andrews, Emma J. Chory, George M. Church, Eric D. Brown, Tommi S. Jaakkola, Regina Barzilay, and James J. Collins. 2020. A Deep Learning Approach to Antibiotic Discovery. *Cell* 180, 4 (Feb. 2020), 688–702.e13. doi:10.1016/j.cell.2020.01.021

[339] Hari Subramonyam, Roy Pea, Christopher Pondoc, Maneesh Agrawala, and Colleen Seifert. 2024. Bridging the Gulf of Envisioning: Cognitive Challenges in Prompt Based Interactions with LLMs. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3613904.3642754

[340] Hariharan Subramonyam, Colleen Seifert, and Eytan Adar. 2021. ProtoAI: Model-Informed Prototyping for AI-Powered Interfaces. In *26th International Conference on Intelligent User Interfaces*. ACM, College Station TX USA, 48–58. doi:10.1145/3397481.3450640

[341] Sangho Suh, Meng Chen, Bryan Min, Toby Jia-Jun Li, and Haijun Xia. 2024. Luminate: Structured Generation and Exploration of Design Space with Large Language Models for Human-AI Co-Creation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–26. arXiv:2310.12953 [cs] doi:10.1145/3613904.3642400

[342] Sangho Suh, Bryan Min, Srishti Palani, and Haijun Xia. 2023. Sensecape: Enabling Multilevel Exploration and Sensemaking with Large Language Models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, San Francisco CA USA, 1–18. doi:10.1145/3586183.3606756

[343] Jiao Sun, Q. Vera Liao, Michael Muller, Mayank Agarwal, Stephanie Houde, Kartik Talamadupula, and Justin D. Weisz. 2022. Investigating Explainability of Generative AI for Code through Scenario-based Design. In *27th International Conference on Intelligent User Interfaces*. ACM, Helsinki Finland, 212–228. doi:10.1145/3490099.3511119

[344] Amanda Swearngin, Chenglong Wang, Alannah Oleson, James Fogarty, and Amy J. Ko. 2020. Scout: Rapid Exploration of Interface Layout Alternatives through High-Level Design Constraints. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–13. doi:10.1145/3313831.3376593

[345] John Sweller. 1988. Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science* 12, 2 (April 1988), 257–285. doi:10.1207/s15516709cog1202_4

[346] John Sweller, Jeroen J. G. Van Merrienboer, and Fred G. W. C. Paas. 1998. Cognitive Architecture and Instructional Design. *Educational Psychology Review* 10, 3 (Sept. 1998), 251–296. doi:10.1023/A:1022193728205

[347] Fuwen Tan, Song Feng, and Vicente Ordonez. 2019. Text2scene: Generating compositional scenes from textual descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6710–6719.

[348] Lev Tankelevitch, Elena L. Glassman, Jessica He, Aniket Kittur, Mina Lee, Srishti Palani, Advait Sarkar, Gonzalo Ramos, Yvonne Rogers, and Hari Subramonyam. 2025. Understanding, Protecting, and Augmenting Human Cognition with Generative AI: A Synthesis of the CHI 2025 Tools for Thought Workshop. arXiv:2508.21036 [cs.HC] doi:10.48550/arXiv.2508.21036

[349] Lev Tankelevitch, Viktor Kewenig, Auste Simkute, Ava Elizabeth Scott, Advait Sarkar, Abigail Sellen, and Sean Rintel. 2024. The Metacognitive Demands and Opportunities of Generative AI. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–24. doi:10.1145/3613904.3642902

[350] Andrew A. Tawfik, Arthur Graesser, Jessica Gatewood, and Jaclyn Gishbaugher. 2020. Role of Questions in Inquiry-Based Instruction: Towards a Design Taxonomy for Question-Asking and Implications for Design. *Educational Technology Research and Development* 68, 2 (April 2020), 653–678. doi:10.1007/s11423-020-09738-9

[351] Michael Terry, Chinmay Kulkarni, Martin Wattenberg, Lucas Dixon, and Meredith Ringel Morris. 2023. AI Alignment in the Design of Interactive AI: Specification Alignment, Process Alignment, and Evaluation Support. arXiv preprint arXiv:2311.00710. arXiv:2311.00710 [cs.HC] doi:10.48550/ARXIV.2311.00710

