NORTHWESTERN UNIVERSITY

Agile Research Studios: Learning Ecosystems to Scale Effective Research Training

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Technology and Social Behavior

(Joint Ph.D. in Computer Science and Communication Studies)

By

Leesha Maliakal Shah

EVANSTON, ILLINOIS

December 2023

Agile Research Studios

Learning Ecosystems to Scale Effective Research Training

Leesha Maliakal Shah

Abstract

To tackle today's most challenging problems, we must explore effective and scalable ways to train the next generation in leading the design and research of high-impact solutions. In addition to core design-research skills (i.e. prototyping techniques and research methodologies), learners must also develop the metacognitive skills necessary for self-directing complex work (i.e. learning to strategically plan, seek help, and reflect on process). Undergraduate research training provides an opportunity to study how we might better prepare the future workforce. However, scaling effective environments to be robust to a growing number of learners is difficult when mentoring resources are limited. Such practical shortcomings can lead students to undervalue research experiences and even lower their self-efficacy in leading complex work, resulting in fewer students receiving the promised benefits of undergraduate research programs.

My focus is to develop ***Agile Research Studios (ARS)***, a learning ecosystem that trains learners in how to execute their metacognitive practices across available community supports to

effectively self-direct research. To enable scalability despite limited mentoring resources, ARS implements a dispersed control approach -- an ecosystem composed of socio-technical *component supports* (i.e. individual studio tools, agile processes, social structures) that distribute learning interactions across a community of practicing researchers. To enable effective research training, ARS fosters self-direction, where students focus on the metacognitive skills required to lead design-research work. To achieve this, ARS implements *subsystems* that interweave component supports via *subroutines* (i.e. sequences of learning interactions) that enable students to practice planning, helpseeking, and reflection skills.

My thesis contributes three iterations of Agile Research Studios, each motivated by emergent challenges in learning ecosystem design. In Chapter 2, I introduce ***Agile Research Studios*** through a pilot study that explores the potential of the ARS *learning ecosystem* to scale effective research training. In Chapter 3, I introduce *Polaris*, a case study of how to extend *subsystems* within the ecosystem with new *supports* and *subroutines* to address critical gaps in how the subsystems scaffold expert practice. In Chapter 4, I introduce *Compass*, a case study of how to augment the ecosystem with *process scaffolds* that train students to manage their process execution as they practice subroutines within a subsystem, and within the ecosystem more broadly. By designing ecosystems that interweave tools, processes, and social structures together to support metacognitive practice in research communities like ARS, we can achieve increasingly scalable and effective learning environments that train students in the skills they need to drive the high-impact solutions of tomorrow.

Acknowledgements

If there is one thing this PhD has taught me, it is the value of support. There are many folks I would like to thank for continually supporting me on this journey.

- To my advisor, Haoqi Zhang: Thank you for the gift of learning to regulate in a supportive community, for challenging me to reach the bars I set for myself, for your patience, and for your unwavering faith in me as I practiced.
- To my mentors, Darren Gergle, Nell O'Rourke, Matt Easterday, Liz Gerber, Julie Hui, Elena Glassman: Thank you for modeling how to engage my mind and heart in my mission as an educator and researcher.
- To my peers, from the Delta Lab and beyond: You all know who you are :) For the pair sessions, the status updates, the Coffee Lab walks, the Tomate runs, thank you. You taught me the value of working alongside people that feel like family.
- To my undergrad army: Thank you for letting me practice as your mentor. It was an honor to witness all the ways in which you grew as researchers, and as people. I hope that you are all design-argumenting your way through life. :)
- To the HCI and Cyberlearning research community: Thank you for being willing to engage in conversations that challenge and expand how we collectively think and work.
- To my friends, you all have been so incredibly patient and supportive, from near and far, on this long road. I'm free to hang out now.
- To my siblings: What a tumultuous few years we've been through as a family. There were so many moments where the waves of life knocked us out of regulation. But you were the ones always there with me on that ship, hanging on together. I love you, my ship-mates.
- To Ivy Aunty: If anyone modeled how to leverage the resources around them, how to get things done, and how to rigorously train young minds, it was you. I didn't realize it while you were here, but you were one of my first teachers. I love you and miss you.

- To my Daddu: I remember when I told you I wanted to do a PhD, you made me promise I would see it through. I remember the day I shared my first draft of a grant with you, the initial ramblings of my vision for changing undergraduate research experiences. You held my hand, blinking back tears, and told me you were proud of me, and what I was trying to do. And just like that, the next morning you had passed away. God willing, we'll meet again, and I'll tell you how it went.
- To my Mommu: If it wasn't for all the long night shifts you pulled as a Cook County ICU nurse, we wouldn't have had the resources to engage in any of the learning communities that shaped me growing up. Your efforts are what allowed me to care so much about education, and to ask how we can make the most of it. I know I've added to your gray hairs on multiple occasions, but each time, you've responded with exemplary strength, and loving support. May I be even half as good a mother as you were to all of us.
- To my Aizu: Oh my little pishudamole, there were many moments over the last year when I wasn't sure I would finish. And moments when I questioned whether I even wanted to do research. But your pappa reminded me that one way you will ask me, and I wanted to be proud of my answer to you. I love what I do. And I pray that you will, too.
- To my Shahaba: Somehow, you put on every hat for me. You were my Coach when I would try to design a lab logo instead of writing my discussion section. You were the Siphon that helped me talk out my ideas when they wouldn't come out on the paper. You were the Arms that hugged me when I cried saying I couldn't do it. You were the Pappa that played with Aizu so I could squeeze in extra hours. You were the Process Nerd that reminded me why research like this matters. You were You. And my God, do I love you.
- To Allah, s.w.t..: You are the Best Planner. Thank you for this journey, and for everyone who made it possible.

Table of Contents

Abstract **3**
Acknowledgements **5**
Table of Contents **7**
List of Figures **10**
Chapter 1. Introduction **12**
1.1 An Ecosystem to Scale Effective Research Training 15
1.2 Addressing Critical Gaps in the Planning Subsystem 19
1.3 Managing Planning Process Execution Across an Ecosystem 22
1.4 Thesis Overview 29
Chapter 2. Agile Research Studios: **An Ecosystem to Scale Effective Research Training** **30**
2.1 Introduction 30
2.2 Background 33
2.2.1 Existing Models for Orchestrating Research Training 34
2.2.2 Technical Limitations to Learning Communities 36
2.3 Challenges in Orchestrating Scalable and Effective Research Training 36
2.3.1 Planning 37
2.3.2 Help-seeking 38
2.4. Agile Research Studios Ecosystem 39
2.4.1 Agile Methodologies 43
2.4.2 Social Structures 43
2.4.3 Studio Tools 45
2.4.4 Community Participation, Project Selection, and Social Norms 50
2.5 ARS Pilot Study 52
2.5.1 Context: Design, Technology, and Research (DTR) 52
2.5.2 Design Iterations 53
2.5.3 Data Collection and Analysis 54
2.6 ARS Findings 55
2.6.1 Conducting and planning research 55
2.6.2 Getting help and making progress 62
2.6.3 Orchestration Time 71

2.6.4 Building self-efficacy in independent research 71
2.7 Discussion 72
2.7.1 Implications for Research Training 73
2.7.2 Study Limitations 75
2.7.3 Future Work in Learning and Orchestration Technologies 78
Chapter 3. Polaris: A System to Scaffold Risk Assessment in Design Arguments 79
3.1 Introduction 80
3.2 Background 83
3.2.1 How Experts Use Design Argumentation 84
3.2.2 Pedagogy Limitations to Training Design Argumentation 85
3.2.3 Existing Tools to Scaffold Design Argumentation 87
3.3 Challenges in Identifying Project Risks and Focusing Plans 89
3.3.1 Students Lack Focus on Design Argumentation 90
3.3.2 Students Fail to Recognize Issues in Design Argument Structure 91
3.4. Polaris System 92
3.4.1 Design Argument Template 93
3.4.2 Reflection Prompts 95
3.5 Polaris Pilot Study 98
3.5.1 Pilot Study Setting 99
3.5.2 Part 1: Student Issue Identification 99
3.5.3 Part 2: Mentor Evaluation and Use of Student Issues 101
3.6 Polaris Findings 103
3.6.1 Students using Polaris identified issues focused on design argumentation 103
3.6.2 Students identified structural issues in design arguments 105
3.6.3 Mentors rated student issues as accurate and severe 110
3.6.4 Mentors found issues articulate and actionable for coaching students 115
3.7 Discussion 124
Chapter 4. Compass: A Framework to Manage Planning Process Across an Ecosystem 129
4.1 Introduction 131
4.2 Background 134
4.2.1 How Experts Adapt their Plans as they Execute 134
4.2.2 How Expert Designers Manage their Planning Process 135
4.2.2 Existing Approaches to Scaffolding Planning Process Management 136

4.3 Challenges in Managing Planning Process in an ARS Ecosystem 137
4.3.1 Expanding the Planning Subsystem in the ARS Ecosystem 137
4.3.2 Student Challenges in Managing Planning Process 143
4.4. Compass: A Process Management Framework for Planning 152
4.4.1 On-action Dashboard 153
4.4.2 In-action Cues 157
4.4.3 Compass Integration into an ARS Ecosystem 162
4.5 Compass Pilot Study 164
4.5.1. Study Setting and Participants 165
4.5.2. Study Procedure 167
4.5.3. Data Collection 169
4.5.4. Data Analysis 171
4.6 Compass Findings 174
4.6.1 Students revise plans midweek when using Compass 174
4.6.2 Students plans were more structurally aligned when using Compass 178
4.6.3 Students plans were better aligned with mentor feedback when using Compass 186
4.7 Discussion 190
Chapter 5. Discussion 193
5.1 An Approach for Scaling Effective Learning Ecosystems 196
5.2 Situating Learning Ecosystems in Their Context 200
5.2.1 Participants Value Self-Direction 200
5.2.2 Participants Maintain a Culture of Self-Direction 201
5.2.3 Community Focus on Design-Research 202
5.3 Evaluating Learning Ecosystems Beyond “Expert Practice” 203
5.3.1 Evaluating Self-Directed Practice in Response to Student Needs 204
5.3.2 Evaluating the Efficacy of Mentors 205
5.3.2 Evaluating the Performance of Subsystems 205
5.4 The Future of Learning Ecosystems for Self-Direction 206
5.5.1 Expanding Learning Ecosystems to Train Different Skills 206
5.5.2 Expanding Learning Ecosystems to Support Mentor Development and Practice 207
5.5.3 Adapting Learning Ecosystems to Alternative Learning Settings 208
Conclusion 210
References 212

List of Figures

Figure 1. A Scalable and Effective Approach to Research Orchestration..... 16
Figure 2. Agile Research Studios Ecosystem to Scale Effective Research Training.....18
Figure 3. Critical Gaps in the Planning Subsystem..... 20
Figure 4. Polaris to Scaffold the Creation and Evaluation of Design Arguments..... 21
Figure 5. Planning Process Breakdowns for Students..... 24
Figure 6. Compass Dashboard to Scaffold Planning Process Management..... 26
Figure 7. Compass Cues to Scaffold Planning Process Management..... 27
Figure 8. Agile Research Studios Ecosystem..... 32
Figure 9. Dispersed Control Approach..... 35
Figure 10. Agile Research Studios Subsystems.....40
Figure 11. Spring Log tool.....47
Figure 12. Pair Research tool.....49
Figure 13. Planning in ARS.....56
Figure 14. Distributed Sprint Log Edits.....59
Figure 15. Average Sprint Log Edits..... 60
Figure 16. Helpseeking in ARS..... 63
Figure 17. Helping Statistics.....64
Figure 18. Helping Graph..... 64
Figure 19. Helping Categories..... 66
Figure 20. Slack Categories..... 68
Figure 21. Slack Helping Interactions..... 69
Figure 22. Critical Gaps in the ARS Planning Subsystem.....80
Figure 23. Polaris System..... 92
Figure 24. Structure of a Design Argument.....94
Figure 25. Reflection Prompts..... 96
Figure 26. Distribution of Issues across Design Argument Structure..... 104
Figure 27. Types of Structural Issues in Design Argument..... 105
Figure 28. Types of Structural Gaps..... 106
Figure 29. Types of Structural Misalignment..... 109
Figure 30. Student Coverage of Issues Identified by Mentors.....112

Figure 31. Mentor Coverage of Issues Students Identified....114
Figure 32. Types of Mentor Feedback for Accurate Issues....116
Figure 33. Types of Mentor Feedback for Inaccurate Issues....117
Figure 34. Breakdowns in Managing Planning Process for Students....130
Figure 35. Expanded Planning Subsystem in ARS....138
Figure 36. The Compass Framework - On-Action Dashboard....154
Figure 37. The Compass Framework - In-Action Cues....157
Figure 38. Compass Cues Before SIG Meeting....158
Figure 39. Compass Cues After SIG Meeting....160
Figure 40. Compass Cues Before Office Hours....161
Figure 41. Compass Cues Before Studio Meeting....162
Figure 42. Compass Study Participants....166
Figure 43. Compass Study Procedure....167
Figure 44. Summary of Midweek Revision Sessions with and without Compass....174
Figure 45. Per-Team Midweek Revision Sessions with and without Compass....175
Figure 46. Summary of Revision Data....176
Figure 47. Detailed Revision Data Per Week, Per Team....177
Figure 48. Structural Alignment of Plans with and without Compass, per Team....178
Figure 49. Structural Alignment of Plans Across Each Week, per Team....180
Figure 50. Team B Example Plan Prior to SIG Meeting (Without Compass)....182
Figure 51. Team B Example Plan Prior to SIG Meeting (With Compass)....184
Figure 52. Alignment of Plans to Mentor Feedback with and without Compass, Per Team....186
Figure 53. Team C Example Plan with Completed Deliverables (Without Compass)....188
Figure 54. Team C Example Plan with Completed Deliverables (With Compass)....189

Chapter 1. Introduction

To tackle today's most challenging problems, we must explore effective and scalable ways to train the next generation in leading the design and research of high-impact solutions. To be effective, learning environments can not only focus on core design-research skills (i.e. prototyping techniques and research methodologies) -- they must also train the metacognitive skills necessary for self-directing complex work. For instance, one must learn to strategically plan out their work, seek help from their network of resources, and regularly reflect on and improve their ways of working in order to make meaningful progress. To scalably train these essential skills, these learning environments must also be robust to a growing number of learners, creating more opportunities to train individuals in the skills they need to lead complex work.

Undergraduate research training provides an opportunity to study how we might better prepare the future workforce. Authentic undergraduate research experiences where students lead core research tasks (i.e. designing a research plan, collecting and analyzing data, and preparing manuscripts) provide numerous personal, professional, and societal benefits, including enhancing student learning, and broadening student participation and retention in diverse fields of study.

However, scaling effective learning environments is difficult when mentoring resources are limited. Often, student researchers learn through 1:1 apprenticeship alongside an expert. While effective, this approach is hard to scale, limiting most faculty to training a small number of graduate students, and leaving undergraduates with rote tasks such as data cleaning or

transcription. Some research groups scale by adopting a hierarchical model, where undergraduate training is delegated to graduate students. While scalable, this approach is less effective, as graduate students have yet to master research, let alone effective research advising. These practical shortcomings can lead students to undervalue research experiences and even lower their self-efficacy in leading complex work, resulting in fewer students receiving the promised benefits of undergraduate research programs.

My focus is to develop ***Agile Research Studios (ARS)***, a learning ecosystem that trains learners in how to execute their metacognitive practices across available community supports to effectively self-direct research. To enable scalability despite limited mentoring resources, ARS implements a dispersed control approach -- an ecosystem composed of socio-technical *component supports* (i.e. individual studio tools, agile processes, social structures) that distribute learning interactions across a community of practicing researchers. To enable effective research training, ARS fosters self-direction, where students focus on the metacognitive skills required to lead design-research work. To achieve this, ARS implements *subsystems* that interweave component supports into *subroutines,* or sequences of learning interactions, that enable students to practice planning, helpseeking, and reflection skills.

My thesis contributes three iterations of Agile Research Studios, each motivated by emergent challenges in learning ecosystem design. The first iteration of ARS introduces a series of component supports and learning interactions that come together as subsystems that enable students to practice planning, helpseeking, and reflection. In Chapter 2, I introduce Agile

Research Studios through a pilot study that explores the potential of an ARS to scale effective research training. The second iteration of ARS responds to critical gaps in how these ARS subsystems scaffolded expert practice, and expands the subsystems with new tools and processes to address these subsystem gaps. In Chapter 3, I introduce *Polaris*, a case study of how we integrate a new planning tool to address critical gaps in how the planning subsystems scaffolded expert practice -- namely, how experts focus their iteration plans by first structuring the design problem, and assessing that structure for risks. The third iteration of ARS responds to the challenge of executing effective process within these subsystems, and integrates process management frameworks that guide students to enact *subroutines* of learning interactions that build their practice within a subsystem. In Chapter 4, I introduce *Compass*, a case study of how we augment the planning subsystem with a process management framework that trains students to manage and revise their planning process as they work, in ways that thoughtfully leverage existing ecosystem supports.

By designing ecosystems that support metacognitive practice in research communities like ARS, we can achieve increasingly scalable and effective learning environments that train students in the skills they need to drive the high-impact solutions of tomorrow. Moving forward, I seek to explore how we can further expand these learning ecosystems to support the practice of regulation skills in a variety of communities and domains, advancing alternative learning environments for innovation.

1.1 An Ecosystem to Scale Effective Research Training

Undergraduate research experiences provide numerous personal, professional, and societal benefits, including enhancing student learning and broadening student participation and retention in diverse fields of study. Research experiences that incorporate factors like quality mentoring, sustained training, and practicing core modes of thinking in the discipline contribute to student intellectual growth and interest in science and research careers [78, 85, 31, 79], especially among women and underrepresented minority students [58, 78, 59, 89]. Effective mentorship is central to training students as they practice the diverse set of skills needed to lead such work.

However, **scaling effective environments for this kind of research training is difficult when mentoring resources are limited**. Typically, student researchers learn through the 1:1 apprenticeship model: a combination of observation, coaching, and practice alongside an expert mentor [28]. While effective, 1-on-1 mentoring is time-intensive, and thus hard to scale. Consequently, most faculty are limited to training only small numbers of graduate students, often leaving undergraduates to rote activities such as data cleaning, transcription, or tagging. In an effort to scale, some research groups adopt a 1:1:1 hierarchical model, where faculty mentors delegate undergraduate mentoring to graduate students. While scalable, this approach provides less effective training to students, as graduate students themselves have yet to master research, let alone methods for effectively mentoring others.

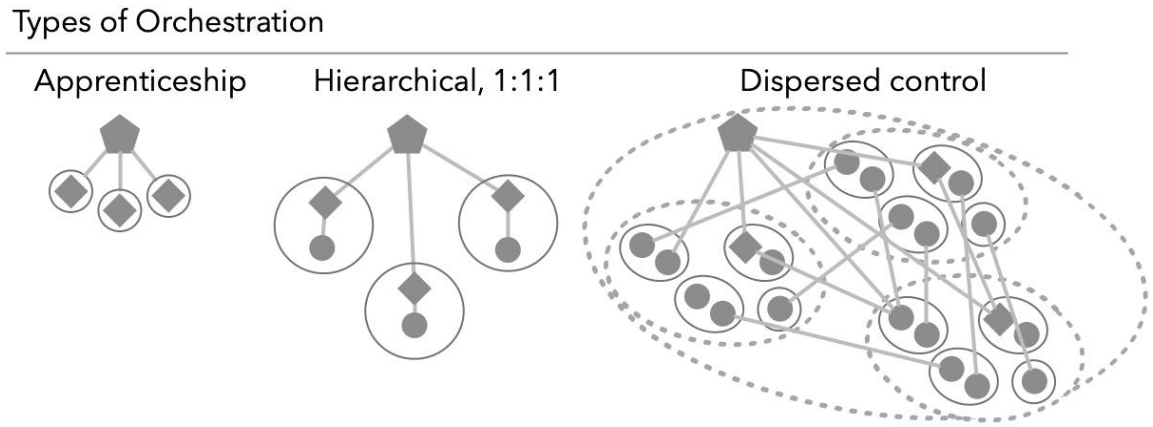


Figure 1. A Scalable and Effective Approach to Research Orchestration
To scale effective research training given limited mentoring resources in the community, *Agile Research Studios (ARS)* fosters a *self-direction,* taking a *dispersed control approach* to distribute learning and practice across an ecosystem of existing supports in the community.

To overcome the practical limitations of scaling effective training environments, we introduce ***Agile Research Studios (ARS)***, **a learning ecosystem of socio-technical supports designed to scaffold students to plan, seek help, and reflect as they learn to self-direct complex work within this community** [1]. To enable effective research training, ARS fosters a self-directed learning environment that focuses on developing the underlying metacognitive skills requisite for leading complex work like design-research. To enable scalability, ARS takes a dispersed control approach that distributes learning interactions across available supports in the community, enabling the practice of core metacognitive skills (see Figure 1).

To practically implement these approaches, the ARS ecosystem is composed of *component supports* (i.e. studio tools, agile processes, social structures) that each help students hone aspects

of their metacognitive practice (see Figure 2, in orange). For instance, students can use a component tool like their Sprint Log tool to lay out an actionable and agile iteration plan for the next two weeks, or a component social structure like connecting to a peer on another project team for help debugging their prototype. In this way, each component in the ecosystem affords unique learning and practice interactions that enable students to distribute their learning, rather than depend solely on a faculty advisor or graduate student mentor for all their learning needs.

Further, ARS interweaves these components as *subsystems* -- collections of supports designed to advance *subroutines,* or sequences of learning interactions that help students practice specific metacognitive skillsets (e.g. planning, help seeking, and reflection), as seen in Figure 2, in teal. For instance, to practice planning, students can use the Sprint Log tool to construct their project plans, and bring that plan into the Special Interest Group (SIG) planning meeting venue to get feedback from their peers, graduate student SIG head, and the faculty advisor on the risks in their approach and planned next steps. ARS integrates these component supports into subsystems that form a learning ecosystem rich with socio-technical support, designed to equip students with the skills they need to advance their practice.

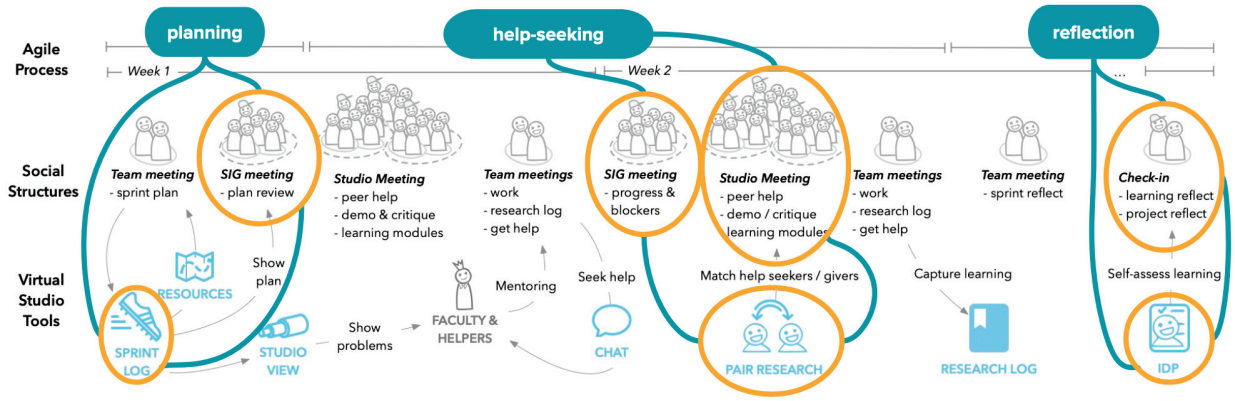


Figure 2. *Agile Research Studios* Ecosystem to Scale Effective Research Training
The Agile Research Studios (ARS) model introduces an ecosystem of socio-technical supports -- studio tools, agile processes, and social structures. The ecosystem weaves these components together as subsystems designed to scaffold students to plan, seek help, and reflect as they learn to self-direct complex work within the community.

Our 2-year pilot study demonstrated how learning ecosystems like ARS can scale effective research training. First, we saw initial evidence that suggested the ARS model was scalable. The Design, Technology, Research (DTR) studio that piloted the model supported a community of 21 student researchers, including 18 undergraduates, 1 post-bac, and 2 Ph.D. students, led by a single faculty with less than 12 hours of faculty time each week. Further, our results demonstrated that the ARS model could be effective. Over two years, the DTR studio hosted 36 students who led 18 research projects, published 9 papers and extended abstracts at peer-reviewed ACM and AAAI conferences, and won 3 awards at ACM Student Research Competitions. Perhaps more remarkable than research outcomes, analysis of ARS activities showed significant engagement in metacognitive processes as students learned to self-direct

complex work. Results showed that students engaged in planning activities, and reported developing planning skills. Further, students helped one another regularly across project teams, and reported shifts in their dispositions on help-seeking and help-giving in the community. This 2-year pilot study of ARS demonstrated the efficacy of learning ecosystems focused on metacognitive skill development, and distributing those interactions across a community of support.

1.2 Addressing Critical Gaps in the Planning Subsystem

As I studied how students practiced in the ecosystem, I observed that **students still faced critical gaps in their planning, helpseeking, and reflection processes**. I empirically studied the expert processes that mentors would coach their students to follow in an ARS. Building on existing literature, I then defined models of expert practice for each of the three skills. When we identified a critical gap in our studio scaffolding, we introduced new tools and processes to augment these planning, helpseeking, and reflection subsystems. For instance, upon studying the expert planning process, I recognized that expert design-researchers visualize the argumentation structure of their design problem, diagnose risks in that structure, and then focus their iteration plan on first tackling the most critical risks. Our previous ARS supports focused on using the Sprint Log tool to help students break down their plans into detailed tasks, and SIG planning meetings, where they could get mentor feedback on their planning strategies for the upcoming

week. However, we lacked explicit scaffolding for two key components of the expert planning process -- visualizing the design problem and diagnosing risks (see Figure 3).

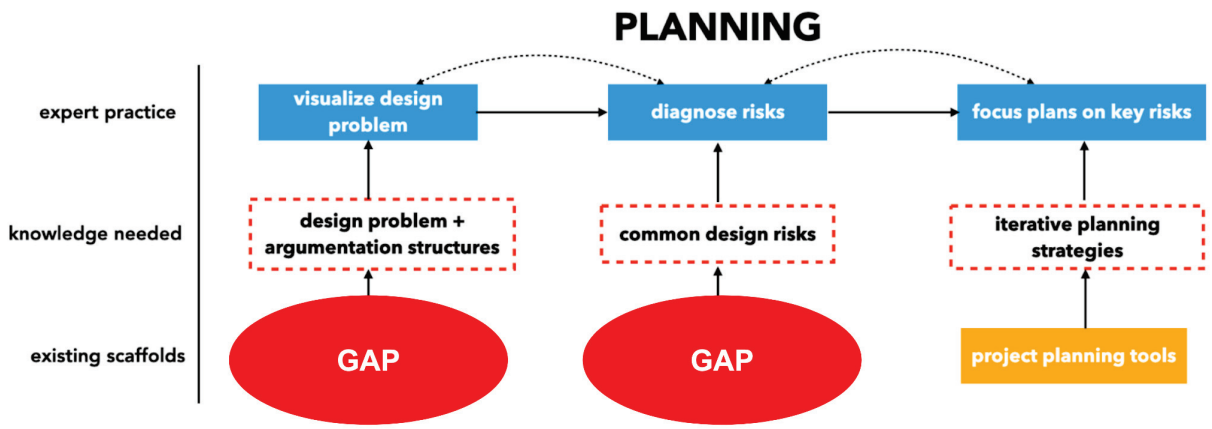


Figure 3. Critical Gaps in the Planning Subsystem
While the existing ARS subsystems scaffolded students in some aspects of expert practice, we found critical gaps where our existing supports lacked scaffolding to support other aspects of how experts work. For instance, ARS scaffolded students to construct iteration plans via the Sprint Log tool and receive feedback on planning strategies in SIG meetings. However, the planning subsystem still had gaps -- namely, scaffolding students to visualize design problem structures and diagnose them for risks as experts do.

To address such gaps, we introduced new supports, extending the corresponding subsystems to better scaffold core aspects of expert practice. For example, we extended the planning subsystem with ***Polaris***, **a representation and risk-assessment tool to scaffold students in constructing and evaluating the argumentation behind their designs** [2]. Polaris embeds expert structural knowledge and diagnostic practice into a *design argument template* and *reflection prompts* (see Figure 4) that focus a novice's attention on three types of issues: gaps (i.e. are there components

of the argument that are missing?), lack of depth (i.e. what is the depth required for each component to form a convincing argument?), and misalignment (i.e. what are the links required between and across components to form a cohesive argument?). Using Polaris, students are able to visually pinpoint the critical risks in their project work, on which to focus their planned next steps. Further, Polaris serves as a conversational scaffold between students and mentors, allowing mentors to easily assess and provide concrete feedback to students on their design rationale and the risks within it.

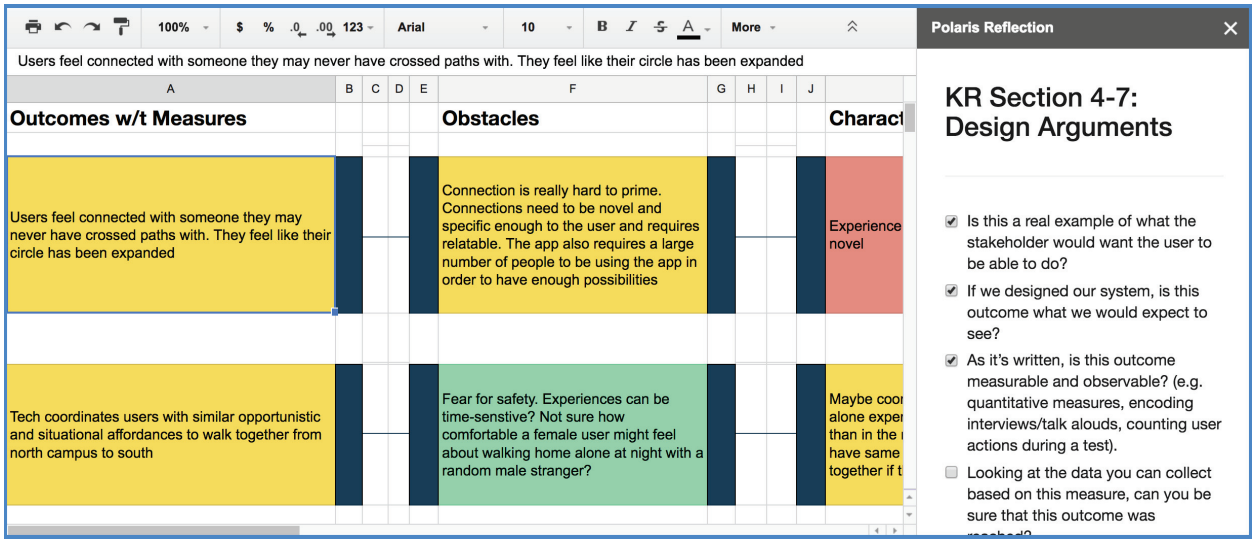


Figure 4. *Polaris* to Scaffold the Creation and Evaluation of Design Arguments
Polaris is a learner-centered diagnostic tool that embeds expert knowledge and practice to scaffold novices to construct and evaluate their design arguments. Novices use the *design argument template* (left) and the *reflection prompts* (right) to visually and procedurally evaluate their design arguments for structural issues.

We conducted a pilot study with 13 undergraduates leading 10 independent research projects. Of the 91 issues that students identified with Polaris, 95.6% of the issues focused on core design argument components, and 96.7% of issues focused on three core types of structural issues: gaps, a lack of depth, or misalignment in their design argumentation. To explore the quality of the issues from the perspective of a mentor, we also asked mentors to generate their own issues, and to evaluate the issues generated by 6 of the 10 teams (71 issues reviewed). Findings show that on average, mentors rated the issues that students identified as accurate (4.06/5) and severe (4.11/5). When comparing mentor and student issues, findings also show that students identified 55.17% of the critical issues identified by mentors. Moreover, mentors felt that student issues expressed a student's current understanding, and informed how mentors would coach students to overcome misunderstandings. These findings suggest that diagnostic tools such as Polaris can scaffold novices to identify structural risks in their design argumentation, as experts do in practice. With these risks in mind, students can then plan iterations that focus on improving the critical aspects of their designs. Further, such scaffolds demonstrate the value of externalizing student understanding into expert structures, enabling better feedback interactions between students and mentors.

1.3 Managing Planning Process Execution Across an Ecosystem

However, this approach of extending the subsystems with new tools, processes, and revised subroutines in response to each emerging gap can have unintended consequences that disrupt the overall learning ecosystem. Namely, the introduction of more tools and processes means that the

subsystems and the overall ecosystem grow in complexity. As students practice via the sequences of learning interactions laid out through these subsystems, **the introduction of more component supports and processes across subsystems can complicate process execution for a learner.** For instance, the introduction of Polaris helped students represent and assess risks in their design argumentation. However, after implementing the support tool, we learned that students conducting design-research need additional representations for other aspects of their project rationale (i.e. interface and systems arguments to argue for how a design should be instantiated and how it will functionally work, or an approach tree to help students argue for the novelty of approach compared to existing approaches.) In response, we expanded Polaris as *linked canvas tools*, a series of templates with additional risk-assessment prompts that helped students represent and assess different layers of argumentation for their approach. However, in testing we also learned that students needed direct coaching and feedback on their argumentation specifically, not just planning feedback provided in SIG meetings. In response we introduced *Mysore*, a feedback and practice venue where students can workshop a risky section of their argumentation alongside a research mentor. While well intended, **a complex learning ecosystem rich with supports can actually inhibit a novice trying to navigate that ecosystem as they practice**.

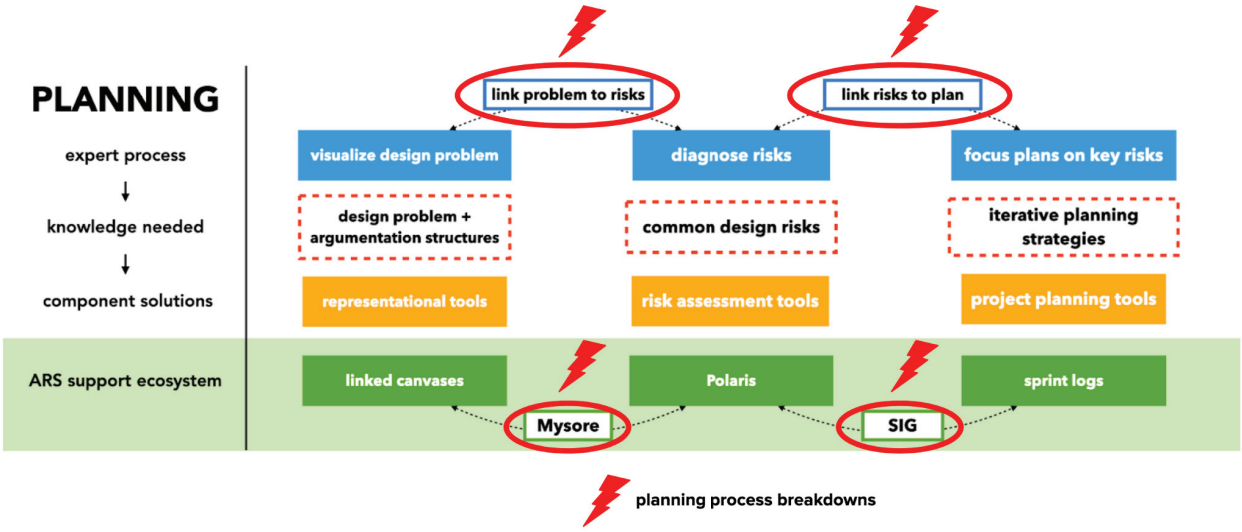


Figure 5. Planning Process Breakdowns for Students
As we extend existing subsystems with new tools and processes, the overall learning ecosystem grows in complexity. We began to observe process breakdowns as students attempted to leverage the ecosystem towards their learning. Here, we see planning process breakdowns, where students struggle to connect their understanding of the problem to critical risks they have, and then connect those risks to their iteration plan.

While the ARS model introduces this rich ecosystem of supports and subsystems designed to scaffold students to practice these learning interactions, the approach fails to explicitly train students to think about how to manage and execute these strategies across the ecosystem of supports available to them, as experts coach them to do. In my work, I observed **process management breakdowns** (see Figure 5), where students may have known the ideal process to execute, but struggled to execute that practice across ecosystem supports. As an example, students who are planning out their work may set deliverables that address risks they identified at the beginning of their sprint. As they continue working, the Polaris risk assessment tool, a

Mysore argumentation feedback session, or a SIG planning meeting may surface a new risk. In such cases, we've observed that students often continue with their previous plan until their mentor raises an issue, rather than adapting their plan to the new risk that was surfaced. This is an example of a **planning process breakdown, where students fail to link the planning feedback they've received via ARS supports to the ways in which they revise their planning process**. As a consequence, students miss out on opportunities to implement the new strategies that the ecosystem of ARS supports helped them surface. For the ecosystem, this can mean poor utilization of the existing supports and subsystems, limiting its potential scalability. For the students, this can mean hindering their metacognitive practice and overall development as a self-directed design-researcher.

To overcome barriers to fully leveraging the ecosystem of available supports, I introduced ***Compass*, a planning process management framework that guides students in how to monitor and revise their metacognitive practice as they execute subroutines -- sequences of learning interactions -- across subsystems in the ecosystem**. The process management framework uses a combination of on-action dashboards and in-action cues to promote reflection and strategy revision retrospectively, and as they practice [76]. On-action dashboards help students zoom out to plan out, monitor, diagnose, and improve the strategies they practice across supports in the ecosystem (see Figure 6). For example, the planning dashboard helps students assess whether the risks they identified in the Polaris tool are aligned with the deliverables they detailed in their Sprint Log tool. In-action cues help students identify opportunities to enact

desired practices as they work with the supports in the ecosystem (see Figure 7). For example, in-action cues can prompt a student to incorporate planning feedback they received in a SIG meeting (e.g. reprioritizing their planned next steps) into a revised plan, or prompt students to update their the status of their plan prior to a mid-week feedback venue, like office hours with their mentor or studio meeting.

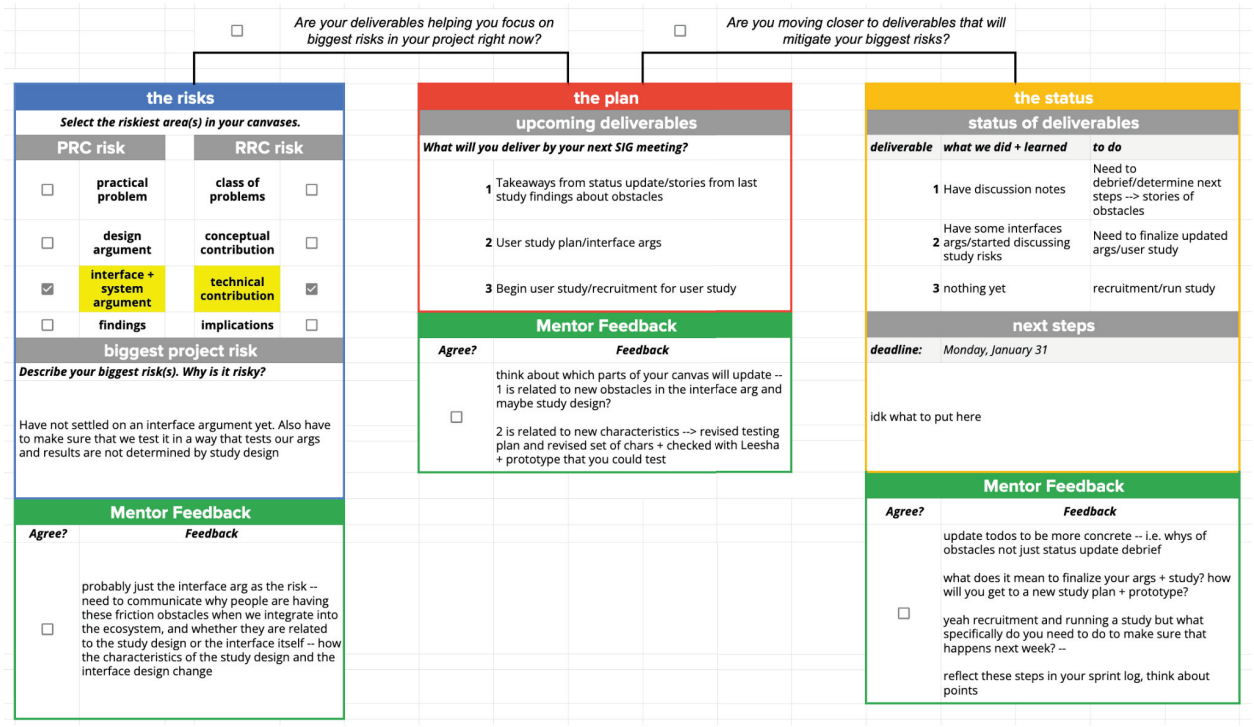


Figure 6. *Compass Dashboard* to Scaffold Planning Process Management
A planning process management framework implemented as (A) a weekly planning dashboard and (B) planning cues via Slack. The dashboard provides students with a view that helps them assess whether their weekly deliverables will address their project risks. The planning cues serve as in-action check-ins that remind students to execute their planning processes at opportune moments (e.g. having students sprint plan before SIG meeting, asking them mid-week if they are still on track for their deliverables, or checking in after a mysore or SIG session to see if they updated risks or plans based on mentor feedback.

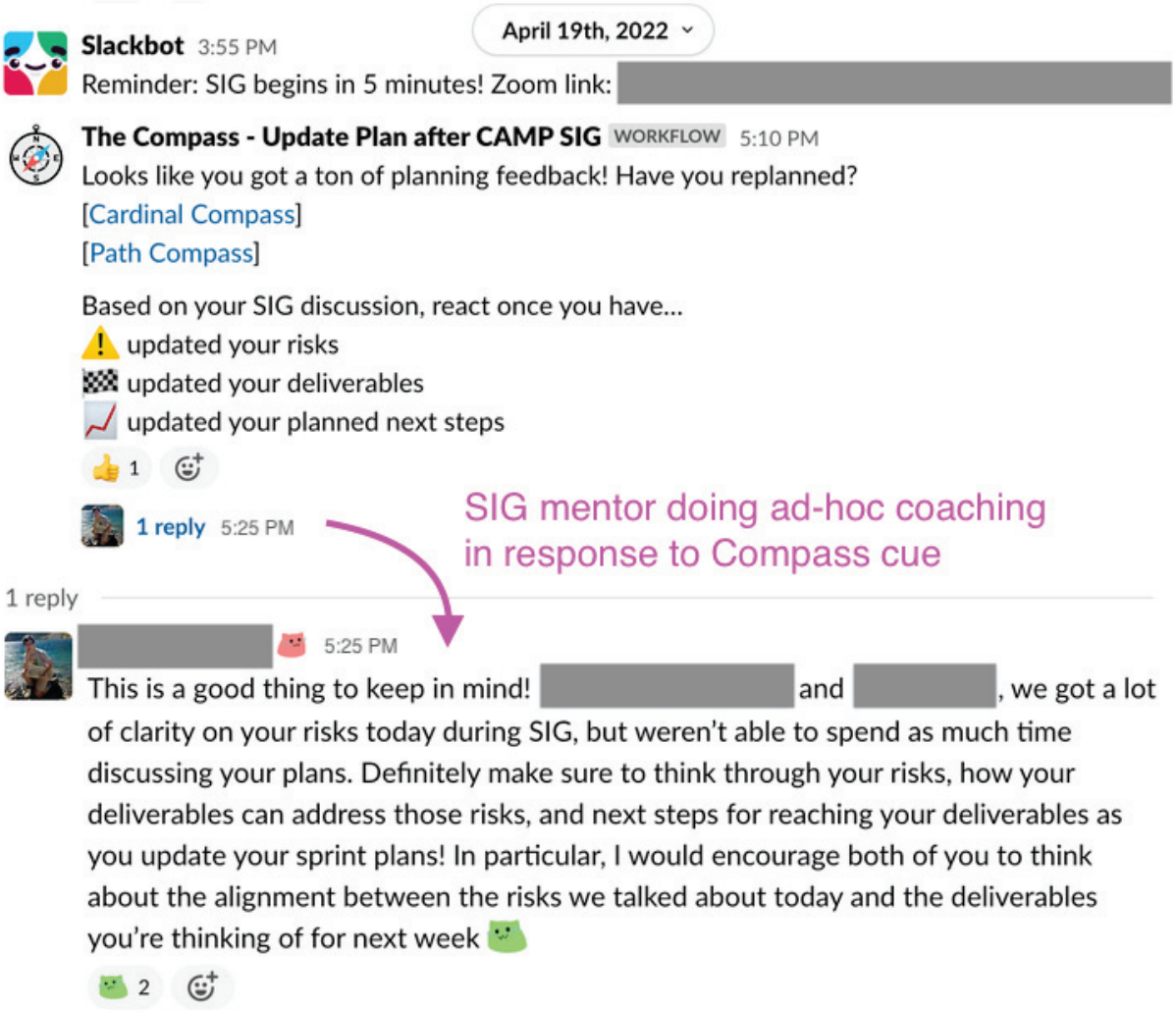


Figure 7. *Compass Cues* to Scaffold Planning Process Management
An example planning cue is sent to a SIG channel with two project teams via Slack. Compass cues them to integrate their planning feedback after their SIG meetings. Mentors can monitor how students use these cues, and even add in their own ad-hoc coaching as needed, as seen here.

This process management framework embeds expert process strategies as *subroutines* that enable students to execute their practice within the ecosystem. For example, while a student is executing a plan, they are now also able to assess whether they are moving towards a deliverable that mitigates the riskiest parts of their design and research work, and replan as needed. Further, this framework integrates into the existing ecosystem supports that students already use to practice. For example, the dashboard view conceptually links to iterative planning activities from the sprint log tool or risk assessment activities from Polaris, and the Slack cues are sent within the SIG channel, where regular SIG communication occurs.

In an 8-week pilot study comparing students' planning process with and without the Compass framework in place, we found that students using Compass revised their plans throughout the week before and after key feedback venues. Further, mentor assessment of the plans demonstrated that student plans were more structurally aligned as students adapted and executed them throughout the week, and that the plans had better implemented the planning feedback students received from their mentor. These findings suggest a need to augment learning ecosystems like ARS with process management frameworks like Compass, to guide students to strategically adapt and execute effective processes as they move forward, learning to leverage the many supports available to them in the ecosystem. Such scaffolding better equips students to learn how to fully harness the power of socio-technical learning ecosystems like ARS.

1.4 Thesis Overview

- Chapter 2 introduces ***Agile Research Studios***, a *learning ecosystem* of socio-technical supports, interwoven as *subsystems* designed to scaffold students to plan, seek help, and reflect as they learn to self-direct complex work within this community.

- Chapter 3 introduces ***Polaris,*** a representation and risk-assessment *tool* that extends the planning subsystem, designed to scaffold students in constructing and evaluating the argumentation behind their designs.

- Chapter 4 introduces ***Compass***, a planning *process management framework* that augments the planning subsystem, designed to guide students through *subroutines* that enable them to monitor, revise, and execute their metacognitive practice as they move across available supports in the ecosystem.

- Chapter 5 reviews the contributions of the thesis, discusses implications and principles for learning ecosystem design, and proposes a vision for future exploration.

Chapter 2. Agile Research Studios

An Ecosystem to Scale Effective Research Training

2.1 Introduction

To tackle today's greatest challenges, it is imperative that we explore effective and scalable ways to train the next generation of problem solvers. In addition to core design-research skills (i.e. prototyping techniques and research methodologies), effective training requires a focus on the metacognitive skills students need to self-direct their work [36]. For instance, one must learn to strategically plan out their work, seek help from their network of resources, and regularly reflect on and improve their ways of working in order to make meaningful progress. To be scalable, these learning environments must also be robust to a growing number of learners, creating more opportunities to prepare individuals with the skills needed to lead complex work.

Undergraduate research training provides an opportunity to study how we might better prepare the future workforce. Undergraduate research experiences provide numerous personal, professional, and societal benefits including enhancing student learning and broadening student participation and retention in diverse fields of study. Quality mentoring, sustained training, and practicing core modes of thinking in the discipline are crucial factors that contribute to student intellectual growth and interest in science and research careers [78, 85, 31, 79], especially among women and underrepresented minority students [58, 78, 59, 89].

However, providing effective mentoring to undergraduate researchers is often limited by practical implementation and orchestration challenges [28]. 1-on-1 mentoring is effective but time-intensive [24]. As a research group expands in size, faculty often struggle to distribute their time and attention to mentor each student. Without significant mentoring, undergraduate students have difficulty engaging in *authentic research* consisting of (a) core design-research activities including forming a hypothesis, collecting and analyzing data, and preparing manuscripts, and (b) metacognitive activities like planning, monitoring and replanning research work, and learning to seek help. Given limited mentoring resources, undergraduate students are usually relegated to rote activities such as data cleaning, transcription, or tagging [84], which is often less challenging and engaging. Some undergraduates perform more engaging and challenging activities, but may struggle to make consistent progress while waiting for busy mentors to help them with encountered obstacles [84]. These practical shortcomings can lead students to discount the value of research experiences and their own self-efficacy [31], resulting in fewer undergraduate students participating in research and few receiving the promised benefits of undergraduate research programs.

The following question drives our research: *How might we design effective and scalable learning ecosystems that foster the skills needed to self-direct authentic research, while being robust to a growing number of learners?* We introduce ***Agile Research Studios (ARS)***, **a learning ecosystem of socio-technical supports designed to scaffold students to plan, seek help, and reflect as they learn to self-direct complex work within a community** [3, 86], see Figure 8. To

enable effective research training, ARS fosters a self-directed learning environment that focuses on developing the underlying metacognitive skills requisite for leading such research work [53, 60, 36]. To enable scalability, ARS takes a dispersed control approach that distributes learning across the community. To support students in practicing core self-regulation skills, ARS introduces subsystems that weave existing supports together into chains of learning interactions that scaffold the practice of planning, helpseeking, and reflection. In other words, the ARS model explicitly trains students to distribute their self-directed learning and practice across the ecosystem of available supports in the community. The ARS approach aims to more fully leverage the support of the research community, which we argue will allow more students to engage in authentic research activities and produce research.

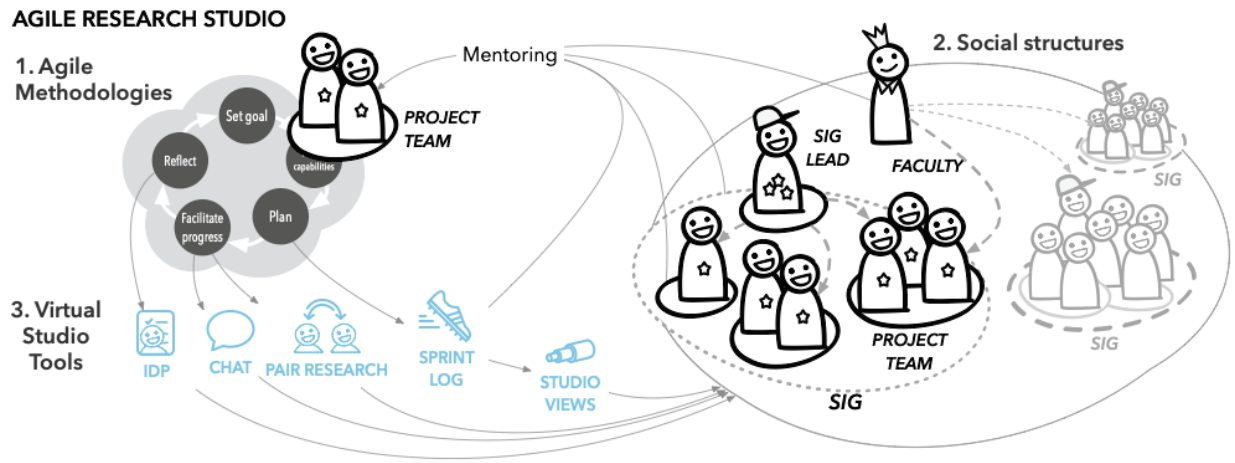


Figure 8. Agile Research Studios Ecosystem
To scale effective research training given limited mentoring resources in the community,*Agile Research Studios (ARS)* fosters a *self-directed learning environment* that takes a *dispersed control approach* that distributes learning across an ecosystem of existing supports in the community. ARS methodologies, social structures, and tools help groups learn better together so more undergraduates can conduct authentic research.

We study the ARS model through an exploratory case study of the Design, Technology, Research (DTR) program, a design-research learning environment implemented and led by a faculty researcher. DTR uses the ARS model to support a community of 21 student researchers including 18 undergraduates, 1 post-bac, and 2 PhDs with less than 12 hours of faculty mentoring each week. Over two years, DTR hosted 36 students who led 18 research projects, published 9 papers and extended abstracts at peer-reviewed ACM and AAAI conferences, and won 3 awards at ACM student research competitions. Analysis of ARS activities show significant engagement in planning activities and students reported developing planning skills. Students helped one another regularly across project teams and reported shifts in their help-seeking and help-giving dispositions. This 2-year pilot study of ARS demonstrated the efficacy of an ecosystem focused on metacognitive skill development, and distributing that development across a community of support.

2.2 Background

To self-direct complex work like design-based research, students must develop *regulation skills*, i.e., cognitive, motivational, emotional, metacognitive, and strategic behaviors for reaching desired goals and outcomes [53, 60, 36]. For instance, students must develop metacognitive skills to improve their ways of working, such as (1) self-directed research planning, monitoring, reflection, and replanning [3, 36]; and (2) adopting effective help-seeking and collaboration to overcome challenges [62, 60]. By developing metacognitive skills like planning and helpseeking,

students can gain quicker access to more central activities within the community of practice, allowing them more effective training that provides an authentic research experience.

However, training many researchers effectively is difficult when mentoring resources are limited. Developing as an effective design-researcher requires students learning the processes, cultures, and mindsets of expert researchers [33]. A faculty member must balance the time and effort required for training students with the research group's need for productivity [29]. With limited time, faculty typically reserve 1-on-1 mentoring for a small number of well-prepared graduate students. Providing effective training to an increased number of students, including undergraduates who are less-prepared, thus requires scaling faculty time in ways that honor the research group's need for research productivity.

2.2.1 Existing Models for Orchestrating Research Training

Traditionally, young researchers develop regulation skills through apprenticeships. Apprenticeship provides a powerful model for research training but imposes an enormous orchestration burden on individual faculty mentors [28]. In the *1:1 apprenticeship* model, students work directly with expert researchers (Figure 9, left). This form of training is effective but demanding on mentoring resources, as it "...requires a very small teacher-to-learner ratio that is not realistic in the large educational systems of modern economies'' [24]. As a consequence only small numbers of graduate students receive training. In the *hierarchical, 1:1:1* model, faculty mentors delegate undergraduate mentoring to graduate students (Figure 9). But graduate

students themselves have yet to master research, let alone effective methods for effectively guiding others [80].

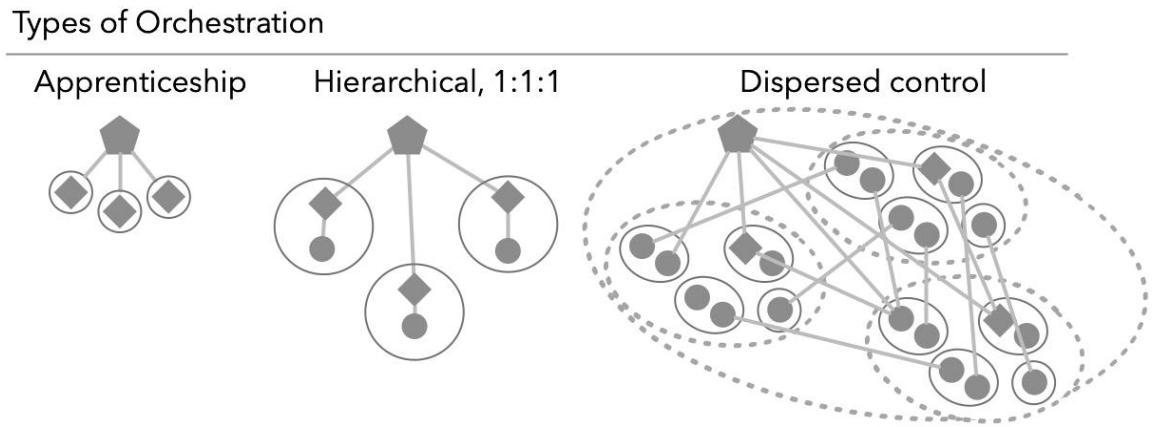


Figure 9. Dispersed Control Approach
To scale effective research training given limited mentoring resources in the community, *Agile Research Studios (ARS)* fosters a *self-directed learning environment* that takes a *dispersed control approach* that distributes learning across an ecosystem of existing supports in the community.

In order to address the "1:X" challenge of enabling one teacher to respond to the learning needs of many students, we must ultimately disperse control of orchestrating learning and support across a network of student researchers [4] (Figure 9, right). In the *dispersed control* model, research activities are situated within a learning community [11, 16] --- a community of practice [88] designed for learning. Learning communities leverage the diversity of member expertise, value individual contributions, support continual advancement of knowledge and skills, emphasize learning how to learn, and provide mechanisms for sharing what is learned. We aim to apply the insights and design principles of learning communities to support research training for large numbers of student researchers in ways that scale faculty mentors' time and their need to make effective research progress.

2.2.2 Technical Limitations to Learning Communities

To scale mentor time, our work aims to distribute planning and help activities across students and their peers by (a) developing students' research ability and self-directed learning skills so that they self-regulate learning (SRL) to conduct research more independently; (b) promoting student teams' socially shared regulation of learning (SSRL) so teams orchestrate their own learning; and (c) connecting students to peers, resources, and instruction to co-regulate learning (CoRL) so that students support each other beyond immediate project teams [53, 60, 3]. While these activities should distribute the mentoring load, orchestrating them is itself time-consuming for mentors and difficult to support with software alone [53]. To successfully scale effective research training communities, we propose a socio-technical ecosystem that consists of methods, community structures, and tools that collectively orchestrate research training so that such activities can feasibly take place within the community.

2.3 Challenges in Orchestrating Scalable and Effective Research Training

Our work seeks to address practical socio-technical challenges in orchestrating the development of regulation skills in a learning community, centered around the metacognitive skills of planning and help-seeking.