[352] Michael Terry and Elizabeth D. Mynatt. 2002. Recognizing creative needs in user interface design. In *Proceedings of the 4th Conference on Creativity & Cognition* (Loughborough, UK) *(C&C '02)*. Association for Computing Machinery, New York, NY, USA, 38–44. doi:10.1145/581710.581718

[353] Michael Terry, Elizabeth D. Mynatt, Kumiyo Nakakoji, and Yasuhiro Yamamoto. 2004. Variation in element and action: supporting simultaneous development of alternative solutions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. Association for Computing Machinery, New York, NY, USA, 711–718. doi:10.1145/985692.985782

[354] Suzanne Tolmeijer, Naim Zierau, Andreas Janson, Jalil Sebastian Wahdatehagh, Jan Marco Leimeister, and Abraham Bernstein. 2021. Female by Default? – Exploring the Effect of Voice Assistant Gender and Pitch on Trait and Trust Attribution. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, 1–7. doi:10.1145/3411763.3451623

[355] Andreas Triantafyllopoulos, Björn W. Schuller, Gökçe İymen, Metin Sezgin, Xiangheng He, Zijiang Yang, Panagiotis Tzirakis, Shuo Liu, Silvan Mertes, Elisabeth André, Ruibo Fu, and Jianhua Tao. 2023. An Overview of Affective Speech Synthesis and Conversion in the Deep Learning Era. *Proc. IEEE* 111, 10 (Oct. 2023), 1355–1381. doi:10.1109/JPROC.2023.3250266

[356] Andrew Turner. 2019. Using Data from Git and GitHub in Ethnographies of Software Development. In *The 2018 Yearbook of the Digital Ethics Lab*, Carl Öhman and David Watson (Eds.). Springer International Publishing, Cham, 35–49. doi:10.1007/978-3-030-17152-0_4

[357] Barbara Tversky and Masaki Suwa. 2009. Thinking with Sketches. In *Tools for Innovation*. Oxford University Press, 75–84. doi:10.1093/acprof:oso/9780195381634.003.0004

[358] Nurcan Gecer Ulu and Levent Burak Kara. 2015. DMS2015-33: Generative Interface Structure Design for Supporting Existing Objects. *Journal of Visual Languages & Computing* 31 (Dec. 2015), 171–183. doi:10.1016/j.jvlc.2015.10.016

[359] Nobuyuki Umetani, Takeo Igarashi, and Niloy J. Mitra. 2012. Guided Exploration of Physically Valid Shapes for Furniture Design. *ACM Transactions on Graphics* 31, 4 (Aug. 2012), 1–11. doi:10.1145/2185520.2185582

[360] Piet Van den Bossche, Wim Gijselaers, Mien Segers, Geert Woltjer, and Paul Kirschner. 2011. Team learning: building shared mental models. *Instructional Science* 39, 3 (May 2011), 283–301. doi:10.1007/s11251-010-9128-3

[361] Maaike Van Den Haak, Menno De Jong, and Peter Jan Schellens. 2003. Retrospective vs. Concurrent Think-Aloud Protocols: Testing the Usability of an Online Library Catalogue. *Behaviour & Information Technology* 22, 5 (Sept. 2003), 339–351. doi:10.1080/0044929031000

[362] Remko van der Lugt. 2002. Functions of sketching in design idea generation meetings. In *Proceedings of the 4th Conference on Creativity & Cognition* (Loughborough, UK) *(C&C '02)*. Association for Computing Machinery, New York, NY, USA, 72–79. doi:10.1145/581710.581723

[363] Maarten van Someren. 1995. The think aloud method: A practical guide to modelling cognitive processes. *Information Processing & Management* 31, 6 (Nov. 1995), 906–907. doi:10.1016/0306-4573(95)90031-4

[364] Kurt VanLehn, Randolph M. Jones, and Michelene T.H. Chi. 1992. A Model of the Self-Explanation Effect. *Journal of the Learning Sciences* 2, 1 (Jan. 1992), 1–59. doi:10.1207/s15327809jls0201_1

[365] Geoffrey Vaughan. 2022. Metacognition and Self-Regulated Learning: Recent Perspectives for an International Context. *Opus et Educatio* 9, 2 (Aug. 2022), 134–138. doi:10.3311/ope.501

[366] Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. 2013. Crossing the bridge over Norman's Gulf of Execution: revealing feedforward's true identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1931–1940.