2.3.1 Planning

- **Students kept from driving projects.** In many research labs, even experienced undergraduate students may only perform a single research step (e.g., data collection); they rarely engage in all central tasks such as planning projects, generating and testing hypotheses, and authoring publications [84]. While this allows less experienced undergraduates to participate in research, it limits their learning.

- **Mentors do research planning.** As novice researchers, undergraduates lack skills for forming feasible and effective research plans, monitoring these plans as work progresses, and adjusting these plans in response to the inevitable challenges that arise in the course of research [3]. As a consequence, mentors often take responsibility for planning.

- **Poor student planning due to limited mentoring resources.** When faculty prioritize short-term research productivity, they may use scarce 1-on-1 mentoring time predominantly for communicating work progress and overcoming technical problems. As a consequence, coaching self-directed planning and reflection is often deferred [84], even though these practices are vital for developing metacognitive planning skills [76, 17, 24]. Consequently, faculty must tolerate projects going off track when they have too many students for whom to explicitly plan out work

2.3.2 Help-seeking

- **Distributing help.** Without sufficient guidance, student researchers can easily become lost, confused, and frustrated [84]. Distributing help by encouraging helpseeking among peers is difficult to encourage as it runs counter to common educational norms in which educators are the main source of help [38]. Students may waste a significant amount of time before seeking help on research projects, and often only at a project's end [7]. While senior mentors are the most capable of addressing a wide variety of needs, their availability is most limited [41] and are seldom able to provide undergraduate researchers with all the help they need [84].

- **Practicing help-seeking.** Even with a learning community where helpers are readily available, getting help can be challenging for students because the community consists of many individuals working on different projects, making the task of identifying, selecting, and enlisting qualified helpers difficult to execute [62].

- **Coaching help-seeking.** Students often need additional help-seeking skills in order to learn effectively [62, 67, 75]. Without them, students may be reluctant to ask for help even when they need it and help is available [74]. In a survey of 123 university students undertaking student-led research projects, only 3% of students reported that getting help on research would support progress [7].

2.4. Agile Research Studios Ecosystem

How might we design effective and scalable socio-technical learning ecosystems that foster the skills needed to self-direct authentic research, while also being robust to a growing number of learners? Specifically, in this project we focus on planning and help-seeking skills by asking:

> **RQ1:** How can we orchestrate student researchers learning to effectively plan authentic research inquiry, monitor and reflect on their progress, and make adjustments?
>
> **RQ2:** How can we orchestrate help-seeking and collaboration, to effectively leverage the distributed expertise of community members to support progress-making and learning?

We propose that we can address these questions by integrating learning communities, agile methodologies, and online technologies to create the Agile Research Studios (ARS) ecosystem that supports students (1) learning self-directed research planning, monitoring, reflection, and replanning so that they can lead their own projects; and (2) adopting effective help-seeking and collaboration so that they support each other to learn and make progress. To achieve an effective and scalable learning community, ARS consist of a series of supports -- (a) *agile methodologies*, (b) *social structures* including team meetings, special interest group meetings and studio meetings; and (c) *studio tools* including sprint logs, resources, studio views, chat, pair research, research logs, and individual development plans -- that weave together as *subsystems* designed to advance student practice in core regulation skills, like planning and helpseeking (see Figure 10).

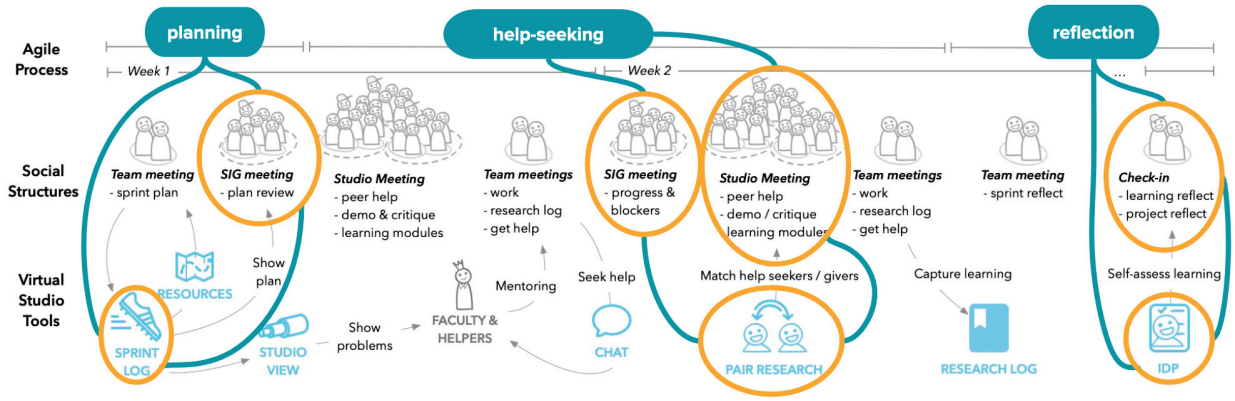


Figure 10. Agile Research Studios Subsystems
Agile Research Studios support regulation skill development by introducing subsystems that interweave existing supports into chains of learning interactions that guide students to practice research planning, progress making, and reflection in 2-week sprint cycles. In these subsystems, studio tools extend social structures to more effectively orchestrate learning and support in and out of the classroom.

To overcome orchestration challenges of learning to plan research, ARS introduces a *planning subsystem,* that supports the following learning interactions:

- **Driving all research steps.** To engage more students in authentic research, ARS adapts agile methodologies to *slice* research work vertically to fit student competencies [23] and promote progress across all phases of research. In this way, students are able to complete frequent iterations, allowing them to grow their project in complexity and generalizability over time, as their skills and the research work matures.

- **Practicing planning.** In the ARS model, students take on the responsibility for planning their work at frequent intervals following agile methodologies. Students break down their

plans by recording tasks and progress in *sprint logs* that support students' and mentors' awareness of progress and potential needs for replanning.

- **Coaching planning.** To help students learn to plan research work on their own, mentors in an ARS focus on delivering weekly plan feedback through *special interest group (SIG) meetings*. This meeting facilitates peer review and feedback by mentors and students to help student teams develop their planning skills, devise strategies to overcome challenges, and connect to studio *resources* [53, 60]. To promote reflection, student teams use *research logs* to record and reflect on research progress throughout a sprint and complete self-assessments in the form of *independent development plans* at quarterly intervals.

To overcome orchestration challenges of learning to seek help, ARS introduces a *help-seeking subsystem,* that supports the following learning interactions:

- **Distributing help.** To better support students while respecting the limits on mentor time, in an ARS the responsibility for providing help is shared across the entire community. Instead of relying on a single mentor to resolve problems, the ARS model seeks to make effective use of the diverse sets of expertise that individual community members have by connecting students to those who can best help on a particular problem. This should enable the community to fulfill numerous help requests without exclusively depending on mentor time, and lead to students feeling more supported.

- **Practicing help-seeking.** To help students connect to peers who can help them, ARS scaffolds the process of getting help by using *pair research* [61] to match students to help one another, *SIG & studio meetings* to facilitate students connecting to helpful peers and mentors in and out of their SIG, and *chat* programs such as Slack to enable students to seek and receive help on-demand.

- **Coaching help-seeking.** ARS normalizes help-seeking and trains students to seek help effectively. Further, as students are connected to help and help themselves with the support of the above-mentioned scaffolds, we expect the common practice of getting and giving help to over time lead to broad shifts in students' help-seeking dispositions [74].

Having presented our research questions and design arguments, we provide in this section a detailed description of the ARS model, as it is implemented in the Design, Technology, and Research program. Figure 10 presents an overview of the ecosystem, and the specific subsystems designed to scaffold planning, helpseeking, and reflection practice. The ecosystem is composed of components and interactions among the processes, social structures, and tools that collectively describe how we propose to create Agile Research Studios.

Agile Research Studios supports a research community of undergraduates, graduates, and faculty with varying interests, expertise, and experiences to (learn to) conduct research. A studio is typically organized around a broad research theme that matches a faculty mentor's research interest. Studios can fit within typical academic calendars and allow students to continue for

multiple quarters or semesters to promote deep, interest-driven learning and to develop research skills and products over time. Importantly, *all students, regardless of seniority, conduct independent research and receive authentic research practice.* Students and mentors pursue projects that match their mutual interests. Each project team is kept small (1--3 people) to promote research ownership.

2.4.1 Agile Methodologies

Agile Research Studios supports students' development of regulation skills by adapting agile methodologies [83, 22, 56, 52] to research through *sprint* cycles. Each sprint is a set period of time (e.g., 2 weeks) during which students plan specific work to be completed that delivers value to the research project. Student teams meet over the course of each sprint to plan research work, make and share progress, identify difficulties, replan, and reflect on progress to improve future plans. They also regularly receive coaching and feedback from mentors and peers who help them (learn to) set appropriate and feasible goals, and devise strategies for making progress and overcoming blockers.

2.4.2 Social Structures

To support student researchers learning regulation skills within a community of practice and to scale faculty time, we propose three social structures to orchestrate shared regulation of learning throughout each sprint:

- ***Team meetings*** bring together students on the same project to: (a) make sprint plans; (b) perform research; (c) get in-team help; (d) record progress through *research logs* [70] and (e) reflect on sprint outcomes.

- ***Special Interest Group (SIG) meetings*** bring together undergraduate students, graduate students, and faculty working on different projects in the same research area [88]. Each SIG is its own mini-studio initially led by a faculty member whose leadership fades over time as a graduate student *SIG head* gains competencies in mentoring and becomes the leader of their own SIG. At the start of a sprint, teams share the outcome of their last sprint and present their current sprint plans for review. Halfway through a sprint, teams present their progress and SIG members help devise strategies for overcoming blockers. The purpose of planning together is to help students gain planning skills [53]; students receive coaching and feedback from mentors and peers to increase learning [3, 24, 49, 50]. Mentors and peers prompt teams to: (a) describe how their sprint goals connect to their larger research goals; (b) clarify what the actual deliverables of the sprint would be, and what value it would have; (c) consider alternative plans; (d) assess whether tasks can feasibly be completed in time; and (e) generate strategies for better *slicing* [23] work. While newer students receive more scaffolding and direction, senior students are challenged to demonstrate and practice their regulation skills in formulating and revising plans.

- ***Studio meetings*** bring together all researchers in a studio to promote progress, learning, and collaboration across SIGs. The studio meeting consists of: (a) *work time*, during which students work on their own projects, (b) *peer help*, during which students formally or informally seek and receive help from others; (c) *demo & critique*, which allows students to share work progress and receive valuable feedback from the entire community; and (d) *learning modules*, which are led by faculty members and students to share "tools of the trade" and teach research skills that benefit all students. Studio meetings embed students in a larger community of practice beyond the problem domain of their SIG. Multifaceted community support provided via multiple mechanisms address diverse students needs at different stages of research and their personal development.

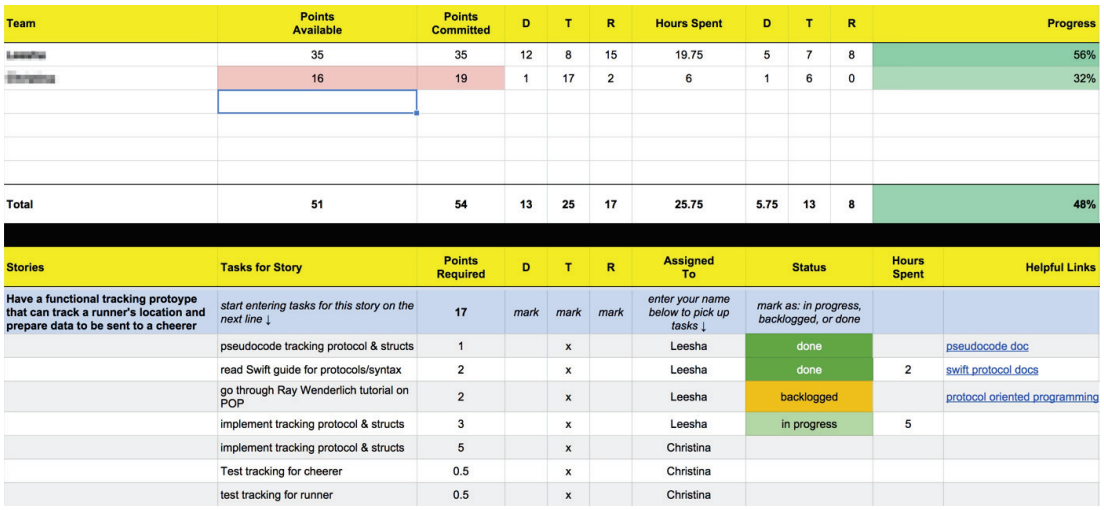
These social structures support meeting in-person to plan work, conduct research, provide and receive help, and reflect and share progress. Depending on its purpose, meetings can be 1-on-1, in small groups, and across the entire studio. To further support student reflection and development over time, bi-quarterly *check-in* meetings facilitate faculty and graduate student mentors meeting with students individually and in their teams to reflect on their learning and project progress.

2.4.3 Studio Tools

To effectively orchestrate learning and instruction, we have developed *studio tools* that extend social structures and in-person meetings to orchestrate learning and support progress-making in

and outside of the classroom [39]. Studio tools promote 3 interrelated feedback processes: (a) a sprint planning and replanning process in which students and mentors receive feedback on project plans and progress, (b) a help and collaboration process that helps students scope help requests and connects to helpers, and (c) a reflection process that promotes awareness of learning and project progress to support growing over time. We describe below the affordances that studio tools should provide:

- **Sprint Logs** are interfaces that allow student teams to record all tasks they plan to do for a sprint, update task progress throughout a sprint, and replan as needed (see Figure 11). Following professional sprint planning practices, students use the sprint logs to enter high-level deliverables, or *stories*, and the *tasks* for completing those stories. To prioritize work, students assign *points* to stories and tasks to estimate the value of the work to the research and the effort required to complete it. This helps students to think through the process of scoping out work that they can feasibly accomplish within a sprint that delivers value for their research. During a SIG meeting, displaying the sprint log facilitates students, peers, and mentors communicating their plan, reporting progress, and devising strategies to overcome blockers.



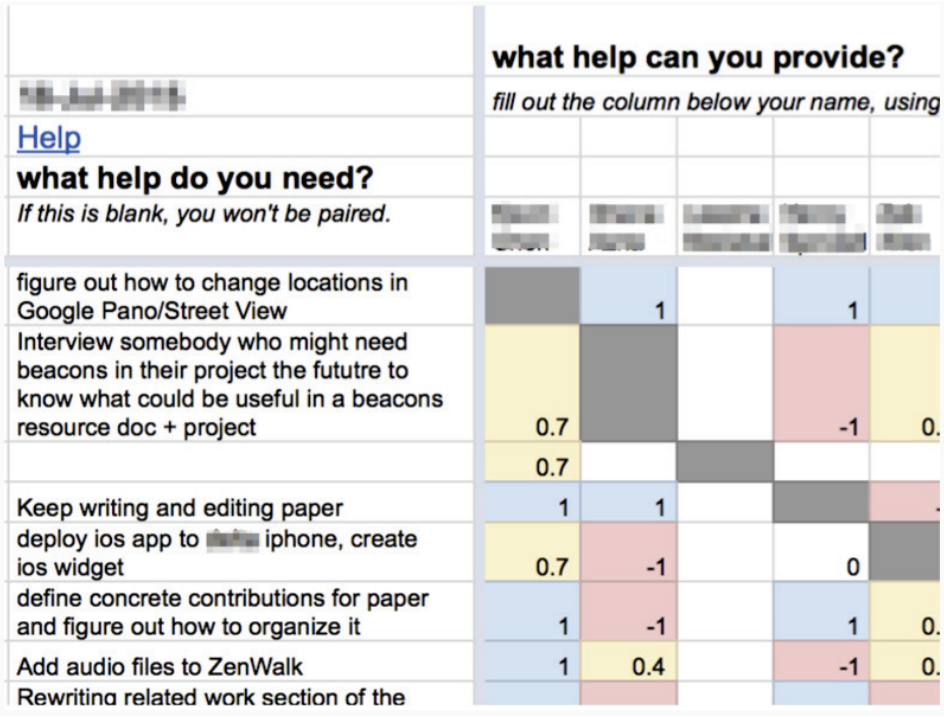
Team	Points Available	Points Committed	D	T	R	Hours Spent	D	T	R	Progress
[redacted]	35	35	12	8	15	19.75	5	7	8	56%
[redacted]	16	19	1	17	2	6	1	6	0	32%
Total	**51**	**54**	**13**	**25**	**17**	**25.75**	**5.75**	**13**	**8**	**48%**

Stories	Tasks for Story	Points Required	D	T	R	Assigned To	Status	Hours Spent	Helpful Links
Have a functional tracking protoype that can track a runner's location and prepare data to be sent to a cheerer	*start entering tasks for this story on the next line ↓*	**17**	*mark*	*mark*	*mark*	*enter your name below to pick up tasks ↓*	*mark as: in progress, backlogged, or done*		
	pseudocode tracking protocol & structs	1		x		Leesha	done		pseudocode doc
	read Swift guide for protocols/syntax	2		x		Leesha	done	2	swift protocol docs
	go through Ray Wenderlich tutorial on POP	2		x		Leesha	backlogged		protocol oriented programming
	implement tracking protocol & structs	3		x		Leesha	in progress	5	
	implement tracking protocol & structs	5		x		Christina			
	Test tracking for cheerer	0.5		x		Christina			
	test tracking for runner	0.5		x		Christina			

Figure 11. Spring Log tool
Screenshot of a spreadsheet prototype of a project team's *sprint log*. The top half of the sprint log provides an overview of commitments, hours spent, and progress on the current sprint. Students plan their sprints in the bottom half by recording high-level deliverables, or *stories*, and the *tasks* for accomplishing those stories. Students use a *point system* to estimate required effort to avoid committing more time than they have available for the sprint. As students make progress they mark tasks as *done*, *backlogged*, or *in progress* and record hours spent. Students also link to useful resources next to stories and tasks.

- **Resources** are references and guides that describe how to achieve commonly shared stories and tasks, such as writing a paper or setting up a technology. While students work on different projects, within an ARS there is likely to be significant overlap in methodology that allow for similar resources and guides to be useful across projects. We curate these resources and provide affordances for students to find helpful resources from their sprint log and to suggest resources to others during SIG meetings.

- **Studio Views** are dashboards that summarize the sprint logs of all teams. This helps surface potential problems to mentors by helping them see at a glance progress across teams. For example, a mentor can see which teams are behind and connect with those teams for a quick check-in or an impromptu office hours to resolve larger challenges.
- **Chat** is a collection of online chat rooms, each with a dedicated topic. Chat provides a medium for students to reach out to mentors and peers for help online and on-demand. We replicate the social structures of Agile Research Studios by providing chat rooms for project teams, SIGs, and the whole studio. We also create rooms to support other community needs such as planning fun activities or discussing interview candidates. Team and studio members can connect through private messaging, and use bots to stay aware of others' activities and facilitate progress updates.
- **Pair Research** [A35] is a system for pairing students to help one another on their respective projects based on their reported task needs and ratings of how well they can help others (see Figure 12). By finding globally optimal matches across a studio, pair research distributes help across the community of learners within studio meetings to make effective use of the community's collective expertise and time resources. This in turn frees up faculty and graduate student mentors to provide help where they are most needed.



	what help can you provide?				
[redacted]	*fill out the column below your name, using*				
Help					
what help do you need?					
If this is blank, you won't be paired.	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
figure out how to change locations in Google Pano/Street View		1		1	
Interview somebody who might need beacons in their project the fututre to know what could be useful in a beacons resource doc + project	0.7			-1	0.
	0.7				
Keep writing and editing paper	1	1			
deploy ios app to [redacted] iphone, create ios widget	0.7	-1		0	
define concrete contributions for paper and figure out how to organize it	1	-1		1	0.
Add audio files to ZenWalk	1	0.4		-1	0.
Rewriting related work section of the					

Figure 12. Pair Research tool
Screenshot of a Google Spreadsheet prototype of the *Pair Research* tool. Students enter their needs for help and how well they can help others. Based on collected preferences, the system automatically pairs students to help each other on diverse research needs.

- **Research Logs** are personal diaries in which students are prompted to capture and reflect on their research progress and learning [76]. The research log serves primarily as a research diary rather than a tool for communicating plans.
- **Independent Development Plans (IDP)** are a set of close- and open-ended questions about project progress, including what challenges students encountered, how they overcame challenges, what challenges remain, and perception of personal growth. IDPs

help students self-assess their learning and progress. Questions are divided into sections on Research Work, Collaboration, Growth, Research Process, and ARS Process to promote students reflecting on different facets of their development and work progress.

We created initial prototypes of the studio tools described above by using, building on, and integrating existing and free collaboration and communication tools including collaborative editors and team messaging programs. Figure 11 shows a Sprint Log we prototyped using Google Spreadsheets that supports student teams planning their work, sharing their work progress, and linking to helpful resources in Google Drive and from the Web. A spreadsheet-based Studio View prototype loads student teams' sprint log data and displays to mentors how far along teams were on their sprint versus how much time was left in the sprint. We use Slack as our chat program and created a spreadsheet prototype for Pair Research (see Figure 12). Students maintain research logs using Google Docs and store their project files in personal cubbies in Google Drive folders. The IDP is implemented as a Google Spreadsheet.

2.4.4 Community Participation, Project Selection, and Social Norms

With more demand than available mentoring resources, we use structured interviews following industry established best practices [12] to select students who are prepared to make research contributions within a research learning community. Similar to advanced classes and labs, students joining an ARS are required to have the requisite subject area expertise.

Beyond subject area expertise, we select for students who show aptitude for embracing collaboration, growing through challenges and failures, taking initiative, managing ambiguity, and seeking deep understanding [86]. Given limited interviewing resources, students in an ARS interview candidates together with the faculty mentor and recommend other students who they think are good candidates for joining the ARS.

To promote student engagement and ownership of their project, students select the project they work on in the SIGs that they are interested in. The faculty mentor meets with incoming students to brainstorm project ideas and discuss project fit and the technical training necessary. The mentor helps students identify a set of scoped project ideas that are feasible for the student and are likely to lead to research contributions of interest to the SIGs the student is interested in. Students have the final say on which project they choose from this set. Students may change projects when their project comes to a close or their interests change; the faculty mentor also supports students selecting a new project in such cases.

An ARS may establish a variety of social norms to promote a culture of collaboration and to help students build regulation skills. For example, to promote effective help-seeking, students may be frequently reminded by mentors to ask for help if they are stuck for more than 30 minutes. To emphasize the importance of learning regulation skills, students may receive positive feedback from mentors for trying to regulate (e.g., exhibiting understanding of their planning processes, or trying to clarify goals and decide among alternative plans) rather than praise for students'

successful work outcomes. This is consistent with helping students develop a growth mindset [32] for learning and practicing regulation skills.

2.5 ARS Pilot Study

We take a design-based research (DBR) approach to develop the ARS model for overcoming the orchestration challenges of research training. DBR [20, 15, 21, 68, 34] requires iterative cycles of defining design arguments (hypotheses), implementing the design argument, collecting data, evaluating the design, and refining the design. We study the ARS model through a case study of the Design, Technology, Research (DTR) program, a research learning environment of 20+ students implemented and led by a faculty advisor.

2.5.1 Context: Design, Technology, and Research (DTR)

DTR was implemented at Northwestern University in Spring 2014. The goal of DTR is to realize and develop undergraduate and graduate students' potential for developing novel technologies and creative solutions through design, engineering, and research. Such practice is often lacking in lecture-based classrooms yet is crucial for preparing students for the complex social and technical problems they will face in STEM careers [44]. As participants in this for-credit program, students lead research projects in social and crowd computing, cyberlearning, and human computer interaction.

In repeated 10 week-long studio sessions, students work with mentors to identify a research project, explore and iterate over designs, prototype at varying fidelities, build working systems, conduct evaluative studies, and report findings through progress reports, blog posts, workshops, and conference publications. Students follow the entire research process each session; projects grow in complexity and generalizability over multiple studio sessions.

Students learn about DTR through word of mouth, recruitment emails, and course catalogs. They apply to join the DTR program and receive course credit for enrollment. Students can enroll repeatedly to satisfy computer science major course requirements or as an independent project course. Students are not obligated to stay in the program, though the majority of undergraduate DTR students continue until they graduate.

2.5.2 Design Iterations

Over a span of two years from Spring 2014 to Winter 2016, we iterated on the structure, practices, and tools of the DTR studio to arrive at the design arguments for Agile Research Studios we presented in the previous section. The DTR studio grew over time; our initial studio had 8 undergraduate students; our current studio has 18 undergraduate students, 1 post-bac researcher, and 2 second-year PhD students. While the faculty mentor initially had 30 minute 1-on-1 meetings each week with each student and followed up via frequent office hours in-person and online, this quickly became impossible to orchestrate as the studio grew. This led us to adopt the SIG structures we proposed. In SIG meetings, we did not initially have a formal

sprint planning process and instead used research logs to communicate progress and plan next steps. But the time required to get grounding took up valuable meeting time and did not scale as SIGs grew in size. This inspired the use of agile methodologies for planning and tracking sprints. But at first, students planned sprints but did not record their progress; this made it difficult for communicating progress and for identifying where help is most needed. Now, students log progress and hours in their sprint log, which facilitates teams sharing their progress and blockers.

2.5.3 Data Collection and Analysis

To study the participation, productivity, and learning outcomes from the ARS model we collected records of students' enrollment in DTR, tallied major design and research products produced, and analyzed student quarterly self-assessments. We deductively coded the self-assessment reports for students' perceived development of regulation skills (i.e. in planning and help-seeking) and shifts in self-efficacy, attitudes, and dispositions.

To study how students planned and replanned their work, we used the Google Drive API to collect the revision histories on each project's sprint log over the two completed quarters during which the tool was deployed (Fall 2015 & Winter 2016). Each revision represents a set of edits grouped into a short time period. We also recorded the final status of each sprint, including points committed, hours spent, tasks completed or backlogged, and resources linked.

To study students' helping behaviors, we surveyed students for the names of people they helped and were helped by in DTR each quarter, as well as what they helped with. This survey was

included as part of the self-assessments in the last four quarters (Winter 2015, Spring 2015, Fall 2015, Winter 2016). We coded each instance of help by the type(s) of help requested and fulfilled, divided into subcategories within design, technology, and research. To understand how helping behaviors may be affected by social structures in DTR, in our analysis we combined the helping data with a graph of the closest relationship between each pair of students as teammates, SIG-mates, or studio-mates each quarter.

To understand how students connected online, we collected Slack usage statistics and public message histories during the six-month period from November 7th, 2015 to May 5th, 2016. While Slack's team statistics webpage only surfaces basic stats on the front-end, it contains within its source code a JSON data object with a more detailed breakdown of the number of messages sent in the channels and groups that the authors are a part of, which we analyzed.

2.6 ARS Findings

Our pilot study results provide early indications of the potential effectiveness of the ARS model for providing authentic research training to increased numbers of students.

2.6.1 Conducting and planning research

Figure 13 summarizes how ARS aims to address the orchestration challenges for learning to plan research. We present results on each of the points below.

Common Obstacles	Student Before	ARS Approach	Student After
Driving all research steps Students aren't involved in all phases of authentic research (e.g. perform 1 technical piece)	Less experienced students relegated to less central or rote research tasks	mentors use *agile methodologies* to slice project work to fit student competencies	More students engage in all phases of research activities
Practice planning Students have limited planning skills, so mentors with many students take responsibility of planning or tolerate projects going off track	Mentors plans projects and drive direction, students engage in few explicit planning activities	Use agile planning methods and tools (e.g. *sprint logs, resources*) to scaffold planning	Students explicitly responsible for research planning
Coaching planning Scarce mentoring resources used to resolve technical challenges, rather than teach effective planning	Students may develop subject area expertise, but don't learn to plan research work on their own	Mentors provide plan feedback in *SIG meetings; research logs* and *IDPs* promote regular reflection	Students develop regulation skills to plan research work on their own

Figure 13. Planning in ARS
ARS addresses the orchestration challenges for learning to plan research by adapting agile processes to research training so students are responsible for planning research inquiry and learn to plan more effectively over time.

2.6.1.1 Driving all research steps to produce research outcomes

Our pilot of DTR used the ARS model to provide authentic research training to a large number of students, produce multiple research outcomes, and sustain participation over time. Over the last two years from Spring 2014 to Winter 2016, we hosted 6 academic-year, quarter-long studios with 4 graduate students (3M, 1F) and 32 undergraduate students (22M, 10F). In Winter 2016, we hosted 21 students (2 PhD, 1 post-bac, and 18 undergraduate) who led 13 research projects.

Engaging in all phases of research, DTR students iteratively designed, built, user tested, and reported on 18 new systems. 19 students have received university undergraduate research grants. 6 student-led papers and 3 extended abstracts have been accepted at ACM and AAAI conferences. Further, three students placed 1st, 2nd, and 3rd at ACM CHI and Grace Hopper student research competitions.

Students followed the agile research process each quarter and grew their projects in complexity and generalizability through sustained participation. While students were not obligated to stay in DTR beyond a quarter, 94% of students completed at least two quarters of DTR (34 out of 36 students). Students who have graduated stayed in DTR for an average of 3 quarters; 64% of them continued in DTR until they had graduated (14 out of 22 students). 12 out of the 36 students left DTR before graduating. They cited a number of reasons, including being more interested in building technology than the research work (4), needing to finish course requirements to graduate on-time (4), switching to another research area and lab (2), and having completed a PhD rotation (2).

In their reflections, students reported learning a wide range of research skills that span authentic research practice. For example, students discussed framing a problem, understanding related research, designing user studies and experiments, preparing an IRB application, conducting user studies, interviewing and surveying, analyzing results, writing an academic paper, and presenting work to an academic and general audience. Students also reported gaining a better understanding of what it means to self-direct research, developing mindsets of “completely owning [their]

project,'' learning to lead research inquiry, and viewing themselves more as "a contributor rather than just a student.'' Through preparing papers, presentations, and research grants, students reported developing effective communication skills and noted the importance of learning to communicate their research effectively to a scientific audience. Students also reported that writing and presenting helped them gain a better understanding of the significance of their work, which contributed to their longer term vision and understanding of their research questions.

2.6.1.2 Practicing planning

DTR students explicitly engaged in planning activities by updating their sprint logs regularly. In Fall 2015 when sprint logs were first introduced, student teams made an average of 4.4 revisions per week (515 revisions total). By Winter 2016, students made an average of 7.3 revisions per week (857 revisions total). Figure 14 shows that teams in Winter 2016 made edits to their sprint log throughout the week, and almost all teams made at least one edit each week.

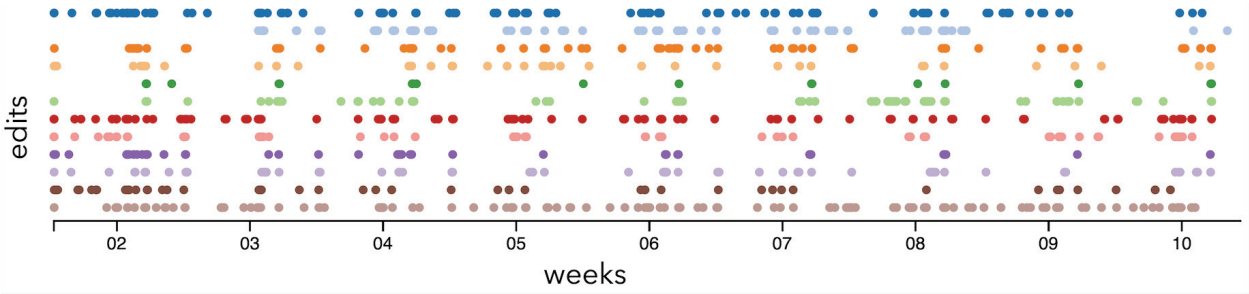


Figure 14. Distributed Sprint Log Edits
Visualization of revisions to each team's sprint log across the weeks of the Winter 2016 DTR studio. Each row of colored dots presents the edits made by a team. Students planned a shortened `sprint 0' in the first week; each subsequent 2-week sprint started and ended with SIG meetings scheduled 2-3 days after the start of each week.

Looking more closely at when students made edits, Figure 15 shows that while students made a majority of their edits throughout the week and in particular in the day prior to the SIG meeting, they also regularly made edits during the SIG meeting and immediately after (62% and 35% of the time in Winter 2016, respectively). This is consistent with our observations of students updating their sprint based on feedback and suggestions from peers and mentors during SIG meetings, and of student teams meeting up after SIG meetings (sometimes immediately after) to sync up and replan their sprints to refocus immediate project goals.

Students were fairly accurate in estimating how long it would take to complete their sprints, but did have a general tendency to underestimate. Students' committed points were within 20% of students' reported work times on 70% of sprints (59 out of 84 sprints); students took even longer to finish their commitments 20% of the time (17 out of 84 sprints). To compensate for some

stories taking longer than anticipated, students backlogged 23% of their tasks (468 out of 2,003 tasks) for completing in a future sprint.

		-1D	SIG	+1D	Week
P(≥1 revision)	W16	88%	62%	35%	75%
	F15	72%	57%	24%	54%
# of revisions	W16	2.9	0.7	0.6	3.1
	F15	1.6	0.7	0.4	1.7

Figure 15. Average Sprint Log Edits
The likelihood and average number of sprint log revisions made by teams each week in the day before their SIG meeting (*-1D*), during their SIG meeting (*SIG*), the day following their SIG meeting (*+1D*), and throughout the rest of the week (*Week*).

Students followed a number of other planning practices recommended by faculty and graduate student mentors. Students were advised to stay within the points allotted for each sprint; in only 3 out of 84 sprints did students spend more hours than the recommended point allotment by more than 20%. Students were encouraged to identify resources that may help them complete stories on their sprint; students recorded 120 resources as `helpful links' that they found or were suggested by peers and mentors (5.5 per team per quarter).

2.6.1.3 Coaching planning

From self-assessments, students reported developing planning skills to drive effective research inquiry by using sprint planning to help them break down big tasks into smaller goals, prioritize goals, and to "see what success looked like at every step." Students reported developing a

number of strategies and skills for delivering value within two week sprint cycles, including (a) building at the fidelity appropriate for the current stage of research, (b) prioritizing important features and research questions, (c) sequencing tasks, (d) defining concrete outcomes, and (e) moving on despite uncertainty or imperfect knowledge. Students noted learning the importance of being able to reason about alternatives and to understand tradeoffs. Beyond individual sprints, students also reported learning to perform "careful prior planning with end goals in mind" at the quarter level to ensure that they completed studies and met paper deadlines.

Students reported developing metacognitive strategies for effectively planning and conducting open-ended research. In order to respond effectively to failures and changes in project needs, students recognized the need to be flexible, to re-evaluate their goals frequently, and to pivot as needed. One student notes: "Failure is a huge part of research, and even if it's frustrating and disheartening, it doesn't invalidate your previous results or progress. You can pivot. Some hypotheses will not be proved, and you need to continue to design studies and build technologies to test other hypotheses." Such strategic planning skills are commonly seen in experts, yet rarely adopted by novices [13].

Many students struggled initially with effectively planning their design and technical work around their research contributions. Instead of focusing on stories that yielded clear research value (e.g., plans that test a design argument), students planned and spent too much time on stories around complex features that were less critical to the research. With mentoring, these failures led students to develop more effective strategies over time. Students reported learning to

conduct iterative, small-scale tests to orient their research direction, prioritizing tech features to answer research questions, syncing their hypotheses and study designs to their technology, and avoiding "getting sidetracked by other nifty design features and remember that everything has to be tied back to research outcomes."

Students noted that SIG meetings and follow up discussions with faculty mentors and SIG leads helped them learn to better plan at appropriate fidelities, see the big picture and refocus on higher-level research goals, refine research directions and questions, and to better manage team issues. Students also report that talking to faculty mentors and SIG leads helped them to set appropriate expectations, and to reorient their perspective on research given frustrations and problems. As developing mentors, SIG leads expressed that they felt that they were able to help students drive their research, and expressed a desire to continue learning to more effectively manage young researchers.

2.6.2 Getting help and making progress

Figure 16 summarizes how ARS proposes to address the orchestration challenges for learning to get help. We present results on each of the points below.

Common Obstacles	Student Before	ARS Approach	Student After
Distributing help Students need diverse technical help to make progress, however mentoring resources are limited	Students wait for mentors and do not always feel supported when facing research obstacles	Share the responsibility of helping across the entire community	All students are helping one another and feeling supported by community
Practice help-seeking Even when distributed help is available, students lack skills in accessing help (e.g. identifying helpers and connecting to help)	Students don't receive the help they need to make effective progress	Facilitate connections between students via *pair research, chat, studio views, SIG,* and *studio*	Students have access to diverse pathways of help across the community
Coaching help-seeking Students are reluctant to ask for help and lack help-seeking dispositions	Students are reluctant to ask for help from mentors and peers in the community	Help-seeking is normalized and explicitly trained via scaffolds and community practices	Students develop help-seeking skills and dispositions within the community

Figure 16. Helpseeking in ARS
ARS addresses the orchestration challenges for learning to get help by supporting students (learning to) make effective use of the expertise and resources across the research learning community.

2.6.2.1 Distributed help

DTR students helped over a third of the other students in their studio in any given quarter, and fulfilled 372 help requests over the four quarters in our dataset. See Figure 17 for the helping statistics from each quarter. DTR students were willing to help others working on different projects in their SIG and across the studio; 329 reported help requests (89%) were fulfilled by a student working on a different project. 115 requests were fulfilled by a fellow SIG member working on a different project (31%), and 214 requests were fulfilled by a student in another SIG

(58%). Figure 18 shows a dense network of helping behaviors from the Winter 2016 that connects students both within and across SIGs.

	N	Total	Median	Average	% of studio
Winter 2016	21	160	7	7.6	38%
Fall 2015	14	61	4	4.4	34%
Spring 2015	14	65	5	4.6	36%
Winter 2015	15	86	5	5.7	41%
Total/Average	33	372	5	5.8	37%

Figure 17. Helping Statistics
Helping statistics showing the number of fulfilled help requests reported in different quarters of DTR, the average/median number of people receiving help, and the % of people in the studio receiving help.

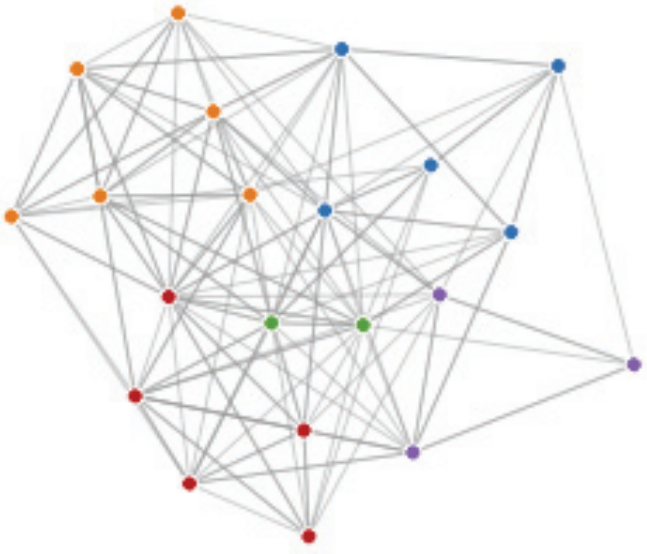


Figure 18. Helping Graph
Helping graph from Winter 2016. Each node represents a student and links represent fulfilled help requests. Students helped and received help from both students in their own SIG (same color node) and across the studio.

Students helped other students who also helped them in return. Of the 329 help requests made by students working on different projects, 196 requests were reciprocated in the same quarter (60%) and 231 requests were reciprocated across quarters (70%). Students also helped other students when they did not receive help in return. Many students expressed wanting to pay it forward, and felt that it was natural to want to help others given all the help they had received in the past. One student said "Nobody feels guilty receiving help because they know they've helped others as well."

Students reported that seeing other people's "love for helping others" in DTR led them to want to cultivate and spread the helping culture, saying they want to both "share [their] experience and utilize others." Students who initially dismissed their own abilities recognized that "there's always something [they] can help someone else with." Students also commented that they enjoyed giving help to others who needed it; they noted that helping others allowed them to "get a much better sense of what everyone else was doing" and usually gave them additional ideas that benefited their own project and learning.

Further, students shared that the way in which help was distributed reinforced feelings of community support. Specifically, students felt that they were in a community of "similarly motivated students who care about each other's work" and felt they could rely on this community in ways that they never had before. One student wrote: "DTR is a class but I do think it is more importantly a collaboration between all students. I have never been this close to other classmates and DTR makes it such a collaborative environment."

2.6.2.2 Scaffolding help-getting

Students received help from community members on a wide variety of requests across design (n=159), technology (n=129), and research (n=118). Figure 19 shows help requests across these areas by category. Popular requests include asking others to test the latest version of a prototype, helping with web/mobile development (e.g., building a Chrome extension or deploying an iOS app); and refining research directions. Help across requests for design, technology, and research were fulfilled by team, SIG, and studio members at about the same proportions. Among major subcategories, there were two instances where significantly more help was provided by one group than another. This includes help on research directions, where 34 of the 41 requests were fulfilled within a SIG (84%), and for prototype testing, where 72 out of 103 requests were fulfilled by studio members across SIGs (70%). These numbers are consistent with ARS structuring SIGs around particular areas of research focus, and show students reaching more testers by recruiting across the studio.

Design help (n=159)		**Technology help (n=129)**		**Research help (n=118)**	
PROTOTYPE TESTING	103	WEB/MOBILE DEV	81	RESEARCH DIRECTION	41
NEEDFINDING	45	SYSTEM FEATURES	20	STUDY DESIGN	33
BRAINSTORMING	15	DEBUGGING	17	PAPER WRITING	32
FEEDBACK	16	PAIR PROGRAMMING	14	DATA ANALYSIS	15
INTERACTION DESIGN	16	ARCHITECTURE	12	GRANTS	14
UI/UX DESIGN	9	GENERAL	10	USER STUDIES	10
RECRUITING	8	DEV TOOLS	7	RELATED WORK	9
PROTOTYPE DESIGN	4	ALGORITHMS	3		

Figure 19. Helping Categories
Help requests fulfilled, broken down by types of help.

Out of all help requests, 60% (223 out of 372) involved at least one out of the six most senior students in the community (3 PhD students, 3 senior undergraduates). The most senior students helped on 41% of help requests; PhD students were disproportionately more likely to help with research (48%) and technology (44%) than design (19%); senior undergraduates were also more likely to help with research (44%). This suggests that beyond a faculty mentor, students were able to access research mentorship via more experienced members of the community. The most senior students received help on 30% of help requests; graduate students disproportionately received help with design (42%) over tech (19%) and research (14%), and senior undergraduates received more help with research (42% of the time). This further suggests that experienced students, while providing significant support in the community, were still receiving significant design and research help from other students.

Students reported that studio meetings helped them learn from other students in the studio and broadened their perspective on their own projects and on research more generally. Students noted that demo-critique sessions helped them brainstorm new ideas, clarify research directions, and gain fresh perspectives. Students benefited from pair research and found it to be a "fantastic way to scale a class where a professor doesn't have time/specific skills to give each student that much individual attention, and when students can help each other." Students also found that the learning modules led by SIG leads and faculty mentors helped stretch their thinking, for example to consider validity concerns when designing a study, architecting apps that can scale to millions of users, and understanding realistic research timelines and the need to plan for failure.

```
Channels     8403 (100%)
  Team       4948 (59%)    project discussion among teammates (and w/ mentors)
  Studio     1131 (13%)    studio-wide announcements and water cooler discussion
  Bot        1063 (13%)    GIT (code) and Youtube integration (sprint videos)
  Community   416 (5%)     swag, senior gifts, 20% time, job search, fun activities
  SIG         305 (4%)     sharing resources and communicating within a SIG
  Faculty     240 (3%)     faculty discussion about agile research studios
  Interview   211 (3%)     committee coordinating and scheduling DTR interviews
  SIG Heads    71 (1%)     SIG head and faculty check-in
  Other        23 (<1%)    occasional use; testing bots
```

Figure 20. Slack Categories
Slack channels and message counts by group and purpose.

Outside of in-person meetings, DTR members connected extensively with one another via Slack. Over a six-month period, DTR members sent a total of 36,919 messages and shared 660 files. Of these messages, 8,407 of them (23%) are in visible public and private channels the authors are a part of; the rest of the messages are direct messages or other private group conversations. Figure 20 shows the number of messages sent through visible channels categorized by a diverse set of purposes, and the members who participate in them.

Individual requests for help can quickly spur others to offer their help, and also to seek help themselves. Figure 21 shows a conversation in the Water Cooler channel with students requesting, offering, and trading help to test their latest prototypes. An initial request for help was responded to within minutes, and spurred a `helping party' in which six students agreed to test for each other across four projects. While help requests don't always spur this degree of Slack activity, we observed that help requests are often responded to within minutes. DTR students generally felt that if they asked for help, they would receive it from the DTR

community. They reported reaching out to people on Slack whenever they were stuck and not being afraid to directly message other people in DTR they didn't know well to ask random questions.

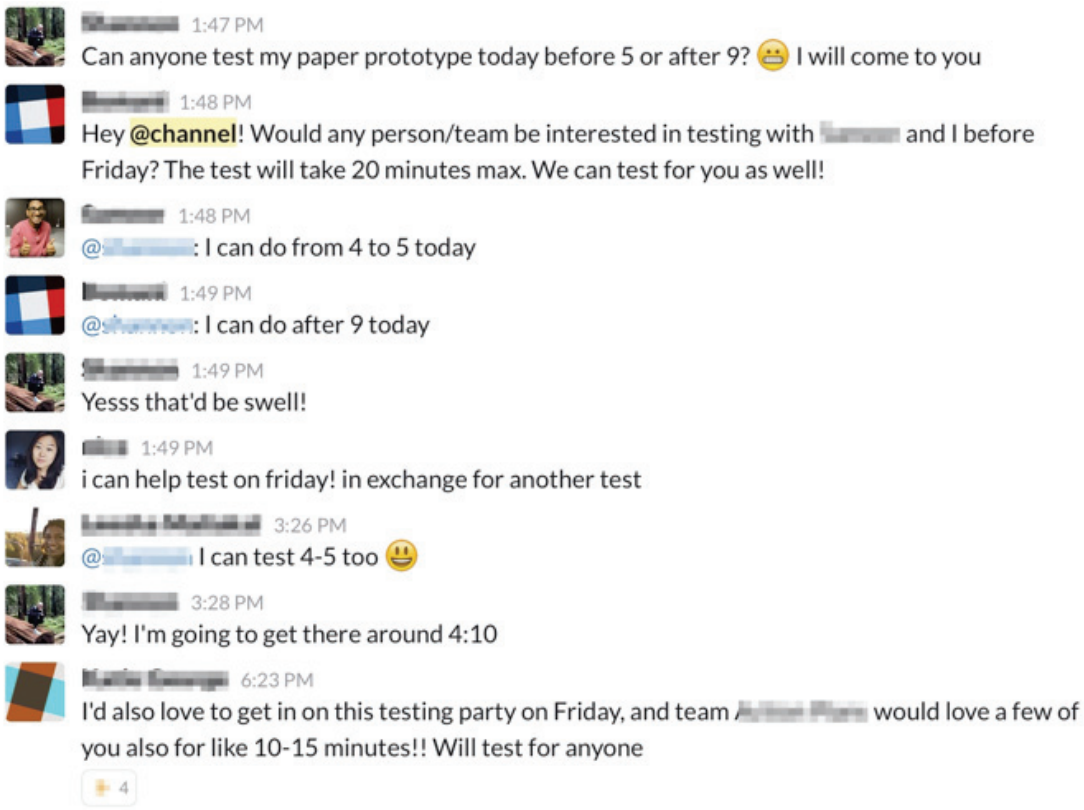


Figure 21. Slack Helping Interactions
A conversation in the Water Cooler with students requesting, offering, and trading help to test their latest prototypes.

2.6.2.3 Learning help-seeking

Students overwhelmingly reported an increase in their willingness to seek out help as they participated in DTR. Students who were initially reluctant to ask for help learned that "[they] can

ask for help and that everyone asks for help and it doesn't make them stupid to need help.'' One student noted that she reached out more easily once she realized that "people in DTR could help me out much more than I could get by myself.'' Students reported adopting a number of effective help-seeking strategies rarely practiced by undergraduate students [62, 67, 45], including doing a little bit on their own to help them formulate help requests better, asking for help online and offline, "expanding [their] network of help-givers and receivers with the new class [of students],'' and learning to become more comfortable asking for help by starting now.

Students recognized the importance of recognizing blockers that impeded their progress and seeking help to overcome them. Through less productive sprints, students learned that it "is detrimental to try to work through blockers on your own. Asking for help should be the first step when you really get stuck on a blocker.'' Students noted their struggles in "admitting that something I'm doing is not working'' and being "adamant in fixing things on my own.'' They commented that DTR "taught me to acknowledge when I need help and that it's perfectly acceptable and important to ask for that help.'' Students also acknowledged the need to ask for help quickly, to not wait till the next SIG meeting and to instead reach out to mentors "more often as that could have prevented the issues I faced or at least I would have discovered earlier.'' These realizations are rare among undergraduate students outside of DTR, yet critical for supporting student learning and progress-making in project-based learning environments [38].

2.6.3 Orchestration Time

With the community structure for planning and support in place, the faculty mentor was able to orchestrate DTR with 2 PhD, 1 post-bac, and 18 undergraduate student researchers leading 13 research projects with less than 12 hours of faculty time each week. The faculty member used: 5 hours for SIG meetings (five SIGs each with up to 3 active projects and 6 students), 3 hours for the all studio meeting, and 4 hours for in-person and virtual help to respond to students on demand. On typical weeks, the faculty mentor was able to maintain awareness of progress across projects and had time to respond to research challenges that they themselves can best address. On "surge weeks," e.g. when approaching paper and grant deadlines, the faculty mentor spent considerably more time reviewing drafts, editing, and helping out in any ways needed.

2.6.4 Building self-efficacy in independent research

Results from the pilot show significant shifts in students' attitudes and beliefs toward their ability to develop novel technologies and conduct research. One student noted that a highlight of DTR was that she actually finished a paper and submitted it to a conference. Another student noted that DTR "pushed [me] outside of my comfort zone in what I could do. [I realized] I was more capable than I thought I was." Students commented that DTR made something daunting do-able: "For a long time, [I] hear people say [it's] ok to get in without knowing everything. [DTR] deeply ingrained that you can get started without having to be experienced."

Students also reported that their DTR experience positively impacted their attitude toward academic research. Students report that despite research being ``much more rigorous than [they] could have guessed,'' they became much more enthusiastic and found ``the opportunity to actually drive new knowledge to be pretty exciting.'' One student wrote that DTR "was a great experience and has drastically shaped how I think about research and big picture problems.'' Students also commented that the experience "opens up career paths" and that "research in grad school seems less intimidating now." These student reflections reinforce the significance of scalable and effective research training, and the long term impact it may have on developing future problem solvers.

2.7 Discussion

In summary, our pilot study of DTR showed that the practices, structures, and technologies that come together as an Agile Research Studio ecosystem empowered undergraduate students to plan research work at weekly intervals and overcome challenges quickly with the support of peers and mentors. This allowed them to conduct independent research along a faculty member's core research directions, as would be possible through dedicated 1-on-1 apprenticeship with faculty members but at just a fraction of the time required to support a much larger research learning community than would be traditionally feasible.

Below, we discuss how the ARS model can impact research training beyond DTR, the limitations of our study, and future work in learning and orchestration technologies.

2.7.1 Implications for Research Training

DTR pilot results suggest that the ARS ecosystem can scale effective research training. By distributing the responsibility of research planning across supports, and connecting students to teammates, SIG members, and the rest of the studio for help, faculty mentors have more capacity to focus on coaching regulation skills and can be more responsive to the challenges they are uniquely positioned to address. As students develop regulation skills, they become less reliant on the mentor and increasingly self-directed; this further scales mentoring resources and allows them to more skillfully orchestrate efforts based on research importance and student needs.

DTR pilot results suggest three ways in which the ARS subsystems and supports enable students to practice research and produce research outcomes. First, by engaging students in planning and providing plan feedback, students learn to deliver research value with each sprint, prioritize research goals, and avoid spending time on less crucial tasks. They also (learn to) catch problems earlier and flexibly replan with the support of mentors and peers. Second, by promoting helping behaviors, students are able to receive the help they need to overcome blockers and make regular progress. Third, by scaling mentoring resources, ARS significantly expands the number of student-led projects producing research outcomes.

Students in an ARS are responsible for not only their own learning and progress, but that of other members of their studio. Establishing a supportive community in which students take on the responsibility of helping one another is crucial for an ARS to exhibit the outcomes observed in DTR. By using practices like pair research that enable direct reciprocity, and also scaffolding and

normalizing help-seeking and help-giving more generally, ARS involves growing a supportive community over time in which generalized reciprocity is commonly practiced [6].

While the ARS ecosystem allowed DTR to scale a research learning community with over 20 members, it may also benefit faculty members who run typically-sized research labs (e.g., 5--10 students). Training students through the ARS may still be worthwhile for advancing research productivity over time as long as the mentor's investment in developing regulation skills pays. Despite having fewer helpers with possibly less diverse expertise, ARS support for help-seeking and help-giving still scales mentoring resources in ways that promote learning and productivity. While the benefits may be somewhat smaller for smaller labs, in these ways ARS can still provide significant benefits and also pave the way to lower participation barriers to include undergraduate students and train more students.

The ARS ecosystem may also be useful for supporting a community much larger than DTR by further distributing faculty responsibilities. One challenge is ensuring that there are enough mentoring resources to support regulation skill development. Communities can scale up in size as graduate students' develop their mentoring ability and become more ready to lead their own SIG; this allows faculty mentors to fade from more established SIGs to start new ones. Another challenge is connecting students to help across a larger community. This imposes additional orchestration burdens, but also challenges in establishing a helping culture should students feel less connected. New orchestration technologies and effective community designs thus become increasingly important as the community expands in size.

ARS provides a powerful example of how socio-technical learning ecosystems can be designed to support cooperative work in ways that allow us to provide research experiences to more students. Support for mentoring self-directed learners, leveraging distributed expertise, and promoting awareness of needs reduce orchestration burdens to scale mentor time and advance work outcomes. We believe socio-technical solutions for supporting cooperative work will play an increasingly important role in empowering us to provide authentic learning experiences to many more students to prepare them for tackling complex challenges in the 21st century [86].

2.7.2 Study Limitations

Our exploratory study has limitations including: measurements of learning, author as researcher and designer, participant selection bias, and isolation of factors.

2.7.2.1 Measurements of Learning

While we used student self-assessments to surface students' perceptions of their learning and growth, we did not directly measure learning and growth. Self-assessments may be biased as students may understate or overstate their learning and skills. To complement these initial measures, we also collected and analyzed traces of student interactions with one another and with the studio tools; this provided evidence that students were following regulatory processes and being mentored to improve their regulatory processes. Prior research shows that the practice and mentoring of regulation skills helped students develop them in a number of science domains [3]; future work can attempt to measure such gains directly in DTR. Consistent with

design-research best practices [5, 34], we use these measures to quickly identify failures in the design and to iterate, with plans to more directly measure learning as the design develops.

2.7.2.2 Author as Researcher and Designer

Design-based research allows researchers the control to simultaneously iterate on and study complex models [A3, A22]. However, this introduces the possibility of bias as one of the researchers was also the faculty member in the design. To limit the risks of biasing results, we centered our analysis primarily on student self-reports and log data. While student self-reports may still be biased in favor of the desired learning outcomes, students were also forthcoming about their struggles with regulation. Beyond biasing results, designing for our own lab runs the risk of creating solutions that don't work well in other settings. To mitigate these risks we have focused on common orchestration challenges informed by the learning literature and designed tools that are largely domain-agnostic (e.g., pair research, sprint logs). But even so, some adoption challenges remain. As one example, while agile methodologies work naturally with the design- and technology-centered research work in DTR, adopting such methods to other fields of study may require new ways of working that are less familiar. In future work we are interested in supporting other research communities adopting the ARS model and studying its effectiveness and any adoption challenges across domains.

2.7.2.3 Participant Selection Bias

This paper argues that ARS can significantly scale mentor time and increase research productivity all while training undergraduate researchers. It is important to note that the ARS model specifically includes screening undergraduates who have the technical skills and interests required to conduct undergraduate research. This raises a concern about the following counterfactual: perhaps students selected for DTR are more qualified than those who typically engage in research, and that their learning and research accomplishments are attributed solely to selection. However, anecdotal comparisons suggest this to be unlikely. First, traditional 1-1 apprenticeship models rely on an even more stringent selection criteria, selecting the best applicants for graduate school and then selecting admitted graduates based on their fit for the lab. ARS lowers the floor for research by widening the pool to qualified students to include many undergraduate sophomores. Second, compared to an honors thesis course at the same university, which has a similar selection requirement and can involve 30+ hours per week, students publish infrequently if at all. Of course, these comparisons are only anecdotal--future work must rigorously measure how selection effects influence ARS outcomes relative to other programs.

2.7.2.4 Isolation of Factors

While our analyses provide evidence that the ARS model led to the observed outcomes, one study is not sufficient to isolate all the factors necessary to DTR's efficacy. As one example, while we observed that DTR students were eager to help others, we cannot say at this point that adopting the ARS model as described is sufficient for creating a strong community culture in

which students are as willing to help as DTR students were. We look forward to refining the ARS model over time as we continue to develop our understanding of other design considerations that may be critical to a studio's success.

2.7.3 Future Work in Learning and Orchestration Technologies

Advancing socio-technical ecosystems for scaling effective research training can help to (a) scale mentor time, (b) produce effective research, and (c) engage more students to self-direct independent research. Our pilot results suggest that ARS subsystems are already helping to support these goals by engaging students in planning and helping activities, surfacing needs for help-seeking and re-planning, and connecting students to help and instruction. A core focus in future work is to advance ecosystems that scaffold students to reflect on and improve their learning when students practice on their own, outside of in-person meetings, and to become aware of needs for mentoring and support. For planning, learners still lack scaffolds for planning effectively on their own and can struggle to reprioritize tasks to deliver research value and fail to recognize their needs for help. For help-seeking, while students use Slack extensively outside of in-person meetings, they still struggle with formulating help-requests, especially for more open-ended help on topics such as framing design arguments or refining research directions. Related, students can benefit from tools that provide additional scaffolds for connecting to available mentors and peers on-demand. Advancing project-based learning platforms capable of monitoring learning activities and progress across the studio and triggering and scaffolding help and instruction can play a significant role in advancing research training at scale.