[367] Steeven Villa, Thomas Kosch, Felix Grelka, Albrecht Schmidt, and Robin Welsch. 2023. The Placebo Effect of Human Augmentation: Anticipating Cognitive Augmentation Increases Risk-Taking Behavior. *Computers in Human Behavior* 146 (Sept. 2023), 107787. doi:10.1016/j.chb.2023.107787

[368] Sruthi Viswanathan, Behrooz Omidvar-Tehrani, Adrien Bruyat, Frédéric Roulland, and Antonietta Maria Grasso. 2020. Hybrid Wizard of Oz: Concept Testing a Recommender System. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–7. doi:10.1145/3334480.3383097

[369] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Tal Linzen, Grzegorz Chrupała, and Afra Alishahi (Eds.). Association for Computational Linguistics, Brussels, Belgium, 353–355. doi:10.18653/v1/W18-5446

[370] Chao-Chih Wang, Peter Kuan-Hao Cheng, and Tzu-Hua Wang. 2022. Measurement of Extraneous and Germane Cognitive Load in the Mathematics Addition Task: An Event-Related Potential Study. *Brain Sciences* 12, 8 (Aug. 2022), 1036. doi:10.3390/brainsci12081036

[371] Dakuo Wang, Elizabeth Churchill, Pattie Maes, Xiangmin Fan, Ben Shneiderman, Yuanchun Shi, and Qianying Wang. 2020. From Human-Human Collaboration to Human-AI Collaboration: Designing AI Systems That Can Work Together with People. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–6. doi:10.1145/3334480.3381069

[372] Fengjie Wang, Yanna Lin, Leni Yang, Haotian Li, Mingyang Gu, Min Zhu, and Huamin Qu. 2024. OutlineSpark: Igniting AI-powered Presentation Slides Creation from Computational Notebooks through Outlines. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–16. doi:10.1145/3613904.3642865

[373] Fengjie Wang, Xuye Liu, Oujing Liu, Ali Neshati, Tengfei Ma, Min Zhu, and Jian Zhao. 2023. Slide4N: Creating Presentation Slides from Computational Notebooks with Human-AI Collaboration. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–18. doi:10.1145/3544548.3580753

[374] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, Anima Anandkumar, Karianne Bergen, Carla P. Gomes, Shirley Ho, Pushmeet Kohli, Joan Lasenby, Jure Leskovec, Tie-Yan Liu, Arjun Manrai, Debora Marks, Bharath Ramsundar, Le Song, Jimeng Sun, Jian Tang, Petar Veličković, Max Welling, Linfeng Zhang, Connor W. Coley, Yoshua Bengio, and Marinka Zitnik. 2023. Scientific Discovery in the Age of Artificial Intelligence. *Nature* 620, 7972 (Aug. 2023), 47–60. doi:10.1038/s41586-023-06221-2

[375] Min Wang and Yong Zeng. 2009. Asking the Right Questions to Elicit Product Requirements. *International Journal of Computer Integrated Manufacturing* 22, 4 (April 2009), 283–298. doi:10.1080/09511920802232902

[376] Qiaosi Wang, Koustuv Saha, Eric Gregori, David Joyner, and Ashok Goel. 2021. Towards Mutual Theory of Mind in Human-AI Interaction: How Language Reflects What Students Perceive About a Virtual Teaching Assistant. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–14. doi:10.1145/3411764.3445645

[377] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent Abilities of Large Language Models. arXiv:2206.07682 [cs.CL] doi:10.48550/arXiv.2206.07682

[378] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22)*. Curran Associates Inc., Red Hook, NY, USA, 24824–24837.

[379] Christopher W. Wiese and C. Shawn Burke. 2019. Understanding Team Learning Dynamics Over Time. *Frontiers in Psychology* 10 (June 2019), 1417. doi:10.3389/fpsyg.2019.01417

[380] Michael Williamson. 2024. Mammoth.js. GitHub repository. https://github.com/mwilliamson/mammoth.js

[381] Judith D. Wilson. 1987. A Socratic Approach to Helping Novice Programmers Debug Programs. In *Proceedings of the Eighteenth SIGCSE Technical Symposium on Computer Science Education - SIGCSE '87*. ACM Press, St. Louis, Missouri, United States, 179–182. doi:10.1145/31820.31755

[382] Thomas Winters and Kory W. Mathewson. 2019. Automatically Generating Engaging Presentation Slide Decks. In *Lecture Notes in Computer Science*. Springer International Publishing, 127–141.