Chapter 3. Polaris

A System to Scaffold Risk Assessment in Design Arguments

While pilot testing of the Agile Research Studios demonstrated that a learning ecosystems approach can enable scalable and effective research training, I recognized critical gaps in how we trained core self-direction skills, such as learning to plan. Students engaged in high level planning practices as they leveraged the planning subsystem, like breaking down their plans into 2-week sprints, and bringing their plans into SIG meeting to receive planning feedback. However, our needfinding exposed practical breakdowns in how students identified what they should focus on in their next iteration. My exploration of literature demonstrated that expert design-researchers focus their iteration plans on addressing the most critical risks in their design rationale. These findings correlated with what we observed practically in coaching interactions between ARS students and their mentors, where mentors spent significant time guiding students to critically evaluate the underlying rationale for their design approach. While the planning subsystem provided supports such as the Sprint Log for breaking down tasks, and SIG meetings for receiving feedback on their plans, the subsystem lacked scaffolding for two key expert practices -- visualizing the design problem, and assessing that structure for critical risks (see Figure 22). In Chapter 3, we describe how we extend the ARS planning subsystem with *Polaris*, a system that scaffolds students to construct and evaluate design arguments for critical risks, and discuss how mentors can use this information to guide their coaching interactions.

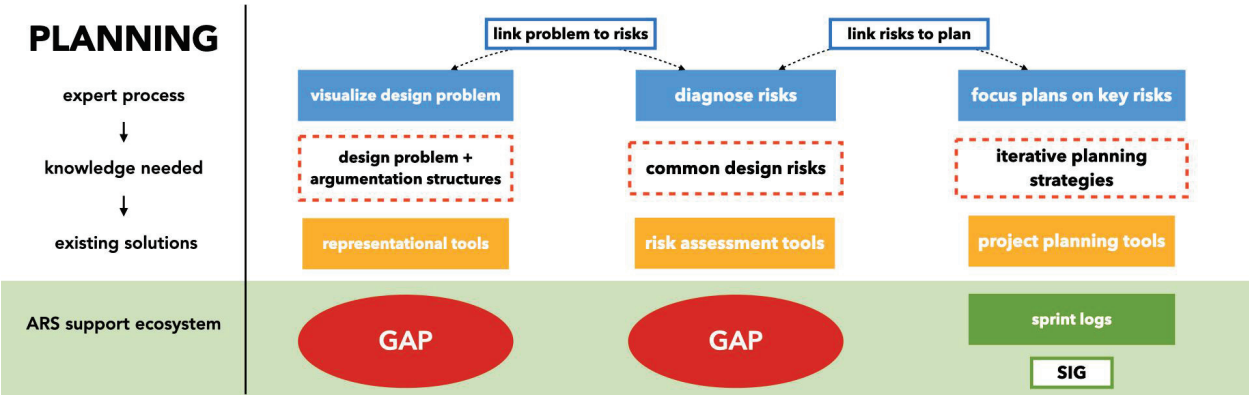


Figure 22. Critical Gaps in the ARS Planning Subsystem
While the existing ARS subsystems scaffolded students in some aspects of expert practice, we found critical gaps where our existing supports lacked scaffolding to support other aspects of how experts work. For instance, ARS scaffolded students to construct iteration plans via the Sprint Log tool and receive feedback on planning strategies in SIG meetings. However, the planning subsystem still had gaps -- namely, scaffolding students to visualize design problem structures and diagnose them for risks as experts do.

3.1 Introduction

It is increasingly important to train the future workforce in the skills they need to design and research solutions to today's most pressing problems. In addition to domain skills (i.e. prototyping techniques and research methodologies), people must also develop core metacognitive skills, like learning to strategically plan out their work. Expert design-researchers iteratively plan [18, 19, 26, 27, 37, 43, 57]. With each iteration, experts identify critical issues in their *design argumentation* -- or the rationale behind their design [35, 40, 43, 77, 87] -- and focus their next steps on mitigating such issues. In contrast, novice designers often struggle to focus on critical issues in their design argumentation when they evaluate their designs, limiting their ability to plan effective design iterations [27].

Literature and our own needfinding suggest that novices struggle to identify critical issues because they lack the structural knowledge and diagnostic practices that expert designers use to evaluate designs [18, 19, 26, 28, 43, 57]. Novices often focus on surface-level issues (e.g. improving the aesthetic of a website), relying on experienced mentors to identify deeper structural issues in their design argumentation (e.g. justifying how the content of the website is serving a user need) [18, 19, 26, 28, 71]. While existing pedagogy is adept at teaching design skills and processes such as storyboarding and design sprints, it often fails to train novices on how to identify and address structural issues in their design argumentation [28, 56, 57, 73, 90]. Expert design mentors are highly effective at coaching novices to recognize and address such issues. However, coaching is time-intensive and such experts are often limited resources in design learning communities [72, 73, 81, 90]. As a consequence, mentors often scaffold past this critical part of iterative planning, leaving students with insufficient coaching to meaningfully improve their designs.

To address this gap, we extend the ARS planning subsystem with ***Polaris***, a learner-centered diagnostic tool that scaffolds novices to construct and evaluate their design argumentation for key structural risks (see Figure 24). Polaris embeds expert structural knowledge and diagnostic practice into a *design argument template* and *reflection prompts* that focus a novice's attention on three types of issues: gaps (i.e. are there components of the argument that are missing?), lack of depth (i.e. what is the depth required for each component to form a convincing argument?) and misalignment (i.e. what are the links required between and across components to form a cohesive

argument?). The design argument template (Figure 24, left) structures the four components of a design argument: outcomes, obstacles, characteristics, and arguments, and visualizes the relationships between them via conceptual links. In contrast to existing tools that primarily help novices represent the components of a design problem (e.g. breaking it down into a problem statement and a solution space) [2, 8, 9, 10, 30, 45, 46, 48, 64, 65, 71], the design argument template visually focuses a novice on the relationship *between and across* components, allowing them to more easily detect structural misalignments (e.g., a proposed characteristic does not address the specific obstacle that it is linked to). To help novices focus on specific structural issues that design experts and mentors often catch, Polaris provides reflection prompts (Figure 24, right) on lack of depth and misalignment issues for each component of a design argument (e.g., "Does the obstacle clearly describe how it prevents the linked outcome from being reached?"). This helps novices procedurally focus their evaluation on identifying core structural issues within and across the components of their design argument, and to avoid fixating on surface issues.

We hypothesized that Polaris' templates and prompts can help novice designers identify critical issues in their design arguments, and can inform how mentors coach students to devise plans to address them. To test this hypothesis, we conducted a pilot study with 13 undergraduates leading 10 independent research projects, where students used Polaris to construct and evaluate their design argumentation. Of the 91 issues that students identified with Polaris, 95.6% of the issues focused on core design argument components, and 96.7% of all issues focused on three core

types of structural issues: gaps, a lack of depth, or misalignment in their design argumentation. To explore the quality of the issues from the perspective of a mentor, we asked mentors to generate their own issues and to evaluate the issues generated by 6 of the 10 teams (71 issues reviewed). Findings show that on average, mentors rated the issues that students identified as accurate (4.06/5) and severe (4.11/5). When comparing mentor and student issues, findings show that students identified 55.17% of the critical issues identified by mentors. Moreover, mentors felt that student issues expressed a student's current understanding, and informed how mentors would coach students to overcome misunderstandings. These findings demonstrate that diagnostic tools such as Polaris can scaffold novices to identify structural issues in their design argumentation and to focus their plans on mitigating the most critical risks in their designs. These findings further highlight a need to critically evaluate ecosystems for gaps in how they support the skills they are designed to train, and to extend ecosystem designs to integrate new supports and learning interactions to better scaffold expert practice.

3.2 Background

Our work explores how to scaffold novice designers to identify critical structural issues in their design argumentation, on which to focus their plans. In order to understand how we might design such scaffolds, we first review how experts use design argumentation to iteratively improve their designs, and why novices struggle. We then review limitations to training design argumentation

in current pedagogy, and the limitations of existing tools for scaffolding novices to construct and evaluate their design argumentation.

3.2.1 How Experts Use Design Argumentation

Experts evaluate and improve their designs by identifying issues in their *design argumentation*, or the underlying rationale that justifies their designs [35, 40, 43, 77, 87]. Experts use design argumentation to (1) build up an understanding of the problem space through problem representations, (2) articulate detailed and novel solutions to address those problems, and (3) construct arguments for why they believe the solution will address the underlying problems [35, 43, 87]. Prior literature underlines the importance of using design argumentation to plan out practical next steps in design work. To plan out iterations that advance designs in impactful ways, expert designers focus on evaluating and improving their underlying design arguments with each iteration [40, 77].

However, novices struggle to evaluate their designs through design arguments because they lack the structural knowledge and diagnostic practice that experts have [18, 19, 43, 57]. Unlike experts, who "focus on problematic areas" of the design, novice designers have "unfocused, non-analytical, and diffused" ways of evaluating when troubleshooting their designs and planning next steps [27]. Recent work suggests that novices struggle to diagnose their designs because they lack knowledge of (a) design problem structure to focus attention on key components, and (b) potential issues to diagnose different risks in the design [19]. For instance, when describing their user's obstacle, a novice might claim that "existing solutions don't have

this feature that our design has" rather than explaining *why* existing solutions don't work. Here, novices fail to recognize a gap in their argumentation -- understand why a user is actually struggling -- and instead jump to their proposed design. Similarly, a novice may state an imprecise desired outcome because they don't yet recognize the need to have measurable outcomes when testing their designs. Without this knowledge of design problem structure and potential risks, novice designers will struggle to evaluate their design arguments as experts do.

To train students to evaluate their design argumentation for the critical risks that drive meaningful iterations, we must first identify the types of critical issues experts find in design argumentation. We can then embed these expert models into effective scaffolds for novices. To address this need, our work characterizes three types of structural issues in design argumentation that experts often detect (gaps, lack of depth, and misalignment), and contributes a tool that scaffolds novices to focus on such issues when evaluating their designs.

3.2.2 Pedagogy Limitations to Training Design Argumentation

Current design education trains a wide range of design skills that make up expert practice. For instance, design practitioners and educators introduce novice designers to common design processes and practice [27, 57], such as needfinding, storyboarding, and prototyping. More recently, agile methodologies have been adopted in design practice (e.g., design sprints) to help innovators incrementally plan and monitor their progress [56, 73, 90].

However, existing pedagogy often fails to train novices on the design argumentation that is the backbone of effective design. For example, a novice designer may use a combination of user

stories and prototyping techniques to develop a low-fidelity design that they believe will realize the user story. However, when a novice fails to argue for how the design addresses a root challenge that is preventing the user story from being realized, this opens up the possibility for a failed design solution that lacks justification. Similarly, a novice designer may use methods like retrospectives and sprint logs to identify issues and plan next steps in their project, but may still focus on surface-level issues (e.g. improving the UI color scheme of an informational election voting website) rather than on core issues that an expert might notice (e.g. including content that voters need when visiting the site). Novices can improve aspects of their designs by employing such design process and practice. However, novices that struggle to construct and evaluate design arguments may continue to generate designs that lack mechanistic explanations, and are unlikely to be effective as a result [27].

While design experts can effectively coach students to focus on critical issues in design argumentation, such experts are often a limited resource in design learning communities [72, 73, 81, 90]. Prior work explores interventions to support mentors in coaching novice designers to identify critical issues in their projects and planning effective next steps to address them, for example by using diagnostic questioning to understand critical issues in students' research projects [90] and developing tools that surface a design team's plans to design coaches to facilitate feedback [72, 73, 81]. While effective, coaching interventions remain mentor-intensive, and mentors' time is still a limited resource in many design learning communities [73, 90]. Consequently, we've observed that when mentors spend significant time diagnosing issues

during a coaching session, it can leave little time to coach students on effective planning strategies and skills they can use to address them.

3.2.3 Existing Tools to Scaffold Design Argumentation

Prior research has extensively explored how to scaffold novices to visualize and evaluate different parts of their design problem. For instance, many representational tools help novices visualize general problem and solution components that make up the structure of a design problem [2, 30, 37, 45, 46, 48]. More recently, business and design practitioners have adopted *canvases* that help break down the different components of a design problem, such as community partners, users, and value propositions [56, 64, 65, 71]. Other researchers have focused on designing diagnostic tools that provide novices with a set of heuristics and a process to help them evaluate their designs for common risks [18, 19, 26, 27, 42, 43, 71], such as diagnosing the design of a parachute using heuristics like whether it has a vent at the top.

However, even with such scaffolding, novices still struggle to identify the deeper structural issues in their design argumentation that experts focus on most. While representational tools like canvases can externalize components of design problem structure (e.g. a problem vs a solution), they often fail to represent deeper structure, such as the way in which experts conceptually link these components to form aligned design arguments. Similarly, while diagnostic tools can externalize heuristics to help novices evaluate a design for common risks, existing tools either fail to capture the generalizable depth of argument that experts think about [26, 27], or provide so many heuristics that novices overlook critical issues in their evaluation [18, 19, 71]. For

instance, heuristics for parachutes may be specific to conceptual knowledge about parachute design (e.g. the role of a vent), rather than highlighting general, structural issues in design argumentation (e.g. the measurability of a desired outcome). Without representing or prompting novices to reflect on the deeper structure of their design argumentation, novices can fail to learn the ways that experts identify more generalizable issues in their design -- issues that have less to do with the domain knowledge of a problem, and more to do with general design expertise. Without this expert practice, novices focus instead on the surface-level issues that appear most salient to them.

While there are some expert systems for supporting design argumentation, these systems are limited in their usefulness, usability, and generalizability for novice designers. A review of these systems [77] found that they over-focus on argumentation logic, which distracts even experts from planning practical design implementations, thereby limiting the practical usefulness of such tools. Moreover, studies show that even expert designers face cognitive constraints in navigating such complex logic representations, suggesting significant limits to usability and usefulness of such systems for novice designers. In another vein, there have been some systems designed to scaffold novices to evaluate and improve their design argumentation for specific design problems such as kitchen design [40]. While studies show that such tools are usable and useful for novices, these interventions focus on conceptual issues that are domain-specific, rather than generalizable, structural issues in their design arguments.

Filling this gap, **Polaris embeds the structural knowledge and diagnostic practice of experts into generalizable templates and prompts that scaffold novices to evaluate their design arguments for critical issues**. Unlike prior approaches, Polaris scaffolds novices to focus their attention on the types of structural issues that experts generally find in design argumentation, across domains: gaps (i.e. are there components of the argument that are missing?), lack of depth (i.e. what is the depth required for each component to form a convincing argument?) and misalignment (i.e. what are the links required between and across components to form a cohesive argument?).

3.3 Challenges in Identifying Project Risks and Focusing Plans

We first sought to understand how expert design mentors coach students to identify critical structural issues in their design argumentation. We observed planning meetings in an Agile Research Studio [90] led by one of the authors and another early career faculty mentor. In the studio setting, undergraduate students learn to lead systems HCI research with the support of faculty and graduate student mentors. For example, a student may design, implement, and study developer tools that help novice web developers make sense of real-world code. Students had been enrolled in the class and working on their projects between 1 and 5 quarters (3 to 15 months). As part of the studio structure, faculty and graduate student mentors coach students on their planning during weekly planning meetings. In this context, we conducted field observations across 25 student-mentor planning discussions in 10 planning meetings over the course of 5

weeks. A typical meeting lasts 1 hour, during which 2-3 different undergraduate research teams take turns getting feedback on their plans. This resulted in 10 hours of observation and 90 pages of written field notes that we inductively coded for themes around novice challenges and coaching strategies, which we synthesize below.

3.3.1 Students Lack Focus on Design Argumentation

We found that students often raised surface-level issues in planning meetings and struggled to identify critical issues in their design argumentation. Students often raised surface-level challenges, such as "tech is really hard to test" or "building tech took longer than I expected," rather than deeper structural issues in their argumentation. Even in instances where students did relate issues to their design arguments (e.g. "I struggle with chaining my design arguments"), issues were not specific enough to support a useful coaching discussion on how to address them. We also observed that mentors frequently asked students diagnosis-type questions to understand how the surface-level issues might relate to more critical issues in the student's design arguments, and redirected their attention to focus on argumentation (e.g. "I see you're excited about this prototype, but how is it addressing the user's obstacles?"). From these observations, we learned that tools would need to scaffold students to focus on evaluating their design argumentation, and to support them diagnosing critical structural issues in their arguments as their mentors did.

3.3.2 Students Fail to Recognize Issues in Design Argument Structure

From our observations, we found that mentors spent most of their coaching efforts helping students identify critical structural issues in design argumentation before they could coach students through plans to address them. We highlight and focus on three types of structural issues that mentors often coached students to recognize during these meetings: gaps, a lack of depth, and misalignment. Mentors would often point out specific *gaps* in a student's basic design argument structure (e.g. a student not articulating the desired outcome of their design intervention). Mentors would also use probing questions to help students articulate details and *depth* they felt were lacking in their design rationale (e.g. a student describing outcomes that were not concrete enough to be measurable). Mentors also noted when they saw fundamental *misalignment* between parts of a student's design argument (e.g. noting when a student's proposed solution was misaligned with the user challenge they set out to overcome).

More generally, novices experience challenges in identifying structural issues such as gaps, a lack of depth, or misalignment because they lack the structural knowledge and diagnostic practice that experts have and use, such as the components that make up basic design argument structure, the ways in which components should be detailed, and the causal relationships between them [18, 19, 26, 27, 43, 57]. By embedding these expert models into scaffolds, novices might be able to better articulate arguments for their proposed solutions, that detail the underlying mechanisms of why they expect their designs to work.

3.4. Polaris System

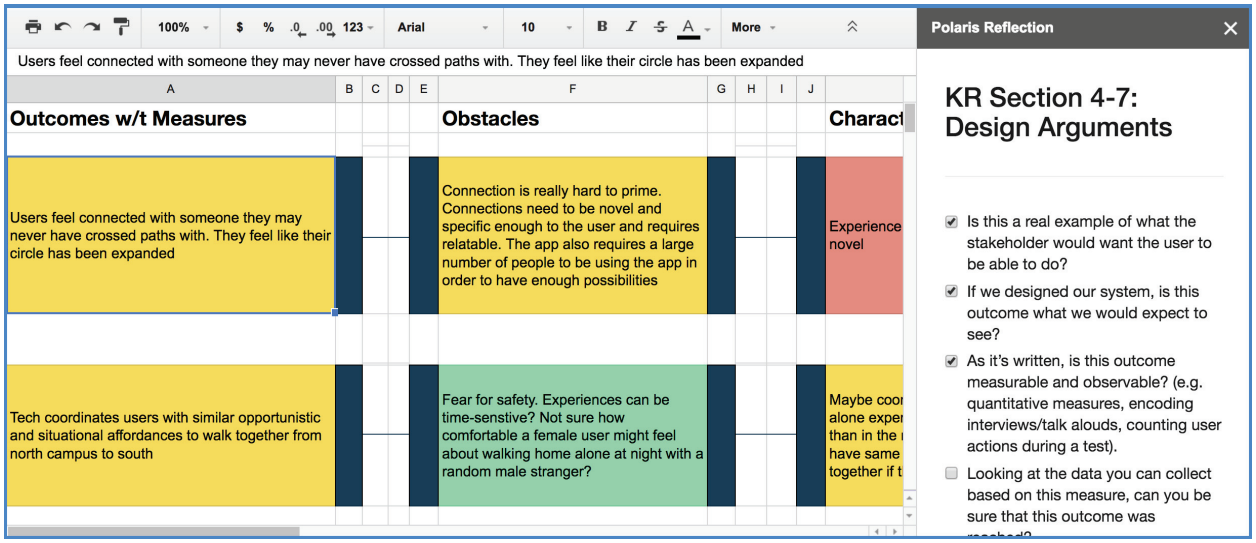


Figure 23. *Polaris* System
We contribute *Polaris*, a learner-centered diagnostic tool that embeds expert knowledge and practice to scaffold novices to construct and evaluate their design arguments. Novices use the *Design Argument Template* (left) and the *Reflection Prompts* (right) to visually and procedurally evaluate their design arguments for structural issues.

To help novice designers construct and evaluate design arguments, we introduce *Polaris*, a learner-centered diagnostic tool that scaffolds novices to visually and procedurally identify key structural issues in their design argumentation (Figure 23). Polaris is composed of (1) a *design argument template* that embeds expert structural knowledge and (2) a series of *reflection prompts* that embed expert diagnostic practice for evaluating their design argument structure for gaps, a lack of depth, and misalignment. By embedding expert knowledge structures and diagnostic practice, we argue that Polaris can scaffold students to evaluate their own design arguments for structural issues they may have previously missed. We detail the features of Polaris below.

3.4.1 Design Argument Template

The *Design Argument Template* is composed of four **components** that represent the structure for the design argument -- Outcome with measures, Obstacle, Characteristic, and Argument (see Figure 24). To compose a design argument, a student first details the parts of their design argument through each component of the template, which correspond to components of an expert design argument structure (Figure 23, left). The design argument template allows students to represent multiple design arguments in different rows, which enables them to represent and distinguish between the different design arguments that together compose the full scope of their interventions. We argue that visually representing the components of an expert's knowledge structure will help novices recognize specific gaps in their own argument structure. For example, a student using the template may articulate their desired user outcome and a proposed solution. When adhering to the template, the student would recognize that their argument is missing the obstacle that is preventing an outcome from being reached.

Using the template, students can explicitly create **conceptual links** between components (see navy links in Figure 23, left) as a way to visually emphasize the necessary causal relationships between parts of the design argument. Overlooking these relationships can lead to arguments that have the basic structural components, but lack causal reasoning. For instance, a student may compose a design argument that includes all individual parts, but their proposed characteristic is not related to the specific obstacle they have written. When a student creates a conceptual link between a Characteristic component and an Obstacle component in the template, it reinforces the

idea that a characteristic of the design is meant to overcome that specific obstacle that an end-user may face. Unlike previous scaffolding tools that solely focus on the components of a design problem, this conceptual linking mirrors an expert practice -- using causal reasoning to connect their argument and explain the mechanisms behind their design.

components of design argument template	**outcome w/t measures:** the specific, measurable, and attainable goal for a design if it is successful. The outcome is a practical design goal rather than theoretical research goal.	**obstacle:** the challenge your design is seeking to overcome in order to reach the outcome.	**characteristic:** the specific set of attributes that define your design and differentiate it from other designs.	**argument:** the reasons for why the characteristics will help overcome the obstacles to achieve the desired outcome.
example of student design argument	*"Tech coordinates users with similar opportunistic and situational affordances to walk together from north campus to south"*	*"Fear for safety. Experiences can be time-senstive? Not sure how comfortable a female user might feel about walking home alone at night with a random male stranger?"*	*"Maybe coordinate the walking home alone experience in the day? Rather than in the middle of the night. Or have same gender users walk together if the event stays at night"*	*"By having tech coordinate easy and opportunistic face to face meeting, a stronger possible emotional connection can be formed. And it's not a burden of the user's time"*
example reflection prompts	Is this a real example of what the stakeholder would want the user to be able to do?	Does this obstacle encompass a core user or stakeholder struggle?	Does this characteristic uniquely describe your approach to solving the problem? Does the characteristic clearly distinguish your design from other possible designs (especially existing designs that do not solve the problem)?	Is there existing evidence from knowledge of problem and users that supports this argument?

Figure 24. Structure of a Design Argument
Here, we explain the structural components of a design argument represented in the *design argument template*, an example design argument that a student constructed using the template, and example *reflection prompts* that a student would use when evaluating their argument.

3.4.2 Reflection Prompts

The *Reflection Prompts* are a set of questions attached to each component of the design argument that use expert heuristics to evaluate the structure of the argument (see Figure 23, right). For example, when a student selects an Argument component in the template, a series of associated reflection prompts will appear, such as: “Is there existing evidence from literature that supports this argument?” As a student evaluates each component of a design argument with the reflection prompts, they can check off each prompt that their current design argument satisfies. As more prompts are checked off for a component, the color of that design argument component changes from red to yellow to green. These visual cues guide students and mentors to quickly spot stronger and weaker areas in their overall argumentation.

template component	reflection prompts
outcome w/t measures	- Is this a real example of what the stakeholder would want the user to be able to do? - If we designed our system, is this outcome what we would expect to see? - As it's written, is this outcome measurable and observable? (e.g. quantitative measures, encoding interviews/talk alouds, counting user actions during a test). - Looking at the data you can collect based on this measure, can you be sure that this outcome was reached? - Given what we can observe of this outcome, would we know that we have addressed a core tension?
obstacle	- Does this obstacle encompass a core user or stakeholder struggle? - Does the obstacle clearly describe how it prevents the linked outcome from being reached? - Does the obstacle present an argument (reason) behind why existing designs are not reaching the outcome (as opposed to just rephrasing that the outcome is not being reached)? - Is the statement of the obstacle something that is concrete enough to be measurable or observable?
characteristic	- Are these characteristics detailed? Could someone read this characteristic and clearly understand what is being communicated? - Does this characteristic uniquely describe your approach to solving the problem? Does the characteristic clearly distinguish your design from other possible designs (especially existing designs that do not solve the problem)? - Is the characteristic stated in a specific enough way that someone can read it and create a set of interface affordances that embody the characteristic? - Have you had a mentor/peer read your characteristic, and correctly describe to you how the characteristic would work when implemented in a system?
argument	- Is this argument logically sound? Is it likely to work? - Does your argument align well the outcomes, obstacles, and characteristics listed for this design argument? - Is there existing evidence from literature that supports this argument? - Is there existing evidence from user testing on past designs that supports this argument? - Is there existing evidence from knowledge of problem and users that supports this argument? - Does this argument explain how the characteristic overcomes the obstacle to reach the desired outcome? - Does this argument explain why the characteristic overcomes the obstacle to reach the desired outcome? - Does this argument explain why your corresponding characteristics address your obstacle? - Is this argument measurable? Can you feasibly gather evidence to support this argument?

Figure 25. Reflection Prompts
A complete list of the reflection prompts used for each component in the design argument template.

To build a set of reflection prompts that represent expert diagnostic practice, we observed 10 planning meetings and coded the common probing questions that mentors asked. We focused on questions related to structural issues like lack of depth and misalignment. See Figure 24 for some

examples of reflection prompts, or Figure 25 for an exhaustive list. Of the reflection prompts, some were *depth-focused prompts,* designed to help students procedurally identify where their argument may be lacking the necessary details for a well-argued claim. For example, "As it is written, is this outcome measurable and observable?" Similarly, some of the reflection prompts were *alignment-focused prompts*, designed to help students procedurally evaluate the relationships between components that must be established for a sound argument, and thus recognize where misalignment issues may occur. For example, "Does the obstacle clearly describe how it prevents the linked outcome from being reached?" For components that had insufficient reflection prompts for assessing lack of depth and misalignment, we further interviewed mentors to capture additional questions that students can use to identify critical issues. We then piloted the prompts with 2 research teams who helped revise wording for clarity. Unlike existing scaffolds that provide evaluation heuristics that are highly specific to the design (e.g. "does this parachute contain a vent?"), these reflection prompts generalize to the structural issues that experts can recognize across designs in different domains.

By focusing student attention on generalizable issues in design argument structure, Polaris trains students in an overall approach for evaluating and improving the arguments for anything they design.

3.5 Polaris Pilot Study

We tested Polaris in a pilot study with 13 undergraduate design-researchers who led ten HCI research projects, and three mentors. Our primary goal in this pilot was to understand how Polaris might scaffold students to identify critical issues in their design argument structures, and how it might help mentors coach students on plans to address those issues. In Part 1 of the study, undergraduates used Polaris to construct design arguments, and listed any issues they identified. In Part 2 of the study, we asked mentors to review students' arguments, list their own issues, review the issues students identified, and generate coaching feedback per student issue. Our core hypotheses were as follows:

1. Students will identify issues focused on design argumentation.
2. Students will find structural issues within their design arguments such as gaps, lack of depth, and misalignment.
3. Students will find issues that (a) mentors consider to be critical and (b) are similar to issues identified by mentors.
4. Mentors will find student issues and design arguments useful for directing how they coach planning.

We detail our pilot study methodology below.

3.5.1 Pilot Study Setting

We sought to test Polaris in a setting where students were both novices in constructing and evaluating design arguments, and had limited access to mentoring resources who could coach them in evaluating their arguments for structural issues. We tested Polaris in an Agile Research Studio [90] (same setting as our needfinding study, described above), in which students learn to lead systems HCI research with the support of faculty and graduate student mentors. Student participants were undergraduate Computer Science majors, and had been enrolled in the class and working on their projects between 1 and 5 quarters (3 month to 15 months). Students are introduced to design arguments as part of the studio course. The three research mentors had between 3-7 years of experience mentoring HCI design research.

3.5.2 Part 1: Student Issue Identification

The first part of the study focused on understanding how Polaris can help novices evaluate and identify structural issues in their design argumentation. Students used Polaris to evaluate their current design arguments for their projects and documented any issues they found. To differentiate between the mechanisms by which the design argument template and the reflection prompts in Polaris helped students identify structural issues, we had students identify issues with just the design argument template, and then again with the reflection prompts alongside the template. The procedure was as follows:

1. **Students construct Design Arguments (15 mins):** Undergraduates were asked to construct their design arguments using the design argument template in Polaris, and create links between the sections.

2. **Students identify issues with template only (10 mins):** Undergraduates were asked to assess their design arguments in the Polaris tool and write down any issues they found into a separate document. They were not allowed to use the reflection prompts during this part of the test.

3. **Students identify issues after using prompts (10 mins):** Undergraduates were asked to open the reflection prompts feature in Polaris and evaluate their arguments using the provided prompts, and again write down any issues they found in a separate document.

To evaluate how Polaris supported students in identifying structural issues within their design arguments, the primary author coded these issues per the three types of critical structural issues identified in needfinding: gaps, depth, and misalignment. To evaluate how different features in Polaris helped students identify issues across components of their design argument, we coded all student issues based on which components of the design argument the students referenced. We used 5 codes in this analysis, one code for each of the four sections of the design argument, and a final “other” code for issues which did not reference the design argument. Issues that referenced multiple components were given multiple codes. To understand the mechanisms behind how the design argument template and the reflection prompts supported students in identifying these

structural issues, all coding was conducted blind to whether students identified these issues using the design argument template, or after using the reflection prompts alongside the template.

3.5.3 Part 2: Mentor Evaluation and Use of Student Issues

The second part of our study focused on understanding the quality of the issues students identified according to their mentors, and how mentors might use those issues to coach students through misunderstanding or through plans to address critical issues. Mentors evaluated the design arguments of a subset of students (6 out of the 10 teams) and documented any issues they found. We then asked mentors to rate the issues students identified, and generate coaching feedback per issue. The procedure was as follows:

1. **Mentors identified issues (10 min per team):** Each mentor reviewed the design arguments of two of their student teams and generated a list of issues they found. Mentors then rated how critical their issues were on a 1-5 Likert scale (1=not critical; 5=very critical). Mentors did not see student generated issues before generating their own.
2. **Mentors assessed student issues (10-15 mins):** We asked mentors to evaluate the issues that two undergraduate mentee teams identified, and rate them 1-5 on a Likert scale on four aspects: accuracy (i.e. whether mentors agreed on the issue), severity (i.e. criticality of issue to the design success), articulation (i.e. the expressiveness of the issue), and actionability (i.e. usefulness for coaching). Mentors could see both the issue list, and students design arguments.

3. **Mentor wrote feedback per issue (20-25 mins):** We asked mentors to write their diagnosis and feedback on each issue their mentees wrote. Mentors could see both the issue list, and students design arguments.
4. **Mentors wrote reflections on their use of Polaris (10-15 minutes):** We asked mentors to reflect upon their use of Polaris in comparison to their normal mentoring in planning meetings.

This process generated the following data: (a) mentor ratings of student issues, (b) mentor diagnosis and feedback from issues, and (c) mentors written reflections on their experiences using Polaris in comparison to their planning meetings with students.

To evaluate the criticality of the issues that students found, we used mentor ratings of the issues students raised on the dimensions of accuracy and severity. To evaluate the degree to which students using Polaris were able to identify issues similar to their mentors, we compared mentor and student issues for the same design arguments that mentors had rated 3 or higher in severity. We then matched the student issues for each project to related mentor issues. Because student issues may vary in their degrees of similarity to a mentor issue, we defined the degrees to which a student issue was similar to their mentor's issue based on the structural similarity (0 = not related; 1 = referring to same component; 2 = same structural issue (e.g. gap, depth, alignment) but lacking specificity; 3 = same structural issue with some specificity; 4 = same structural issue with more specificity, including references to relevant project detail; 5 = near identical issue). The following example illustrates a match between a mentor and student issue with a similarity

degree of 3. The mentor noted that a particular outcome for a design argument lacked depth and were not specific enough to be measurable: *"The outcomes [in outcome 2] identified in the design argument are too general, and therefore are not measurable."* Similarly, the student noted that this outcome lacked measures "*For Outcome 2, difficult/ambiguous to measure because we don't have observable actions we'd be able to see if our design was successful"*.

To evaluate whether mentors could make sense of student understanding from the issues they raised, and whether they found issues useful for their own coaching, we had mentors also rate student issues from 6 of the 10 teams (71 issues) on articulation and actionability for their own coaching. We then asked mentors to record coaching feedback per issue, and note patterns in student misunderstanding. Finally, we qualitatively analyzed the specific feedback that mentors gave students per issue, as well as the broader patterns of misunderstanding that mentors could identify. The primary author conducted multiple rounds of deductive coding on 71 instances of mentor feedback. We use two primary codes in our analysis: feedback focused on (a) planning next steps to address issues or (b) project or conceptual misunderstandings.

3.6 Polaris Findings

3.6.1 Students using Polaris identified issues focused on design argumentation

Students using Polaris diagnosed 91 issues, 95.6% (87/91) of which were related to one or more components of their design argument, rather than surface level issues. 47 of the 91 issues were

identified after using the template, and an additional 44 issues after using the reflection prompts. Student issues spanned all four components of the design argument (see Figure 26 for a component-by-component breakdown with examples of issues students raised). Students identified a wide range of issues, such as recognizing mismatches between a stated outcome and what stakeholders would actually want; lacking reasoning behind why an obstacle exists; lacking details and clarity in characteristics; and missing explanation for how a characteristic overcomes an obstacle to reach a particular outcome in argument.

component	# of issues	examples
outcome	22	*"The Outcome for DA 2 is not a real example of what a stakeholder would want"*
obstacle	17	*"My obstacles should have reasons. As they're specified they really only state a vague problem, not reasons for that problem that I can actually address"*
characteristic	13	*"Our characteristic wasn't detailed enough for someone to be able to replicate our interface because we didn't explain what "points" or "pop-ups" were"*
argument	19	*"Need to explain better why the characteristic overcomes the obstacle to reach the desired outcome"*
other	4	*"Everything is too short"*

Figure 26. Distribution of Issues across Design Argument Structure
A distribution of issues students found across components of the design argument

3.6.2 Students identified structural issues in design arguments

structural issue	# of issues	examples
gaps	13	*"We couldn't think of any characters to overcome or address our original obstacle (being able to detect the growth mindset in coding, both generally and automatically/passively)"*
misalignment	10	*"Inconsistency between parts of chain -> Parts of obstacles not relevant to ideal outcome, characteristics not addressing the obstacles stated"*
lack of depth	70	*"Need to specify what observable actions I'd be able to see if the design is successful"*
not structural issues	3	*"For the Characteristics section for DA 1, Other peers/mentors have not read my arguments"*

Figure 27. Types of Structural Issues in Design Argument
The types of structural issues students found when evaluating their design argumentation

Students using Polaris identified three types of structural issues in their design argumentation: gaps in their argument, a lack of depth within components of their argument, and misalignment between components of their argument. Of the 91 issues students found, 96.7% (88/91) of the issues were related to at least one of these three structural issues (see Figure 27 for a breakdown across issue types with examples of issues students identified). Notably, students found many more depth issues than other types of structural issues. However, this is expected as there are many more possible issues related to depth in design argument structure, relative to gaps and misalignment (e.g. there are at most 4 possible gaps in a design argument, but each of the 4

components could lack depth in many ways). We analyze the issues students identified for each type of structural issue below.

3.6.2.1 Gaps in design argument structure

gap type	# of issues	definition	examples
missing components	6	a component of the design argument is not represented	"or Outcome 1, we don't know what our core obstacle is" "We couldn't think of any characteristics to overcome or address our original obstacle"
missing design arguments	3	one of the design arguments for the intervention is not represented	"Missing arguments for situationally aware" "Incomplete set of design arguments"
mismatched component type	4	two different components are not distinguished from one another	"...what actually is the outcome? The way I've written it, I'm talking about the characteristics of the spaces, not the actual outcome." "Design Argument seems circular. The outcome seems to be the same as the task item"

Figure 28. Types of Structural Gaps.
The types of gaps students found in design argument structure with illustrative examples

9 of 10 student teams identified 13 instances of gaps in their design arguments after using Polaris. In analyzing these instances, we found that students identified three kinds of gaps:

missing components, missing design argument, and mismatched components; see Figure 28. The most common type of gap that students identified using Polaris was when one of their design arguments was lacking one of the four fundamental components (outcome, obstacle, characteristic, argument). For example, one student issue raised was *"We couldn't think of any characteristics to overcome or address our original obstacle."* We also saw cases where students missed a design argument for a core feature of their system (*"Missing arguments for situationally aware [characteristic].*"), and where students wrote the same thing for two different components, suggesting a gap due to mismatched components (*"...what actually is the outcome? The way I've written it, I'm talking about the characteristics of the spaces, not the actual outcome."*) These findings suggest that having students detail each component of each design argument in a template may also help them recognize when they are not distinguishing between parts of an argument, or are missing a design argument all together.

3.6.2.2 Lack of depth in design argument structure

9 of 10 student teams identified 70 depth issues in different parts of their design arguments after using Polaris. We found that students identified 22 depth issues after using the template, and 27 additional depth issues after using the prompts alongside the template.[1] It is possible that students were able to identify many depth issues with templates alone because more experienced students

[1] When comparing depth issues found with templates and after prompts, we removed 21 issues about measurable outcomes when comparing template and prompt issues, because this depth heuristic for outcomes had been encoded in both the template header (Outcome w/t measures), and as reflection prompts.

may already be familiar with some depth issues that have come up in prior coaching meetings. Still, after using the prompts, students found many more depth issues, which suggests that the prompts helped students focus on specific heuristics that they had not considered when they initially evaluated their design arguments using just the template.

To evaluate whether depth-focused prompts helped students identify more detailed issues than with the template alone, we also looked for paired instances of issues where a student found a depth issue with just the template, and then used the prompts to articulate a more specific version of the issue. We found 4 instances from 4 of the 10 student teams that went from a less specific issue with the templates to a more specific issue after having used the prompts. Below is an illustrative example: after using the design argument template, a student wrote the issue *"I feel that my obstacles for DA 1 are not as specific as I'd like them to be, but I'm also unsure how to write them succinctly"*. The same student wrote this more specific version of the issue after using the prompts: *"In the Obstacle section for DA 1, I don't think I've captured both stakeholders"* Given that one of the Obstacle prompts maps directly to the detail in the issue ("Does this obstacle encompass a core user or stakeholder struggle?"), having the prompt likely helped the student more specifically articulate this issue. While there were not many paired instances, we see some evidence that the prompts may have helped students articulate more specifically what the depth issues were in their design arguments.

3.6.2.3 Misalignment in design argument structure

misalignment type	# of issues	examples
between components	9	*"Need a stronger link of argument to characteristics. The more specific the better"* *"Inconsistency between parts of chain -> Parts of obstacles not relevant to ideal outcome, characteristics not addressing the obstacles stated"*
across multiple components	1	*"Argument needs a lot more specific language, and needs to map characteristics to overcoming obstacles to achieve outcomes, and just doesn't"*
general nonspecific	2	*"Mappings aren't clear between sections of the DA (i.e. Not explicit, not clearly implicit)"*

Figure 29. Types of Structural Misalignment
The types of misalignment students found in design argument structure with illustrative examples

6 of the 10 student teams identified 10 issues related to misalignment between components of their design argument. Within these issues, students noted 12 specific relationship links where they found misalignment, which we coded into three types of misalignment: (1) between two components, (2) across multiple components, and (3) general misalignment where no particular relationship was specified. 75.0% (9/12) of misalignments that students identified were between two components (see Figure 29). Of the 9 misalignment issues between components, 5 were about the relationship between outcomes and obstacles. A possible explanation for students

catching so many misalignment issues between components, especially between outcome and obstacle, might be that students have internalized the relationship between outcome and obstacle more than others.

This might be because they practice aligning outcomes and obstacles most often in the early stages of their design-research projects as they iteratively conduct needfinding and develop user stories, a precursor to any solution development like proposing or prototyping characteristics.

Overall, our findings demonstrate that Polaris supported students in identifying three core types of structural issues in their design arguments: gaps, lack of depth, and misalignment. By providing novices with scaffolds that embed expert structures and heuristics in visual templates and procedural prompts, novice design researchers were able to better recognize when the structural state of their own arguments are lacking the rigor that one might see in an expert's model. This suggests that students may be able to effectively identify core structural issues across their design arguments independently, rather than relying on a mentor coaching session to surface the structural issues they may have missed or not focused on in their own evaluation.

3.6.3 Mentors rated student issues as accurate and severe

Mentors generally considered student issues to be accurate and severe. When asked on a 5pt Likert scale the degree to which mentors agreed that a student issue was accurate, mentors on average rated student issues as 4.06 on accuracy (standard deviation = 0.97; between agree and strongly agree). Similarly, mentors on average rated student issues as 4.11 on severity (standard deviation = 1.02; between agree and strongly agree). We also found that mentors considered

71.83% (51/71) of student issues to be accurate (ratings of 4 or 5) and 73.24% (52/71) of student issues to be severe (ratings of 4 or 5).

When comparing the general severity of issues that mentors identified to the issues students identified while using Polaris, we found that student and mentor issues were similarly severe. We first asked mentors to rate the severity of their own issues and then the severity of student issues. We found that the average mentor rating of severity for student issues (4.11 , n = 71) was similar to the average mentor rating of severity for their own issues (4.34 , n = 32). These findings demonstrate that not only did mentors consider student issues as generally severe, but that the average severity of student and mentor issues were relatively similar.

After mapping student issues back to mentor issues, and then coding student issues with the degrees to which they were similar to a corresponding mentor issue (see Polaris Evaluation section), we also found that the degrees of similarity between student and mentor issues ranged from 2 to 5, and that the average degree of similarity was 3.57, between "same structural issue with some specificity" and "same structural issue with more specificity." The following example illustrates a match between a mentor and student issue with a similarity degree of 3. The mentor noted that a particular outcome for a design argument lacked depth and were not specific enough to be measurable: *"The outcomes [in outcome 2] identified in the design argument are too general, and therefore are not measurable."* Similarly, the student also noted that the same way in which this outcome lacked depth: *"For Outcome 2, difficult/ambiguous to measure because*

we don't have observable actions we'd be able to see if our design was successful". Overall, these findings suggest that with the support of Polaris, students were able to independently find accurate and severe issues in their design rationale that were generally similar to the issues found by their mentors.

3.6.3.1 Students identified the majority of severe mentor issues

comparison	% coverage
Severe mentor issues that students also found	55.17% (16/29) of mentor issues
Severe mentor issues that students did not find	44.83% (13/29) of mentor issues

Figure 30. Student Coverage of Issues Identified by Mentors
Student coverage of critical structural issues that mentors identified

We found that 55.17% (16/29) of severe mentors issues were topically similar to issues that their students independently identified with Polaris. This finding supports our original hypothesis by demonstrating that students were able to identify issues similar to the majority of the severe issues their mentors found. However, 44.83% (13/29) of those mentor issues were issues that students did not find with Polaris (see Figure 30). Through further analysis, we found that 8 out the 13 severe mentor issues that students did not find were outside the scope of the Polaris design. These issues tended to be (a) project-specific details or (b) related to parts of the research argument beyond design arguments. For example, for (a), a mentor noted the following

project-specific issue that would not generalize into a tool for any student: *"In first design argument on learning, the design argument does not mention learning at all (i.e. discuss how changing locus of control will produce greater learning)."* For (b), a mentor noted the following issue in another part of the research argument beyond the fundamental design argument: *"Mismatch in tech-specific outcome and design-based obstacle. Obstacle is about whether people want to do it, but outcome is written about what the tech can do?"* Here, the mentor is distinguishing between an obstacle in a general design argument, and a technical system obstacle in a technical argument for things like system framework or architecture designs, the latter of which was out of scope for our initial design and evaluation of Polaris. Because Polaris is designed to support students in identifying common structural issues in their design arguments for their projects, we would not expect students to find either of these issues that were project-specific or beyond the design argument.

Factoring out the 8 mentor issues that were not issues Polaris was designed to support students in catching, we found that 23.81% (5/21) of severe mentor issues were not found by students. The issues that students did not catch were types of structural issues about depth and misalignment that we would expect students using Polaris to identify. However, we would not necessarily expect Polaris to help students catch every structural issue that mentors could catch, and consider that it helped students catch 76.19% (16/21) of severe mentor issues to be a significant indicator of independent student performance.

3.6.3.2 Students identified severe issues mentors had not caught

comparison	% coverage
Severe student issues that mentors also found	40.38% (21/52) of student issues
Severe student issues that mentors did not find	59.62% (31/52) of student issues

Figure 31. Mentor Coverage of Issues Students Identified
Mentor coverage of critical structural issues that students identified

We also found that 59.62% (31/52) of the student issues that mentors considered severe (rating of 4 or 5) were issues that mentors did not find on their own (see Figure 31). A possible explanation for this finding might be that mentors themselves struggle to recall all the possible structural issues and heuristics for evaluating design arguments, which could be due to a multitude of factors. For example, mentors in the experimental setup and in practice face real-world time constraints when evaluating the arguments that students articulate and issues they raise, and are likely unable to exhaustively review a student's design arguments for possible issues in the context of a planning meeting. Similarly, mentors may choose to omit less critical structural issues from their discussions and focus on more critical ones given limited mentoring resources. Alternatively, it's possible that mentors may struggle to recall specific project details that students are more regularly considering when actively working on their own project. These findings suggest that Polaris can not only support students in catching issues that mentors

consider severe, but that it may also help students catch critical issues on their own that mentors might overlook or skip over in time-constrained feedback sessions.

3.6.4 Mentors found issues articulate and actionable for coaching students

Beyond accuracy and severity, mentors rated student issues as generally articulate and actionable for their own coaching. When asked on a 5pt Likert scale, mentors on average rated student issues as 4.14 on articulation (standard deviation = 1.25), falling between agree and strongly agree, and 3.72 on actionability, (standard deviation = 1.06) falling between neutral and easily actionable for coaching. We also found that mentors considered 73.24% (52/71) of student issues to be articulate (ratings of 4 or 5) and 73.24% (52/71) of student issues to be actionable (ratings of 4 or 5). Further, mentors generally found more articulate issues to also be more actionable for coaching students. For the issues that mentors considered articulate (rating of 4 or 5), the average actionability of those issues was 4.15 (n = 52). In contrast, for the issues mentors considered to be less articulate (ratings of 3 and below), the average actionability was 2.53 (n = 19). A possible explanation for these findings is that a more articulate issue means that mentors have more detail that illustrates the student's current understanding with greater clarity, which informs how the mentor would take action to coach the student.

3.6.4.1 Mentors focused coaching on planning next steps when they agreed with student issues

issue type	% of accurate issues (n = 34)	% issues with feedback on plans to address issue	% issues with feedback to correct project or argument inaccuracies
articulate	73.53% (25/34)	72.0% (18/25)	12.0% (3/25) 4% (1/25) - project inaccuracies 8% (2/25) - argument inaccuracies
inarticulate	26.47% (9/34)	22.22% (2/9)	66.67% (6/9) 44.44% (4/9) - project inaccuracies 22.22% (2/9) - argument inaccuracies

Figure 32. Types of Mentor Feedback for Accurate Issues

When mentors considered student issues to be both accurate and articulate (ratings of 4 or 5), mentors were able to offer suggestions for planning next steps to address those issues (see Figure 32). Of the 25 issues that mentors provided feedback for and considered accurate and articulate, 72.0% (18/25) of mentor feedback suggested plans for next steps to address the issue. This is in contrast to issues that mentors considered accurate but *inarticulate*, for which only 22.22% (2/9) of mentor feedback was about planning (see Figure 32). Below is an illustrative example of a mentor giving planning feedback for a student issue that the mentor rated as articulation = 5 and accuracy = 5:

> **Student Issue:** *"Design arguments is bias towards my own expectation/guesswork as opposed to addressing the needs stakeholders"*

> **Mentor Feedback:** *"This is awesomely clear. I would ask the student to structure a test to see what learners can or cannot do using basic/existing tools and designs. I would though first discuss if this is the outcome we actually care about -- and see if we can structure our initial test to get a list of goals/tasks learners struggle with, before focusing on arguments for reaching this particular outcome."*

These findings are consistent with our hypothesis that for student issues that mentors agreed with and considered articulate, mentors would focus their feedback for students on how to address those issues. These findings suggest that by seeing issues that mentors consider accurate and articulate, mentors are able to offer planning feedback to students that is oriented around next steps to address core issues.

3.6.4.2 Mentors focused on student misunderstanding when issues were inarticulate or inaccurate

issue type	% of inaccurate issues (n = 14)	% issues with feedback on plans to address issue	% issues with feedback to correct project or argument inaccuracies
articulate	71.43% (10/14)	10.0% (1/10)	80.0% (8/10) 50% (5/10) - project inaccuracies 30% (3/10) - argument inaccuracies
inarticulate	28.57% (4/14)	25.0% (1/4)	50.0% (2/4) 50% (2/4) - project inaccuracies 0% (0/4) - argument inaccuracies

Figure 33. Types of Mentor Feedback for Inaccurate Issues

We also found that for the issues that mentors considered either inarticulate or inaccurate, mentors focused their feedback on correcting or clarifying the student's understanding of the issue. For issues that were accurate but inarticulate, 66.67% (6/9) of mentor feedback was focused on correcting project or argument misunderstandings that students may have exhibited in their issues. For inaccurate but articulate issues, 80.0% (8/10) of mentor feedback was about correcting student's understanding of the project or general design argument structure (see Figure 33). Below is an illustrative example of a mentor identifying a specific misunderstanding and offering feedback for inaccurate but articulate student issues. Here, the mentor offered feedback to address the student's project-specific misunderstanding:

> **Student Issue:** *"Outcome 1 needs to be more concrete -- actions that promote learning vs. just promoting learning"*
>
> **Mentor Feedback:** *"If outcome 1 was more concrete, it would just become outcome 3. This outcome is intentionally measuring something less direct than outcomes 2 and 3. The ultimate goal of this design is to promote learning, and we hypothesize that creating better collaboration and engagement will increase learning. We want to measure both."*

These findings suggest that even when students express issues poorly or inaccurately, Polaris can still help mentors catch problems with a student's conceptual understanding of the project itself---a precursor to forming better arguments and understanding critical issues. Still, well-articulated issues are helpful (even when inaccurate), as they provide mentors with more

information that illustrates a student's understanding in their project or in the design argument structure.

Below is an illustrative example of project feedback that a mentor gave for a student issue that the mentor rated as articulation = 3 and accuracy = 4.

> **Student Issue:** *"Our obstacle sounds like it is addressing an interface problem but if we could reword it it would go deeper than the interface -- our real problem is detecting the growth mindset (we don't have that many obstacles in terms of the interface that we are aware of yet)"*
>
> **Mentor Feedback:** *"I'm not totally clear on the difference you're identifying between interface obstacles and non-interface obstacles. The point about the real challenge being around detecting growth mindset is good. Our approach is to identify good coding practices (process) and reward those, and then work to understand the relationship between those and the growth mindset."*

While the mentors may have rated the student's issue as accurate, it's possible that mentors focused on giving feedback to correct project and argument inaccuracies because the issue may not have captured enough detail in the issue for mentors to accurately assess whether or not the student was naming the issue that the mentor saw. This uncertainty of the student's understanding of the issue could lower a mentor's confidence that they should coach the students on next steps, and perhaps lead mentors to try to correct perceived misconceptions with whatever

information they have. These findings suggest that while mentors may consider a student issue accurate, if the issue is not articulate, that issue may still be missing detail necessary for a mentor to really assess a student's understanding of an issue. While limiting for coaching students on next steps, mentors find that this is still enough information to focus their feedback on clarifying the root cause of a specific critical issue with a student, which not only helps ensure the mentor and student can agree on the fundamental issue, but gives mentors an opportunity to understand the gaps in a student's project or general design argument understanding.

3.6.4.3 Mentors diagnosed weak areas of understanding based on patterns of student issues

After seeing the series of issues students raised in their design rationale, mentors were able to detect patterns and diagnose generally weak areas of student understanding. Mentors found 14 patterns of misunderstanding for 5 of 6 student teams whose 71 issues they reviewed. We found that 92.86% (13/14) of the patterns that mentors found were patterns of misunderstanding in the fundamental argument structure (e.g. gaps, level of depth, and misalignment) that mentors could now see after reviewing many examples of the same misunderstanding. Mentors noted patterns of students generally struggling with particular gaps in argument structure. For example, one mentor noted *"I think he didn't understand the outcome as it should be about ideal outcomes for our system or research. as it is stated, I saw them as an intermediate/necessary step, not our system/research outcome."* Mentors also noted patterns of students generally struggling with alignment between components of the design argument. For example, a mentor wrote *"The*

students are not doing a good job of stringing together their work to create a full design argument." Mentors also noted patterns of students generally struggling with levels of depth in their design arguments. In some cases, mentors noted how students struggled with a particular heuristic, for example one mentor noted *"The students are struggling to use evidence from the literature in support of their design arguments"*. In other cases, mentors noted how students struggled with levels of depth more broadly: *"The student may struggle with the granularity of a design argument. Some things seemed very specific, others way too general. Student did notice some of these issues, but overall I think it's still a challenge"*. This analysis suggests that, in addition to Polaris supporting students in detecting structural issues like gaps, lack of depth, and misalignment, that seeing these student-detected issues gives mentors insight into patterns of structural misunderstanding that students exhibit in their argumentation.

Mentors also noted 4 cases where one pattern of struggle may be interconnected to another pattern for a student. For example, a mentor described the following pattern: *"The students had trouble writing arguments that incorporated their obstacles because they did not understand how to define obstacles, even though they included a lot of evidence from the literature in their arguments."* In this reflection, the mentor is noting how a student's struggle with defining obstacles (a gap pattern) is affecting their ability to incorporate the obstacle within the argument (a misalignment pattern). Generally, these findings suggest that when mentors can see several examples of structural issues that a student team may identify, mentors can actually detect

patterns of misunderstanding, which allow them to diagnose specific weak areas in student's conceptual models of their research or general design argumentation.

When reflecting on how Polaris might support their coaching efforts, mentors also reported how they found Polaris useful for seeing a student's current understanding and being able to diagnose weak areas of their understanding. After mentors noted patterns in student misunderstanding, we asked mentors how Polaris may or may not have influenced their process of identifying issues in student mental models, particularly when compared to their typical coaching experience. One mentor described how Polaris makes apparent the student's understanding, saying *"the polaris [tool] asks questions that I wanted to ask. and it was easy to see which part of design arguments the student struggles the most"* and that the *"polaris tool helps us better understanding student's perceived weakness at a glance."* Another mentor reflected on how this knowledge might influence how they coach students on project direction and planning, saying *"Polaris made it very clear which areas of the design argument are weak, and where we should focus our efforts going forward."* One mentor described in detail how seeing several examples of student arguments and issues from Polaris made the students' understanding much more apparent than their typical approach to coaching students and making sense of the student's understanding without explicit support:

> *"I think it's incredible how simply seeing the students' arguments in Polaris and the issues that are noted on the arguments, really gave me a sense of where the student's understanding is at, both in terms of thinking about research and in terms of their*

> *thinking on their own project. In this example, there is a lot of issues the student is having in coming up with specific, well-grounded designs backed by good arguments. I knew this at an intuitive level in SIG meetings, and knew this was challenging, but there are so many specific issues surfaced here that we would work off, one by one, with nice actionable items next to each one. This is something that I try to do in SIG but diagnosing can sometimes happen at a pretty high level. And especially when there are lots of issues, it's very difficult to get a view of all the issues and quickly be able to prioritize them in my head, which I felt like I could do with Polaris a bit and even more so as I evaluated student's issues and noted my own (so the process of this experiment was itself very helpful).”*

These findings suggest that seeing several structured examples of student issues may even expose the student’s state of understanding to the mentor in ways where the mentor can recognize weak areas for the student in both their project and for their skill development in general research argumentation, which can inform the ways in which mentors coach novice researchers over time.

Overall, these findings demonstrate that when students use Polaris to identify structural issues in their design rationale, mentors can use the combination of the design arguments and corresponding issues that students articulate to triage and focus coaching conversations in three ways: (1) coaching on next steps when students and mentors are in agreement on the critical issues on which to focus their efforts; and (2) project-specific and design argument model

coaching when students and mentors are misaligned on their understanding of an issue; and (3) longer term coaching where mentors can focus coaching efforts with students on working through weak areas of project and research understanding that they observe from repeated patterns of misunderstanding. These findings suggest that Polaris may help mentors better triage the issues that students raise in their projects, which is particularly valuable in settings like Agile Research Studios, where mentoring resources are heavily constrained.

3.7 Discussion

Our findings demonstrate that Polaris supported novice designers in identifying issues across key components of their design rationale - outcomes, obstacles, characteristics, and arguments, rather than surface-level issues that novices often raise. Literature describes how experts focus on evaluating their design argumentation to identify the critical issues which drive the next steps they take to improve their designs [77, 43, 18]. By scaffolding novices to focus their evaluation on the argumentation that justifies their designs, students may be better equipped to plan out impactful iterations that improve the fundamental logic behind their designs.

Polaris has the potential to improve how students identify critical issues and improve their designs because it embeds the structural knowledge and diagnostic practice that experts use to evaluate their design argumentation into scaffolds for novices. Our findings demonstrate that the design argument template and reflection prompts helped students visually and procedurally evaluate their design arguments for key structural issues that mentors often focus on: gaps, lack

of depth, and misalignment. By embedding such knowledge and practice into scaffolds for students, students can recognize and improve the fundamental structural patterns that make or break a design argument, as experts do.