[383] Terry Wood. 1994. Patterns of Interaction and the Culture of Mathematics Classrooms. In *Cultural Perspectives on the Mathematics Classroom*,

Stephen Lerman (Ed.). Springer Netherlands, Dordrecht, 149–168. doi:10.1007/978-94-017-1199-9_10

[384] Robert Woodbury. 2010. *Elements of Parametric Design.* Routledge, London.

[385] Aoyu Wu, Yun Wang, Xinhuan Shu, Dominik Moritz, Weiwei Cui, Haidong Zhang, Dongmei Zhang, and Huamin Qu. 2021. AI4VIS: Survey on Artificial Intelligence Approaches for Data Visualization. arXiv:2102.01330 [cs] doi:10.48550/arXiv.2102.01330

[386] Tongshuang Wu, Michael Terry, and Carrie Jun Cai. 2022. AI Chains: Transparent and Controllable Human-AI Interaction by Chaining Large Language Model Prompts. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22).* Association for Computing Machinery, New York, NY, USA, 1–22. doi:10.1145/3491102.3517582

[387] Boyin Yang, John J Dudley, and Per Ola Kristensson. 2025. Design Activity Simulation: Opportunities and Challenges in Using Multiple Communicative AI Agents to Tackle Design Problems. In *Proceedings of the 7th ACM Conference on Conversational User Interfaces (CUI '25).* Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3719160.3736609

[388] Humphrey Yang, Kuanren Qian, Haolin Liu, Yuxuan Yu, Jianzhe Gu, Matthew McGehee, Yongjie Jessica Zhang, and Lining Yao. 2020. SimuLearn: Fast and Accurate Simulator to Support Morphing Materials Design and Workflows. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology.* ACM, Virtual Event USA, 71–84. doi:10.1145/3379337.3415867

[389] Qian Yang. 2018. Machine Learning as a UX Design Material: How Can We Imagine beyond Automation, Recommenders, and Reminders?. In *AAAI Spring Symposia.* AAAI Press, Palo Alto, CA, USA, 6 pages.

[390] Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T. Iqbal, and Jaime Teevan. 2019. Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* ACM, Glasgow Scotland Uk, 1–12. doi:10.1145/3290605.3300415

[391] Qian Yang, Aaron Steinfeld, Carolyn Rosé, and John Zimmerman. 2020. Re-Examining Whether, Why, and How Human-AI Interaction Is Uniquely Difficult

to Design. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–13. doi:10.1145/3313831.3376301

[392] Qian Yang, John Zimmerman, Aaron Steinfeld, and Anthony Tomasic. 2016. Planning Adaptive Mobile Experiences When Wireframing. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*. ACM, Brisbane QLD Australia, 565–576. doi:10.1145/2901790.2901858

[393] Xuyong Yang, Tao Mei, Ying-Qing Xu, Yong Rui, and Shipeng Li. 2016. Automatic Generation of Visual-Textual Presentation Layout. *ACM Transactions on Multimedia Computing, Communications, and Applications* 12, 2 (Feb. 2016), 33:1–33:22. doi:10.1145/2818709

[394] Koji Yatani, Zefan Sramek, and Chi-Lan Yang. 2024. AI as Extraherics: Fostering Higher-order Thinking Skills in Human-AI Interaction. arXiv:2409.09218 [cs] doi:10.48550/arXiv.2409.09218

[395] JoAnne Yates and Wanda Orlikowski. 2007. The PowerPoint Presentation and Its Corollaries: How Genres Shape Communicative Action in Organizations. In *Communicative Practices in Workplaces and the Professions: Cultural Perspectives on the Regulation of Discourse and Organizations*, Mark Zachry and V Thralls (Eds.). Baywood Publishing Company, Inc. doi:10.2190/CPIC4

[396] Catherine Yeh, Gonzalo Ramos, Rachel Ng, Andy Huntington, and Richard Banks. 2024. GhostWriter: Augmenting Collaborative Human-AI Writing Experiences Through Personalization and Agency. arXiv:2402.08855 [cs]

[397] Weiqiu You, Anton Xue, Shreya Havaldar, Delip Rao, Helen Jin, Chris Callison-Burch, and Eric Wong. 2025. Probabilistic Soundness Guarantees in LLM Reasoning Chains. doi:10.48550/ARXIV.2507.12948

[398] Kirsty Young. 2005. Direct from the Source: The Value of 'think-Aloud' Data in Understanding Learning. *The Journal of Educational Enquiry* 6, 1 (2005), 19–33.