Not only did Polaris support students finding these structural issues, the tool also helped students raise what mentors considered critical issues in their design arguments, the majority of which were topically similar to issues mentors found themselves. These findings suggest that scaffolds like Polaris may help students increasingly self-direct critical issue identification in their design argumentation. However, tools like Polaris are not meant to replace mentors. As evidenced by our findings, domain-specific project issues are critical issues that may not be represented in generalizable argument scaffolds like Polaris. By helping students identify general issues in argument structure, mentors with necessary domain expertise may be able to better focus their efforts on training students with the knowledge needed to identify domain-specific issues and design effective solutions. Further, our findings show that by externalizing a student's current understanding of project specifics and general design argument structure, Polaris directly informs the coaching strategies that a mentor may enact, from planning out next steps when in agreement on a key issue, to coaching a student through a misunderstanding that is now made apparent. By scaffolding students to raise critical issues that mentors generally agree with, limited mentoring resources can be spent supporting novice designers in their many other learning needs, such as learning to plan out iterations that address critical issues.

Future work may consider how to apply the way Polaris focuses on structural issues of gaps, lack of depth, and misalignment to other layers of argumentation that experts often use to justify technical designs and design research interventions that go beyond the design argument itself [35]. Findings show that mentors were able to identify critical issues in areas of design-research argumentation that were beyond the scope of design arguments and Polaris, such as distinguishing between a user obstacle in the design argument, and a technical implementation obstacle in a system argument. Future systems can adopt Polaris features that highlight generalizable structural issues (i.e. gaps, lack of depth, and misalignment) in the technical argumentation (e.g. justifying a particular interface and system architecture design) or the research argumentation (e.g. positioning their conceptual approaches relative to existing research) often used in technical design and design research work.

However, scaffolding a novice to construct and assess structural issues across different forms of design-research argumentation introduces its own technical challenges, and may not be as simple as generating a series of multiple templates and corresponding prompts. As prior literature suggests, complex structural representations and extensive sets of heuristics can hamper the practical usability and usefulness of such tools for improving arguments and designs [77, 73]. Moving forward, future argument scaffolds can explore how to help novices visualize, navigate, and conceptually link the different layers of argument structure that justify their overall designs (e.g. design vs. technical vs. research argumentation). For instance, future diagnostic tools may visualize multiple layers of argumentation at once with one argument layer actively in focus and

related argument layers just out of focus to reinforce the conceptual links across layers of argumentation (e.g. how a general design argument is instantiated into an interface design, and how the interaction requires novel system architectures and frameworks). As another example, future diagnostic tools could implement visual cues that can explicitly guide novices to focus on the most critical areas across their overall intervention. Such affordances may better scaffold novices to construct and evaluate conceptual and narrative links across the layers of their argumentation.

Critically, future work should also consider how to integrate these diagnostic tools with existing pedagogy and coaching techniques in ways that allow for increasingly scalable and effective training environments. As our findings demonstrate, design mentors on their own may not be able to catch and coach students through the many different structural issues that come up in their argumentation, particularly given the real-world time constraints of most coaching interactions. While scaffolds like Polaris can effectively scaffold novices to catch more of these generalizable structural issues within design arguments, they are not designed to surface the domain-specific conceptual issues that domain experts would catch, as seen in our findings. When integrated in practice, diagnostic tools like Polaris can supplement the efforts of coaches for more comprehensive scaffolding for novice designers, transforming how mentors coach novices in design training environments. For example, mentors could deploy suites of generalizable argumentation structures across many students at once, scaling their coaching efforts to focus on conceptual issues that only they have the domain expertise to recognise.

Similarly, using generalizable argumentation structures could streamline coaching discussions by creating a shared language of expertise between students and mentors, enabling mentors to point out structural concerns that students can quickly recognise and work on, even within a feedback session. Importantly, one must consider where additional coaching on argumentation may occur within the ecosystem design. For instance, SIG meetings are already time-constrained feedback venues that focus on the practical iteration planning. A possible solution may be to introduce feedback venues that focus on improving the argumentation behind designs, where mentors can focus on delivering this coaching. Future work should further consider the socio-technical integration of tools, pedagogy, coaching, and self-practice in forming effective learning ecosystems for design.

Chapter 4. Compass

A Framework to Manage Planning Process Across an Ecosystem

Expanding the ARS Planning subsystem with supports like the Polaris system allowed for the ecosystem to better scaffold students in the component skills required for effective planning. Further, it provided valuable information that mentors could use in their coaching interactions, enabling them to offer much more specific feedback on how students can enact better planning strategies. Over time, we further extended the ARS Planning subsystem, introducing tools like linked canvases, which scaffolded new layers of argumentation beyond the design argument. To ensure students were able to get dedicated feedback on their argumentation, we introduced feedback venues like Mysore to coach students on how to identify and improve the structural risks in their design rationale.

However, the introduction of more supports in the planning subsystem also meant the layering of new learning interactions to enable student practice. Just as the original ARS ecosystem guided students through routines of practice (e.g. compose your plan in your sprint log and bring it into your SIG meeting to receive planning feedback), the extended planning subsystem included a series of subroutines that enabled the practice of each aspect of expert planning (e.g. to practice visualizing the design problem and diagnosing risks, one should regularly update their argumentation in your canvases with learnings from the previous iteration, assess arguments for risks prior to your mysore feedback session, go into your mysore feedback session with a risky

slice of the argument they wish to workshop with a mentor, implement that feedback into revised arguments and a revised prototype, conduct user testing that focuses on the biggest risks, etc). As the sequences of learning interactions grew in complexity, we observed breakdowns in how students managed their planning process across available ecosystem supports (see Figure 34).

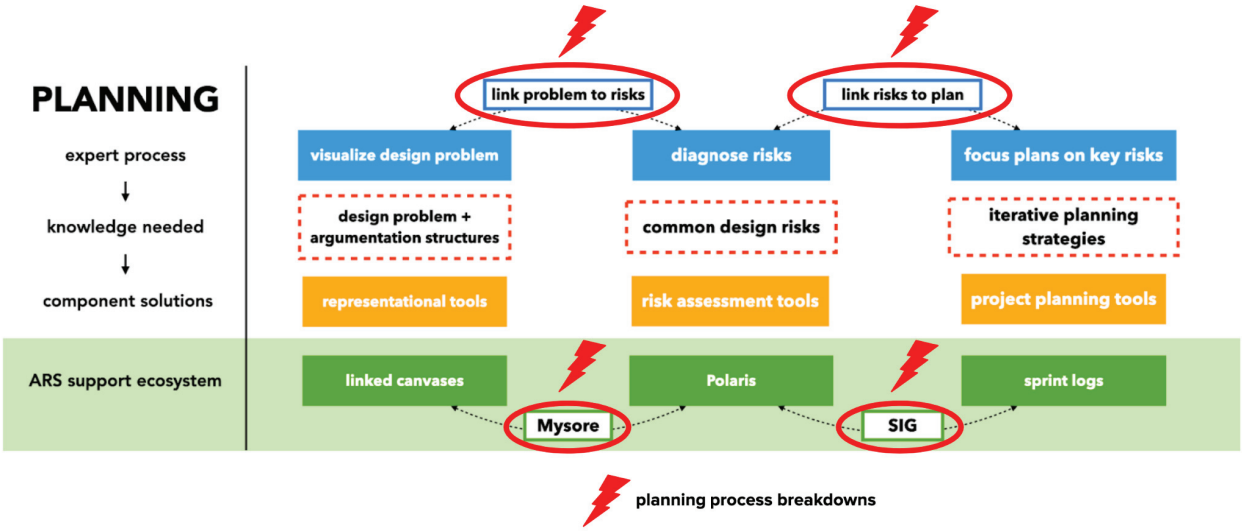


Figure 34. Breakdowns in Managing Planning Process for Students
As we extend existing subsystems with new tools and processes, the overall learning ecosystem grows in complexity. We began to observe process breakdowns as students attempted to leverage the ecosystem towards their learning. Here, we see planning process breakdowns, where students struggle to connect their understanding of the problem to critical risks they have, and then connect those risks to their iteration plan.

For instance, students would receive valuable planning feedback in SIG meetings, and fail to revise their plans in ways that meaningfully and feasibly incorporated that feedback. As a result, we observed students continue to execute their old plans, or simply add on new tasks to their existing plan, not considering how to adapt their former plans to feasibly complete their revised

deliverables. To address these challenges in managing process execution, in Chapter 4 we introduce *Compass*, a process management framework that integrates into the ecosystem, guiding students to capture and feasibly implement planning feedback into revised plans at opportune moments.

4.1 Introduction

Learning to effectively plan out design research is a composition of different skills. Novices must learn to represent the general structure of the problem, identify risks within the problem, and then focus their iteration on a plan that de-risks the most critical aspects of the design [19]. Many design-research learning environments are designed to scaffold students in these different skills. To do this, they typically introduce a variety of tools, processes, and social structures. For example, novices may use design canvases to lay out the rationale of their design, such as user needs, desired outcome, challenge, proposed solution [30, 2, 45, 46, 48]. More recently, researchers have introduced risk-assessment tools to help students diagnose risks in their design argumentation, indicating where they should focus their next iteration [26, 43, 18, 71]. Design sprints have become a popular pedagogical approach for iterative sprint planning [56]. Some design-research studios have even proposed entire ecosystems that weave these scaffolds together to form an environment for training each aspect of design-research planning [90, 73].

However literature suggests that in practice, experts do not follow this planning process procedurally. Rather, they take a flexible approach to managing their planning process -- still

maintaining focus on their goal, but continuously adapting their plans to changing context [51, 82, 55]. Further, in our needfinding, we discovered that even when students practicing design research make and execute plans in ecosystems rich with planning supports, they often fail to capture and implement the planning feedback surfaced by these supports throughout the week.

Being able to flexibly move between the stages of planning to continuously adapt plans, can be challenging to practice while also learning to leverage the corresponding scaffolds available in the community. Even when these scaffolds are designed to work in tandem as subsystems to support effective planning, learning to use them in tandem requires its own training and practice. For example, a student may be skilled at breaking down a large task into actionable steps for the week, or use a sprint tool to scaffold their task breakdown. They may also be skilled at identifying critical risks when assessing their design work, or use a risk assessment scaffold to recognize those risks. However, that student may not recognize how to strategically adapt their planning strategies when they receive planning feedback in the middle of executing their plan. If students are not explicitly trained to bridge these component planning skills across the available scaffolding, they may not leverage the full benefits of an ecosystem meant to train them across the skill of planning.

To address this gap, we introduce Compass. Compass is a process management framework designed to scaffold students in executing their planning process across available supports in the ARS ecosystem. Compass uses a combination of an on-action dashboard and in-action cues to help students continuously adapt their planning strategies throughout the week, in response to

feedback at opportune moments. The on-action dashboard provides a template that helps students align the most critical known risks in their project, the planned deliverables that would address those risks, the status of those deliverables, detailing a series of next steps to take throughout the week. Students present their plan during their SIG meeting, and capture live feedback from their mentors as they review each component of the plan for the week. The in-action cues work in tandem with the dashboard to strategically cue students at opportune moments to prepare for feedback venues and implement planning feedback. For example, students receive a cue to update their plan right after their SIG meeting in order to incorporate valuable feedback, or a reminder to re-plan mid-week, prior to leveraging a Mysore feedback session with their mentor. Together, the dashboard and the cues help students practice managing their own planning process by leveraging the planning supports available to them in the ecosystem.

In an 8-week pilot study comparing student planning process with and without Compass in place, we found that students using Compass revised their plans throughout the week before and after key feedback venues. Further, mentor assessment of the plans demonstrated that student plans were more structurally aligned as students adapted and executed them throughout the week, and that the plans had better implemented the planning feedback students received from their mentor. These findings suggest a need to augment such ecosystems with process management frameworks like Compass, to guide students to strategically adapt and execute effective processes as they engage in sequences of learning interactions designed to help them practice

core regulation skills, such as planning. Through such ecosystems, students can learn to move forward by leveraging the many supports available to them.

4.2 Background

4.2.1 How Experts Adapt their Plans as they Execute

At its core, leading design research is about learning to solve complex problems. To make meaningful progress on such problems, strategic planning is a necessary skill to master [27, 1, 54, 3]. A long history of literature across cognitive and educational psychology defines generalizable models of the planning process when solving problems [63, 14, 47, 69]. For instance, one foundational model breaks down planning into a four step process: (1) understand the problem (i.e. what is being asked for, is there enough information?); (2) make a plan (i.e. look for patterns, organize information); (3) carry out the plan; and (4) evaluate its effectiveness [69].

However, literature across cognitive science, HCI, and the learning sciences suggests that in practice, planning is not quite so procedural or generalizable across contexts [51, 82, 55]. The ways in which experts execute their planning process depends on the specifics of the problem they seek to address. Specifically, experts use mental models of the problem to simulate different scenarios, predict breakdowns, and plan next steps. These mental models vary widely across different problem domains, which suggests practical limitations to generalizable models of expert planning [55]. Further, even when skilled planners have these procedural roadmaps, in

practice, they flexibly adapt their plans as they execute them. Specifically, expert planners orient their plans around primary objectives. Then, as situational context changes or new opportunities arise, they dynamically revise their plans in response to new information to achieve the objective, or adapting the objective when needed [51, 82]. While the planning process has been extensively studied and modeled for domains with simpler, more structured problems like math, less work has been done to understand planning process for some of the most complex problems -- i.e. design problems, where there are no known solutions [55].

4.2.2 How Expert Designers Manage their Planning Process

Within the domain of design-research, experts practice iterative planning. Over the years, researchers have examined different components of iterative planning, such as how expert designers visualize what they know about the problem space [30, 57], diagnose risks in their design rationale [26, 27, 37], and focus their plans on addressing critical risks [18]. More recently, the *design risks framework*, weaved many of these component tasks together to illustrate how expert designers practice the overall iterative planning process. In this framework, experts (1) use knowledge of design problem structure to focus attention on key areas of the project; (2) use knowledge of common risks to diagnose project risks; and then (3) use knowledge of iterative strategies to plan next steps that mitigate these risks [19].

However in practice, expert designers also do not necessarily follow such planning processes procedurally. Rather, they develop an intuitive understanding of how to adapt and execute their

plans as information about the problem space changes. Expert designers take an adaptive approach that is a "goal-directed, non-linear process that utilizes heuristic reasoning processes and strategies" [1]. For instance, an expert may be in the middle of executing their plan when they learn about a new critical risk in their design problem. At this point, they also realize that their plan is no longer focused on the most critical risk in the design problem. In response, they may adapt their planning by popping back out to visualize what they know so far about the design problem and re-assess current risks. Management literature refers to this adaptive approach as *strategic doing* [66]. In addition to honing the component skills of planning, experts also master this skill of planning process management, where they continually adapt and focus their planned next steps to advance the most critical parts of the design problem.

4.2.2 Existing Approaches to Scaffolding Planning Process Management

Existing design and design research pedagogy primarily focuses on training novices in the component skills of planning. For instance, there exist representational tools that scaffold designers to visualize the structure of their design problem [30, 2, 46, 46, 48], risk assessment tools that scaffold designers to identify critical issues in their designs [26, 43, 18, 71], and sprint tools that scaffold designers to decompose plans into actionable tasks [56, 90, 73].

Providing coaching or mentorship is an effective way for novices to learn to manage their planning process like experts do. Experienced design mentors with extensive domain knowledge and knowledge of planning strategies are skillful at managing their iterative planning process

[37, 26, 27, 19]. However, such experienced mentors are a limited resource in most design-research learning communities [90]. To help scale limited mentoring resources, socio-technical approaches use a combination of coaching and tools to scaffold novices in their design planning (e.g. canvases, planning meetings, kanban boards, stands, and design sprints) [90, 73]. Tools and social structures like sprint logs and planning meetings provide externalized and generalizable models that, when paired with expert coaching, have the potential to effectively and scalably train students to manage their planning process like experts.

However, our empirical findings suggest that even with such socio-technical designs, novices can still struggle to continuously monitor and adapt their plans to focus on these critical project issues. In the following section, we examine the socio-technical planning supports available in an Agile Research Studio [90], and present empirical findings from a needfinding study that explores challenges students face even trying to manage their planning process across these supports.

4.3 Challenges in Managing Planning Process in an ARS Ecosystem

4.3.1 Expanding the Planning Subsystem in the ARS Ecosystem

To understand the challenges students face when managing their planning process in an Agile Research Studio (ARS), we must first understand the expanded planning subsystem, and the desired subroutines (i.e. sequences of interactions) designed to scaffold students in design research planning. The planning scaffolding in an ARS is a combination of studio tools, social

structures, and processes. Together, these socio-technical supports are woven together as subsystems that guide students to execute the various components of the expert planning process (see Figure 35). We detail these component supports, and the subroutines that connect them below.

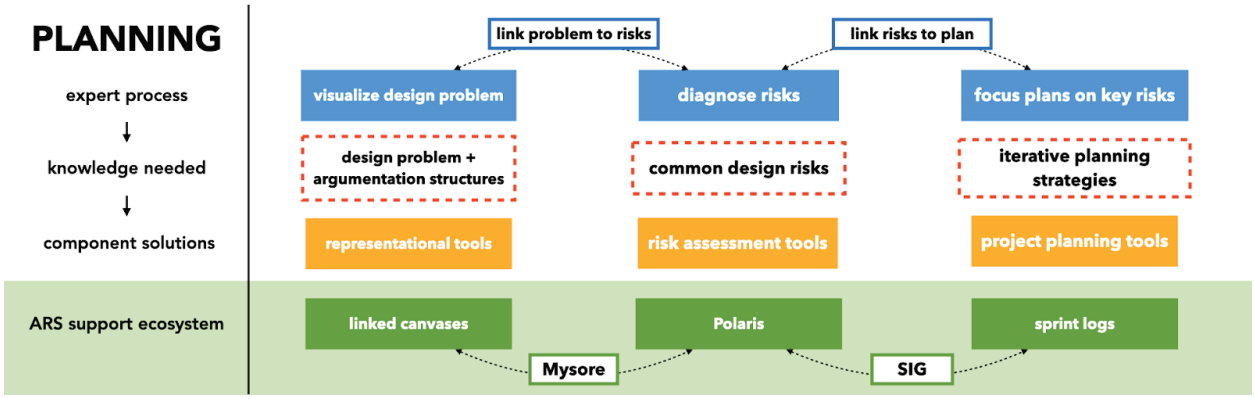


Figure 35. Expanded Planning Subsystem in ARS
Students in an Agile Research Studio have access to a series of supports (e.g. canvases, Polaris, mysore, sprint logs, and SIG meetings) that are designed to help them execute the expert planning process.

4.3.1.1 Linked Canvases to Represent Design Research Argument Structure

Design research work has layers of argumentation that justify both the practical design and the conceptual approach. Providing novices with low-level, detailed representations for each kind of argumentation, as well as high level, summative views across them can help novices visualize how experts skillfully weave these layers of argumentation together to form a cohesive design research narrative.

To help students externalize and revise their design-research argument structures like experts, ARS provides students with a series of tools called **linked canvases** to represent and update their current understanding of the design-research problem. Generally speaking, canvases are visual templates made up of (a) components that represent parts of the argument (e.g a design argument is broken down into an outcome, obstacle, characteristic, argument); and (b) links between these components that illustrate the relationship between the parts (e.g. how a design characteristic overcomes an obstacle to achieve an outcome).

Each canvas represents different views into the layers of argumentation in the work. For instance, we provide students with a *Practical Canvas* (18 sections covering 61 components and 44 links) and a *Research Canvas* (12 sections covering 40 components and 28 links) that scaffold them through the detailed argumentation structure behind a practical design and a conceptual approach, respectively. To help students link the argumentation between their practical design and their conceptual approach, ARS provides a higher level view called the *8-Pack* (1 section covering 8 components and 7 links), which gives a 4-component summary view of each the Practical and Research canvases, and how they link together. For example, the Practical Problem component of the design is linked to the Conceptual Class of Problems component that existing research has not yet addressed.

4.3.1.2 Polaris to Diagnose Structural Risks in Design Research Argumentation

To help students diagnose risks that experts commonly find in their design-research argumentation, ARS provides students with a tool called **Polaris** to evaluate their canvases for structural risks. These risks include: gaps (i.e. the components of the argument that are missing), lack of depth (i.e. the depth required for each component to form a convincing argument) and misalignment (i.e. the links required between and across components to form a cohesive argument). The Polaris tool embeds the expert argumentation structures from the canvas into a template for students to fill out -- visualizing the argument components and conceptual links between them. To further help students diagnose their argument structures, Polaris provides reflection prompts on lack of depth and misalignment issues for each component of an argument (e.g., "Does the obstacle clearly describe how it prevents the linked outcome from being reached?"). This helps novices procedurally focus their evaluation on identifying core structural issues within and across the components of their design argument, and avoid fixating on surface issues. While Polaris initially focused on the structural risks within the design argument itself (see Chapter 3), the tool now represents the full breadth of the Practical Canvas and Research Canvas detailed previously, along with the corresponding structural risks per component.

4.3.1.3 Mysore for Argumentation Feedback

To enable mentors to coach many students at once in argumentation practice, as well as give them argumentation specific feedback on their design-research work, ARS includes feedback venues called **mysore** as opportunities for guided practice on their argumentation throughout the

week. Mysore is a scaled feedback venue that allows mentors to coach many students on specific strategies for addressing critical risks and revising their argumentation. Mentors typically start mysore sessions by asking students to identify the riskiest area of their canvas to workshop during the session. This helps mentors to coach students to focus on "a slice" of their canvas argumentation that they agree is the riskiest, or most worthwhile to practice in the presence of their mentor at that time. Once the mentor and the student agree upon a specific structure, they begin to practice.

Students then independently practice alongside each other, and mentors begin to walk around the room, offering argumentation feedback. Students typically start by drawing the components and links of the structural slice on a whiteboard. In this way, mentors can easily recognize sections of the canvas, the risks they see in student argument structures, and what to focus on with students as they evaluate on the fly. When mentors recognize an opportunity, they can thus quickly guide students with a specific strategy to practice in a short check-in. As they rotate across students in the session, students then have the opportunity to practice those strategies, and iterate on their argumentation within the feedback venue itself.

4.3.1.4 Sprint Logs to Plan Agile Iterations

To plan out agile iterations in their design approach, ARS provides students with a tool called the ***sprint log*** to break down their sprint deliverables into smaller milestones. In the Sprint Log (see Chapter 2), students are able to record all tasks they plan to complete in their 2-week sprint,

update task progress throughout a sprint (e.g. mark tasks as in progress, complete, log hours spent on a task), and replan as needed. Students enter high-level deliverables, or *stories*, and the corresponding *tasks* required to complete those stories. To help students budget their time across tasks, students assign points to their stories and tasks to estimate the effort required to complete it. Students are expected to bring their sprint plans into SIG meetings to discuss planning feedback with their mentors (described below).

4.3.1.5 SIG Meeting for Iteration Plan Feedback

ARS provides students with the ***SIG (Special Interest Group) meeting*** feedback venue to get weekly feedback from their mentors and peers on whether or not their planned next steps are targeting the critical risks in their approach. SIG meetings bring together undergraduate students, graduate students, and faculty working on different projects in the same research area. Each SIG operates as a small-scale studio, initially led by a faculty mentor who fades over time as the graduate student mentor gains competencies in mentoring and research.

During a SIG meeting, students start by displaying their Planning View in their canvas, followed by their Sprint Log. These planning tools serve as conversational scaffolds for feedback, where students, mentors, and peers discuss the plan, progress, and strategies for overcoming any roadblocks. At the start of a two-week sprint, teams share the outcome of their last sprint and present their current sprint plans for review. Halfway through the sprint, teams present their progress and SIG members help devise strategies for overcoming blockers.

During SIG meeting, mentors focus their efforts on coaching students in their planning strategies, with a goal of making sure the student and mentor walk out with a clear understanding of what deliverables the student will complete by the next SIG meeting. For example, some planning strategies may be to (a) argue for how their sprint goals connect to their larger research goals; (b) clarify what the actual deliverables of the sprint would be, and what value it would have; (c) consider alternative plans; (d) assess whether tasks can feasibly be completed in time; and (e) generate strategies for better scoping down or slicing work.

4.3.2 Student Challenges in Managing Planning Process

To understand the process management challenges that students face as they practice planning in an ecosystem like Agile Research Studios, I conducted a series of 30 minute semi-structured interviews with a subset of studio members (n=10). To explore common challenges across user groups, we interviewed both graduate students and undergraduates. However, we primarily focused our interviews on students who had been enrolled in the studio for multiple quarters, allowing us to factor out other challenges related to being a student new to design research, planning practice, and the ARS ecosystem. Interviews were conducted in the middle of the term and mid-sprint. The goal of the needfinding was to understand (a) how different students would ideally manage their planning process during a week in an ARS (across canvases, Polaris, mysore, sprint logs, and SIG meetings); (b) what their planning process management actually looked like midway through their sprint that week; and (c) the hardest or most painful parts of managing their planning process.

Through a first round of coding the interviews, we identified two primary breakdowns across students managing their planning process. First, while students would receive planning feedback during the week, they struggled to integrate that feedback into a revised plan. Second, when students would fail to update their plans, they would default to executing their former plan, given time constraints. Below, we present a few student vignettes to illustrate these challenges, and why we believe they occur.

4.3.2.1 Students Fail to Integrate Mid-week Feedback into a Revised Plan

Participant P1, Naina, is a fairly experienced undergraduate ARS student (enrolled for 5 quarters) who is working on her project with a new partner. During the first half of their two-week sprint, they focused on preparing for their quarterly status update, a feedback venue where students can get project feedback from the entire studio during class. For their status update, they sought feedback on their needfinding insights from some user interviews they conducted. During their status update, Naina and her teammate received specific feedback from their graduate student SIG head, the faculty studio lead, and the general studio that they need to now focus on defining precise use cases for their system after having done all these user interviews.

Naina and her teammate entered the second half of their two-week sprint. A few days after their status update, they had the Mysore feedback venue -- an opportunity where they could get guided argumentation feedback on these use cases with the faculty studio lead. They also had their SIG meeting feedback venue, where they would have the opportunity to check in about these latest

risks and their planned next steps. We asked Naina to describe how their planning process actually played out that week.

Naina reported that she and her project partner didn't have a plan for how they would use the Mysore venue. While they did show up and leverage the practice venue, they dove in without a clear intention of using their needfinding results to workshop use cases with the faculty studio lead. Naina said the following in her interview:

> *"We didn't really think about the planning process when going into Mysore. It's more 'what are the things we need help on right now'. We have the plan all laid out at the start but new risks and blockers come up, and I think we weren't able to update our canvas and the sprint log to reflect that."*

Here, we see that Naina and her teammate missed out on taking the concrete planning strategies surfaced to them as feedback from their status update, and using that feedback to revise their plan. Consequently, they failed to apply those strategies during their scheduled Mysore practice venue later that week.

Later, Naina described how she and her teammate went into their SIG meeting feedback venue later that week, with an outdated canvas and sprint log that reflected their old risks from their risk assessment a week ago. Naina said

> *"For SIG meeting this week, the risk [in their sprint log] was about the status update from last week [insights from our user interviews]. But going into the second week, the*

> *risk is different now -- it's not takeaways from the user interviews or prepping for status update, it's defining the use cases. But when we pulled it up [the sprint log], we hadn't really updated it. We thought it was a thing you just do once a week, versus having a mindset of this being a dynamic process every time you come up with new risks and blockers. I internalized this idea of 'okay I do this [process] once every week'"*

Here, we see that Naina and her teammate -- despite having already discussed and agreed on their most recent project risks with their mentors -- missed out on getting planning feedback for how to address those risks during their SIG meeting feedback venue. So why do experienced students like Naina fail to integrate planning feedback, and revise their plans accordingly? Upon reflection, Naina described some of their difficulties as follows:

> *"Working with [a new student] this quarter especially -- the tendency is we spend more time on the tasks than we originally planned. When we planned, we were planning according to the 16 points [approximately 16 hours], and for each task we went longer -- so either spend more than 16 points or don't finish the tasks. We just don't have enough time to do all these things. And the hard deadlines like the [undergraduate research grant] due next Tuesday or status update [last Friday], knowing that we need to have a bunch of tasks and a hard deadline makes you feel a little bit anxious. Replanning would be nice if we could do it well and fit in our schedule. It means uncertainty. It's currently clear what we need to do next. But new plans introduce more uncertainty. We don't have a clear idea, well if these are the current stories [in the previous plan], how do I replan it?*

> *After I replan, can we still stay on track and achieve the goals we want to achieve? We know the risks but there is still uncertainty in a new plan. So our current approach is, as new things come up, just squeeze them into a new plan."*

Here, Naina describes how receiving planning feedback midweek usually means a need to revise plans midweek. When time already feels constrained, and they have carefully planned how to use the remaining time to complete the former plan, it feels like they have little room to pivot and execute a new plan. For Naina, and many other students, they often receive planning feedback from various feedback venues throughout the week. However, they express feeling unsure about how to take that feedback, think critically about whether the tasks laid out in their plan were in service of the new risks that were surfaced, and how to quickly make the changes necessary before the sprint ends. These findings suggest a need for lightweight ways to help students adapt their plans, in ways that are minimally disruptive to their plan execution.

4.3.2.2 Students "Default" to Previous Plan if Necessary Changes to Plan are Not Clear

Participant P9, Harry, is a second year graduate student in the studio. He is currently working on a research manuscript. He was unsure what might be the risks in his current draft, so he scheduled a meeting with his advisor last Friday to walk through sections of the paper and discuss the biggest risks. Harry described how he hesitated to replan his sprint after meeting with his mentor, despite having a new understanding of his risks, and where to spend his time.