[399] Loutfouz Zaman, Wolfgang Stuerzlinger, Christian Neugebauer, Rob Woodbury, Maher Elkhaldi, Naghmi Shireen, and Michael Terry. 2015. *GEM-NI*: A System for Creating and Managing Alternatives In Generative Design. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, Seoul Republic of Korea, 1201–1210. doi:10.1145/2702123.2702398

[400] J.D. Zamfirescu-Pereira, Eunice Jun, Michael Terry, Qian Yang, and Bjoern Hartmann. 2025. Beyond Code Generation: LLM-supported Exploration of the Program Design Space. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–17. doi:10.1145/3706598.3714154

[401] J.D. Zamfirescu-Pereira, David Sirkin, David Goedicke, Ray LC, Natalie Friedman, Ilan Mandel, Nikolas Martelaro, and Wendy Ju. 2021. Fake It to Make It: Exploratory Prototyping in HRI. In *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction (HRI '21 Companion)*. Association for Computing Machinery, New York, NY, USA, 19–28. doi:10.1145/3434074.3446909

[402] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–21. doi:10.1145/3544548.3581388

[403] Edward Zanders and Lindsay MacLeod. 2018. *Presentation Skills for Scientists: A Practical Guide* (2 ed.). Cambridge University Press. doi:10.1017/9781108567640

[404] Guanglu Zhang, Ayush Raina, Jonathan Cagan, and Christopher McComb. 2021. A Cautionary Tale about the Impact of AI on Human Design Teams. *Design Studies* 72 (Jan. 2021), 100990. doi:10.1016/j.destud.2021.100990

[405] Jiayi Zhang, Conrad Borchers, Vincent Aleven, and Ryan Shaun Baker. 2024. Using Large Language Models to Detect Self-Regulated Learning in Think-Aloud Protocols. doi:10.35542/osf.io/hrtz6

[406] Rui Zhang, Nathan J. McNeese, Guo Freeman, and Geoff Musick. 2021. "An Ideal Human": Expectations of AI Teammates in Human-AI Teaming. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW3 (Jan. 2021), 1–25. doi:10.1145/3432945

[407] Qiwei Zhao, Dong Li, Yanchi Liu, Wei Cheng, Yiyou Sun, Mika Oishi, Takao Osaki, Katsushi Matsuda, Huaxiu Yao, Chen Zhao, Haifeng Chen, and Xujiang Zhao. 2025. Uncertainty Propagation on LLM Agent. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1:*

Long Papers), Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (Eds.). Association for Computational Linguistics, Vienna, Austria, 6064–6073. doi:10.18653/v1/2025.acl-long.302

[408] Zhongyi Zhou, Jing Jin, Vrushank Phadnis, Xiuxiu Yuan, Jun Jiang, Xun Qian, Jingtao Zhou, Yiyi Huang, Zheng Xu, Yinda Zhang, Kristen Wright, Jason Mayes, Mark Sherwood, Johnny Lee, Alex Olwal, David Kim, Ram Iyengar, Na Li, and Ruofei Du. 2024. Experiencing InstructPipe: Building Multi-modal AI Pipelines via Prompting LLMs and Visual Programming. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '24)*. Association for Computing Machinery, New York, NY, USA, 1–5. doi:10.1145/3613905.3648656

[409] Yaqian Zhu and John Kolassa. 2018. Assessing and Comparing the Accuracy of Various Bootstrap Methods. *Communications in Statistics - Simulation and Computation* 47, 8 (Sept. 2018), 2436–2453. doi:10.1080/03610918.2017.1348516

[410] John Zimmerman, Jodi Forlizzi, and Shelley Evenson. 2007. Research through Design as a Method for Interaction Design Research in HCI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, San Jose California USA, 493–502. doi:10.1145/1240624.1240704