> *So in my 1-1 with [my mentor] on Monday, we went through the risks. I didn't update my sprint plan yet. The [new] sprint plan doesn't feel like it's moving the research along in a concrete way, and I'd rather do those things. I think I'm more likely to sprint plan when I have clarity on what could help the work and what to do, and when I can connect what I care about in the research to the tasks related to it. This sprint: I need to work on my data analysis for my study, and the related work section and theoretical framework section -- but more related work because it feels riskiest right now. The hardest part is the revising of the plan. I do it way way less than anything else. I don't plan consistently but I do plan enough I think -- it's when I have to change the plan -- I almost never change my sprint, even when I need it. I think part of it, it's an acknowledgement of not doing something, and partly an acknowledgement of "having less" -- when you're replanning midway through the week, you only have 3 days or 2 days left to get it done. That is a lot scarier than a whole week of time. I think for both of them -- I don't mentally update the plan… and I don't revise it concretely either."*

Here, after meeting with his mentor, Harry recognizes a few critical areas in his paper -- the data analysis, the theoretical framework, the related work -- and even recognizes that the related work is likely the riskiest and a place to focus on next. Similar to Naina, Harry finds it challenging to revise his plan when the changes he needs to make feel less concrete, and there is little time remaining to change the plan and execute it. As a result, he ends up executing the previous plan, despite knowing that those risks and sprint plan are out of date. Without a concrete

understanding of how risks, deliverables, and next steps change, many students reported that they default to executing the same plan that they had scripted out, despite the change in their project understanding.

Participant P4, Ren is a 4th year graduate student in the studio. He is preparing to run a study to evaluate a system he has built. He has been spending time honing the argumentation behind the interface. During his SIG meeting, he received feedback from his mentor to think about the bigger picture -- if you have a goal to complete this paper by the end of the term, your interface arguments are just 3 paragraphs in a 10-page paper. What are the other requirements that also demand your time? As the week progressed, Ren struggled to find focus and hopped between sections -- looking for example abstracts, starting a blank document for his study design, etc. A few days later, Ren had a mysore feedback session, where he wanted feedback on the measurable outcomes of his system. He received the following planning feedback from his mentor.

> *"He basically said 'okay you need to be flying with your research and finish your interface arguments. It seems like you are just perfecting this one piece and that is maybe one slice of it. The bigger risk is discussing how you plan to conduct your study' -- Well, okay, that is useful in terms of planning. Mysore helped me see more of that ideal planning process. [I am] focusing too long on certain deliverables."*

During mysore, Ren received clear planning feedback that he should focus on the execution details of his study, so that he can run the study before the term ends. However, his plan

remained vague. He described how this lack of specificity may have contributed to how he struggled to monitor his plan execution during the week.

> *"Saying study design is a task [in my sprint log] and then starting a new document -- that becomes too overwhelming. I wish in my planning I was more articulate about -- okay, I understand this part which are my interface arguments and my process-based measures. But I haven't really articulated or measured any of the outcomes. So I might show that [the mechanism of my design] overcomes some obstacles in the study, but I don't know if I achieved the [user] outcomes. Really, I should have had in my mind this week -- forming the whole study design. I should have been much more outcome-focused -- but this week was just kinda like 'okay, gotta do study design, and more conceptual. Okay, so do more research canvas stuff, just follow the [canvas] prompts.' I wish I had really thought about -- 'Oh wait, I really need the outcome measures and I don't have a full study design.'"*

Ren then described why it can be challenging to check in with himself midweek using the sprint log tool, especially if the tool does not reflect the concrete changes he needs to make to his plan.

> *"I have a sprint and okay, I am gonna add up the points and once I have done all that the sprint is complete! Which is okay if I have a solid plan. But a much better way of checking in with myself would have been checking in about the deliverables throughout the week. My sprint plan this week is not super clear. I should be always questioning -- like, if I haven't hit my points halfway through the weeks, that's one sign yeah. But*

> *halfway through the week I should be able to check in and see -- okay did I get one of the two arguments done -- for me to see that vs the time totals. I knew I needed a deliverable that was presentable, but yeah -- am I getting closer to the updated understanding I need? and if I don't have that, that's risky. I am putting in half the week's work without being aimed at that specific intention [the deliverable]."*

Through these stories, we see that when students fail to capture and implement planning feedback, their planning tools do not clearly reflect necessary changes in project direction that were surfaced during the week. As a result, students can struggle with their task execution. For example, we saw students continuing with previous plans that are now out of date, appending new deliverables to old plans without reprioritizing or re-allocating the limited time they have, or replanning with vague risks, deliverables, or next steps that are hard to execute in the remaining time.

4.3.2.3 A Need for Process Management Scaffolds in Learning Ecosystems

Our needfinding demonstrates opportunities for designing process management scaffolds that help students easily capture and integrate planning feedback that ARS supports are designed to surface during the week. Such scaffolds can guide students and mentors to anchor their planning feedback conversations around the ways in which a student's plan may structurally shift. Did their risks, planned deliverables, and/or immediate next steps change as a result of the feedback they received? If so, how? And what does that mean for how they execute that plan in the

remainder of the sprint? If students walk away from feedback sessions with a clear and simple process for revising their planned next steps, we hypothesize that students will be more likely to integrate the valuable feedback they receive during the week.

To better support students, we can design systems that scaffold students to flexibly adapt and execute their planning across the component ecosystem supports that are designed to help them learn to maintain focus on the critical risks in their design-research problem. By training students to treat their plans not as fixed contracts, but flexible and disposable resources, we can encourage students to frequently adapt their original plans they make when new information arises, as experts do. In this way, students can then learn how to plan increasingly impactful iterations that adapt to advance the most critical areas of their design and research work, and effectively execute these plans across available ecosystem supports.

4.4. Compass: A Process Management Framework for Planning

To support students in managing their planning process as they execute, we introduce ***Compass***. Compass is a planning process management framework implemented as (1) an on-action dashboard and (2) in-action cues via Slack. The dashboard helps students assess the structure of their plan (e.g. whether their weekly deliverables will address their project risks, and whether their planned next steps move them towards completing their deliverables). The planning cues serve as in-action check-ins that cue students to execute their planning processes at opportune moments during the week (e.g. cueing students to revise their plans after a SIG meeting based on

mentor feedback, or checking in before a mysore feedback session to ask students if they are on track with their deliverables).

Together, the dashboard and cues are designed to help students manage their planning process by capturing and implementing the planning feedback surfaced through existing ecosystem supports during the week. Using the Compass framework, students are guided to reflect on and revise their plans continuously as experts do. The framework deliberately prompts students to compose plans that are structurally aligned, and to monitor and revise their plans as they receive new mentor feedback throughout the week.

4.4.1 On-action Dashboard

The on-action dashboard is a weekly plan template designed to scaffold students to structure their plans in the ways experts do. The dashboard is composed of (1) **the risks**, (2) **the plan**, and (3) **the status** (see Figure 36). In the *risks* section, students are asked to articulate up to three biggest project risks, in order of priority. While students can use Polaris to thoroughly assess any area of their practical or research canvases for risks, the view in the dashboard guides them to focus on, *at most,* the three most critical risks. They are then asked to identify where their risks are within the larger practical and research canvases. This section serves as an anchor, helping students conceptually link their plans back to their canvases, and understand how they are advancing their overall argumentation this week. In the *plan* section, students are then asked to commit to up to three clear deliverables that they will bring into the next SIG meeting. These

deliverables are visually aligned with the risks, to emphasize that their deliverables should be addressing the critical risks they have identified.

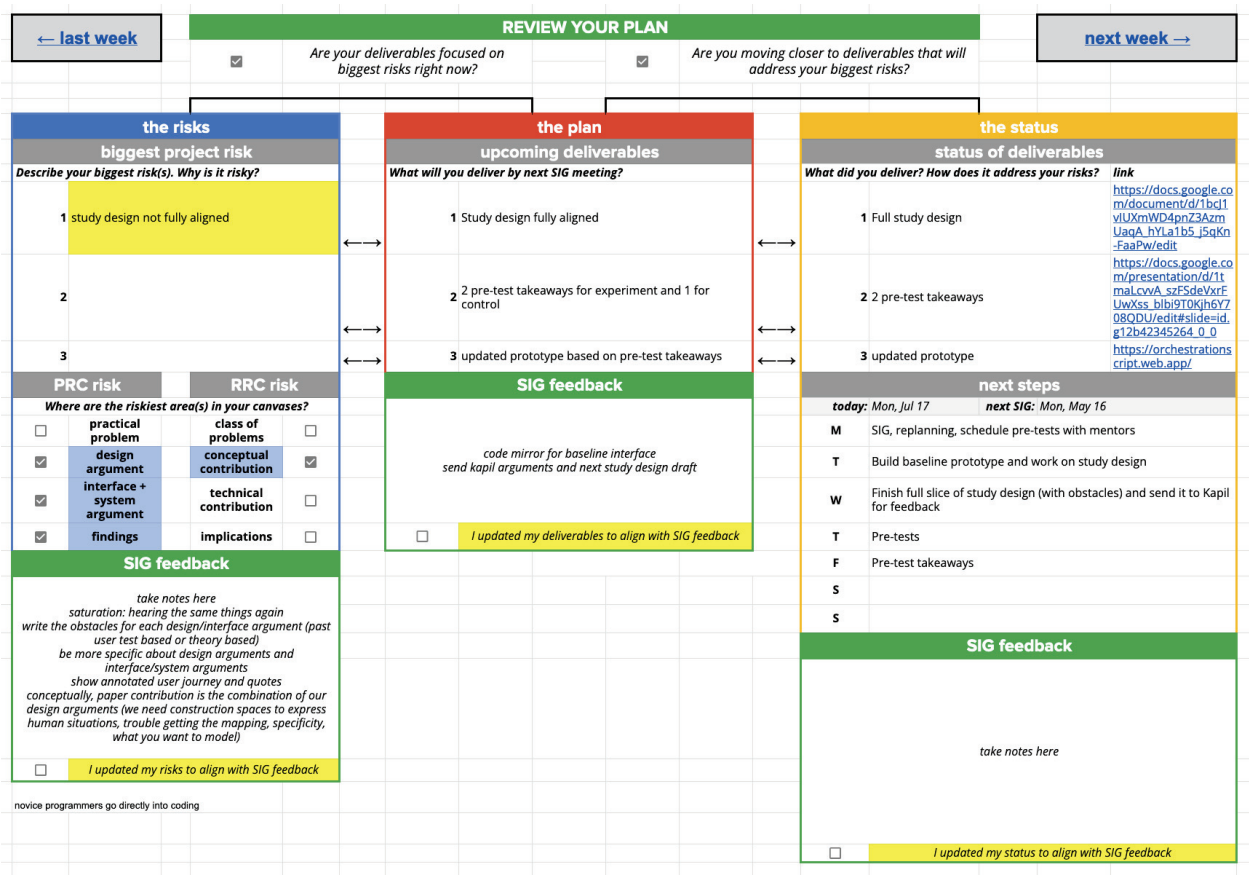


Figure 36. The *Compass* Framework - On-Action Dashboard
Compass is a planning process management framework implemented as (1) an on-action dashboard and (2) in-action cues via Slack. The dashboard helps students assess whether their weekly deliverables will address their project risks, and whether their next steps move them towards their deliverables. The planning cues serve as in-action check-ins that remind students to execute their planning processes at opportune moments during the week (e.g. cueing students to sprint plan before SIG meeting, asking them mid-week if they are still on track for their deliverables, or checking in after a mysore or SIG session to see if their plans need to change based on mentor feedback).

The *status* section is broken down into two parts -- next steps and status of deliverables. Under next steps, students are given a daily timeline of the week, where they can break down their deliverables and plan out how they will complete them before the next SIG meeting. Here, the focus is less on defining the tasks in detail, as one would in a sprint log. Rather, the purpose is to help students visually reflect on the structure of their week, and how they practically plan to get their deliverables done, pacing out their tasks in the time available. Further, it helps students situate their plan in the other weekly support structures that exist in an ARS ecosystem (e.g. if there is a mysore session on Wednesday where I can ask my SIG head for feedback on my arguments, I likely want a draft of my arguments ready before Wednesday.) Under the status of deliverables, students are instructed to link the concrete deliverables that they have completed. Similarly, these completed deliverables are visually linked to their planned deliverables, to help them assess whether what they completed is fulfilling the original plan.

Above these three components is a small reflection section that guides students to assess the alignment in their plan. One link connects the risks and the plan and has a corresponding reflection prompt that asks “Are your deliverables focused on the biggest risks right now?” The other link connects the plan and the status, and asks “Are you moving closer to deliverables that will address your biggest risks?” Students can check/uncheck whether or not they have fulfilled these requirements in their plan. By promoting reflection on the alignment between these plan components, the dashboard models how experts evaluate their own plans as the week progresses. What are the components they monitor as they execute a plan? And if one component shifts (e.g.

a deliverable changes), what impact does that have on the rest of the plan? Are they still addressing the same risks? How should they revise their next steps to account for the changed deliverable? In this way, the Compass framework seeks to embed the ways in which experts strategically evaluate and revise their plans into the dashboard, so that students can model a similar process as they execute their plans.

Students then use the dashboard view to get feedback on their plans during their weekly SIG planning meetings. Below each section, the dashboard provides a space where students are instructed to capture SIG feedback that is specific to that aspect of their plan (e.g. feedback on their risks). In SIG meetings, students and mentors have a shared view of this dashboard. Typically, mentors move left to right, starting the discussion with the risks students chose to focus on, then moving towards the deliverables they want to complete, ending with their next steps and how they plan to break down their week to complete their deliverables. While the dashboard view is up for everyone to see, students can capture feedback within the structure of the plan (i.e. as they are discussing risks, students can take down risk-specific feedback in that section). In this way, suggested plan revisions become explicit, and clearer to implement when students leave the SIG meeting. Further, as students and mentors move from section to section in the plan, mentors can monitor as students capture their understanding of the feedback as they go. This gives mentors the opportunity to correct any misconceptions before the meeting ends. Thus, students and mentors can leave the SIG meeting closer to collective agreement on the plan for the week.

4.4.2 In-action Cues

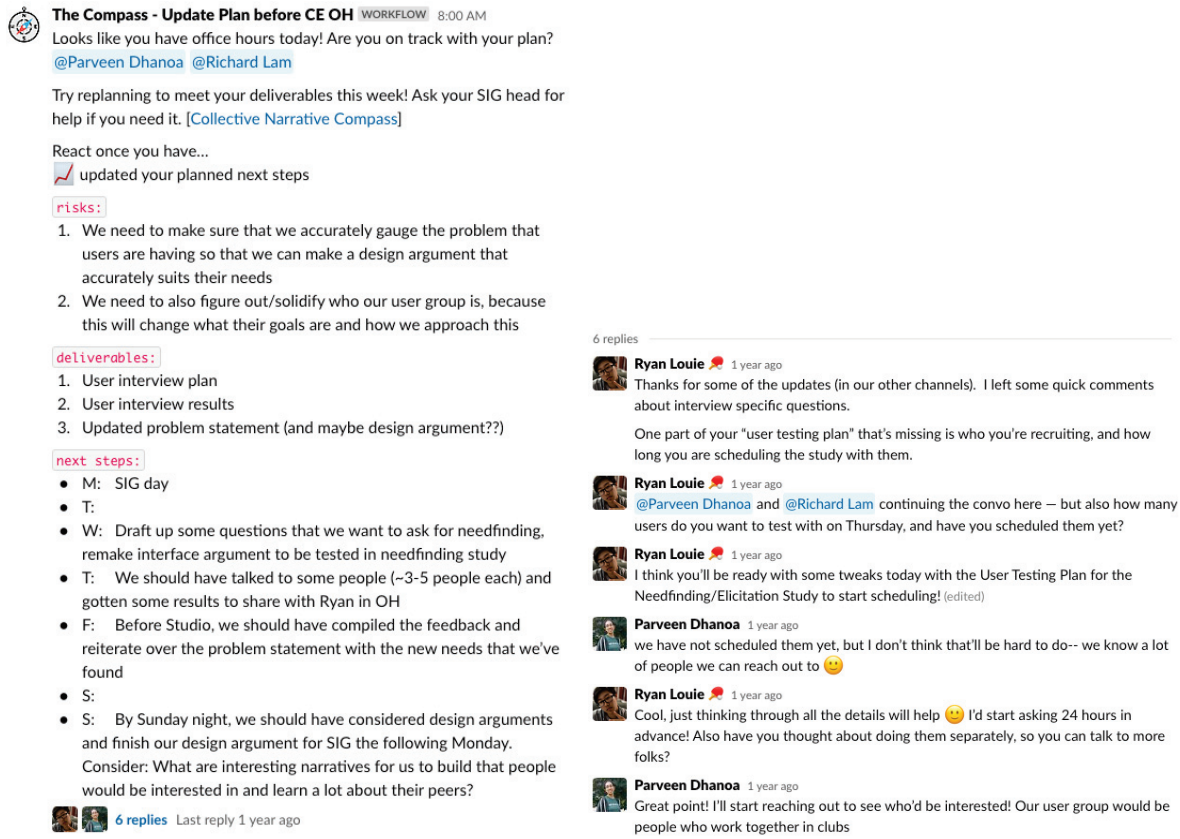


Figure 37. The *Compass* Framework - In-Action Cues
An example planning cue is sent to student's SIG channels via Slack reminders. It cues them to review their SIG risks and planned next steps, and check if they are on track for their deliverables or need to replan after their SIG meetings. Mentors can monitor how students use these cues, and even add in their own ad-hoc coaching as needed.

The planning cues (see Figure 37) are in-action messages sent in team and SIG Slack channels that cue students to execute their planning processes at opportune moments. Typically, a student's week starts with a SIG meeting, where students meet with their mentors to get planning

feedback. As the week progresses, students may have opportunities to get additional help in office hours with their mentors. Every Friday, students gather as a studio for mysore time, where they get argumentation feedback, as well as pair research and status updates. The cues are designed to integrate into these structures, as a way to promote moments of replanning as they relate to moments of planning feedback. Students receive cues at four specific moments during the week: (1) share last week's deliverables and revise their plan before SIG, (2) revise their plan after SIG, (3) revise their plan before office hours, and (4) revise their plan before studio.

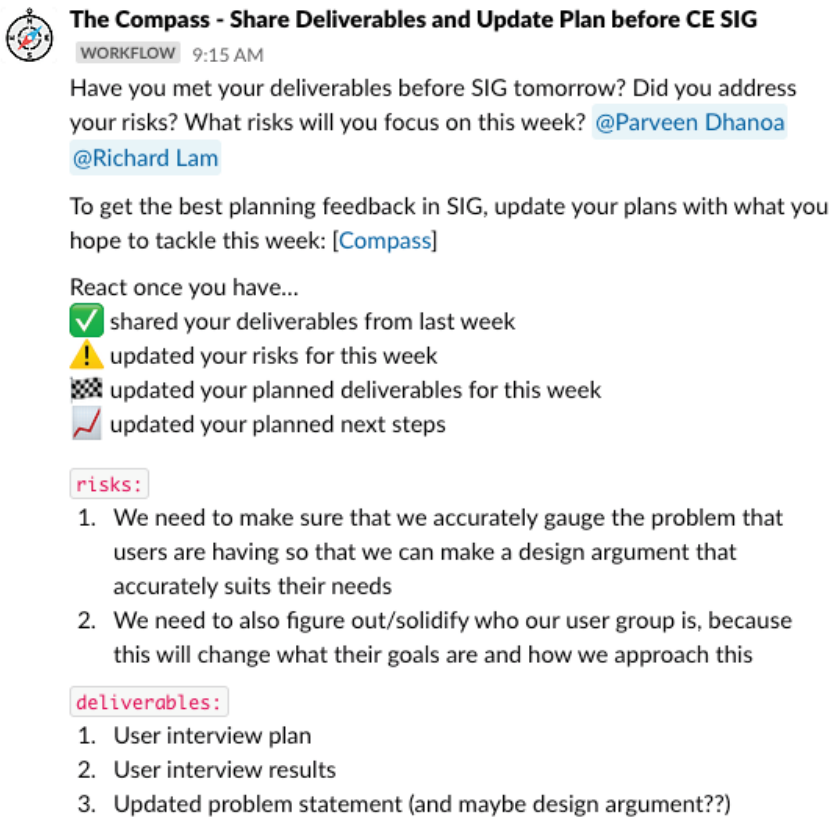


Figure 38. Compass Cues Before SIG Meeting

In preparation for SIG meeting, students are expected to share any completed deliverables from the previous week, and update their plan for the following week. One day before their SIG meeting, student teams receive a reminder to share their deliverables and update their plan (see Figure 38). This cue also carries the context from last week's plan (i.e. their risks and deliverables) as a snapshot to remind the team and mentor what the students set out to achieve this week. By reminding students a day before SIG, students can ask themselves if they have truly fulfilled their deliverables and addressed the risks they planned to address. If not, they have some time to either focus their deliverables, or can consider this as they plan for the following week. Having mentors present in the same channel as the cue means that mentors are also prompted to think about what they should expect going into their SIG meeting tomorrow (i.e. according to the plan, I expect the team addressed these risks, by completing these deliverables), and even review deliverables before SIG.

Following SIG meeting, Compass sends a cue within 10 minutes. Here, the cue reminds them that they just received lots of planning feedback during their SIG meeting, and advises them to make the concrete changes to their plan as soon as they can (see Figure 39). During the SIG meeting, students would have captured feedback within their Compass dashboard. This cue primarily recommends a best practice -- implement the feedback while it is fresh in their minds, before they go on executing their previous plan.

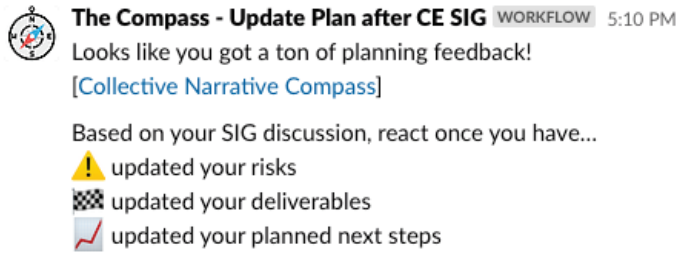


Figure 39. Compass Cues After SIG Meeting

As the week progresses, most students have the opportunity to meet their graduate student SIG head for office hours during the week. This is typically a valuable time for students and mentors to sync up on how their plan execution is going so far, get specific feedback on aspects of their project, or discuss any roadblocks they have encountered. One day before their office hours opportunity, Compass sends students a cue asking them to reflect on their plan execution so far this week (see Figure 40). This cue is shared in the project team slack, where the mentors are present. This cue includes a snapshot of the risks, deliverables, and next steps currently in the team's Compass dashboard. This gives mentors and students an opportunity to reflect on whether they are still on track to meeting their deliverables, and if they need to revise their plan midweek. The cue is strategically placed prior to the mentor's office hours, so that students and mentors can plan to leverage the support opportunity (i.e. discussing new information that might shift the plan). Compass sends a similar cue one day before the studio meeting on Fridays, where all students across SIGs gather as a community for practice, help, and feedback (see Figure 41).

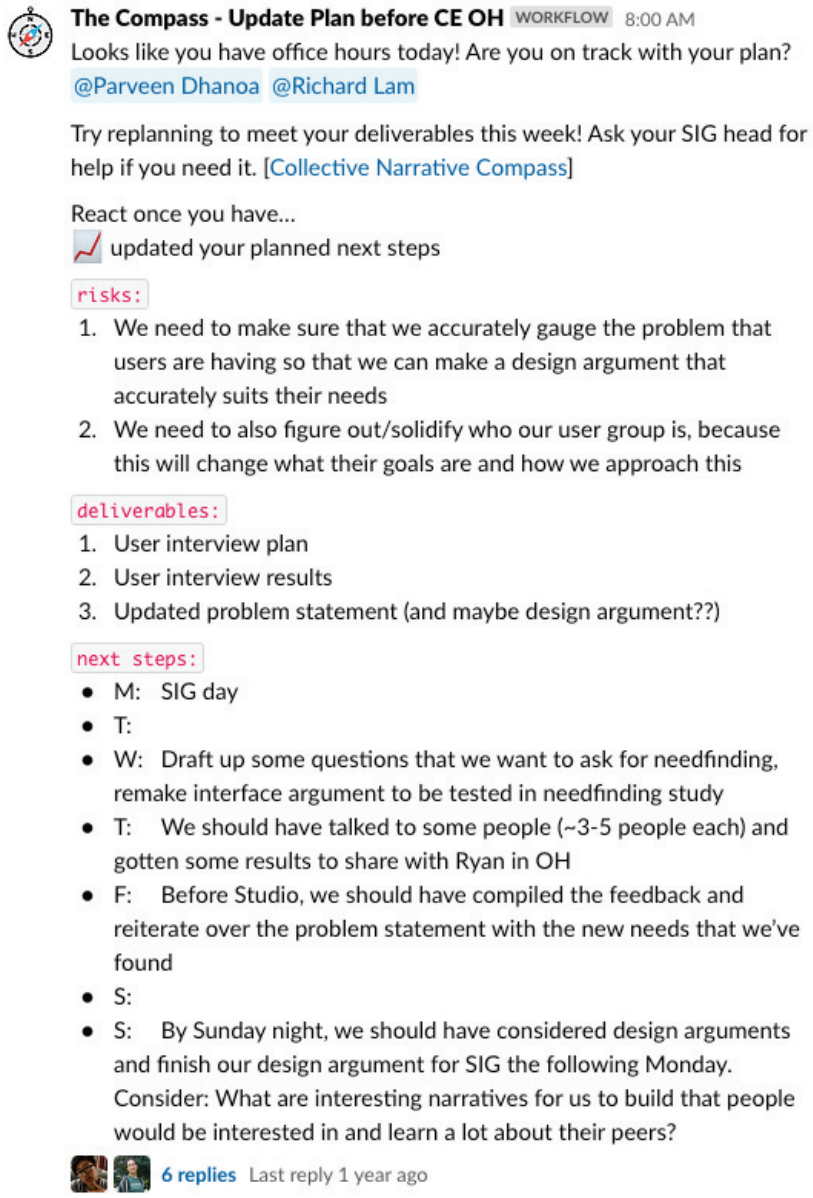


Figure 40. Compass Cues Before Office Hours

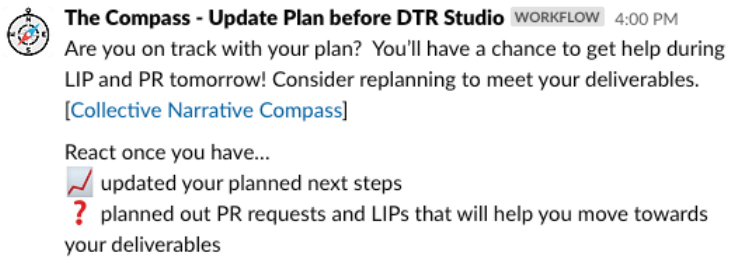


Figure 41. Compass Cues Before Studio Meeting

The combination of the planning dashboard and the planning cues provides students with on-action and in-action scaffolding -- where they are encouraged to assess and improve their overall process, and cued to execute that plan in ways that leverage the supports that surface planning feedback.

4.4.3 Compass Integration into an ARS Ecosystem

Importantly, Compass serves as a framework situated within the ARS learning ecosystem. As such, it is critical that the design integrates into the ecosystem of supports already available to students. Rather than introduce more tools and processes that may overwhelm a learner, Compass guides students through how an expert might manage their planning execution across these available resources.

The Compass dashboard is designed to integrate into the weekly SIG meetings, where mentors and students are already discussing planning feedback. Notably, SIG meetings are already heavily time-constrained. During a SIG meeting, mentors are already trying to coach and model planning strategies ranging from how to identify, specify, and prioritize a precise risk, to how to

define clear deliverables that are well-scoped, to strategizing about how to execute a plan well given the available resources that week. Further, mentors are typically coaching at least two teams in this hour. Beyond planning discussions, the mentors also spend time syncing up with students to understand how their project execution has gone over the past week, and what hurdles they may face moving forward. These are just a few of the many strategic ways in which mentors use SIG meetings with their students. Consequently, mentors also deliberately scope out certain discussions from the SIG meeting, typically routing them to other supports in the ecosystem. For instance, Coaches in SIG meetings do not want to replan the whole sprint with students. Coaches also do not want to get into low-level details about the students' design and research arguments. Rather, they want to model ways in which a student can enact certain planning strategies when managing their own plans, typically by focusing on one or two core examples (i.e. describing why one risk is important, or workshopping one sprint story to be a clear deliverable).

To help structure SIG meeting discussions, the Compass on-action dashboard is designed to serve as a structured conversational scaffold for mentors and students to discuss their plans. What are the critical pieces one needs to consider when setting up their plan? Compass draws focus to the risks, the deliverables, and the planned next steps laid out across the week. Using this visual scaffold during the meeting, mentors and students explicitly discuss these structural pieces, and how one change (e.g. a changed risk) affects another component of the plan (e.g. how the corresponding deliverable changes). The dashboard gives a shared view of the pieces experts

typically consider when managing their plan. Using the dashboard in SIG meetings models these structural pieces of an expert plan, and the expert skill of checking how they align, to students.

Similarly, the cues are designed to integrate into the regularly used communication channels on Slack, where mentors and students already engage in ad-hoc coaching and feedback. The intention of the cues is to go beyond a simple bot or reminder system. Rather, they are designed to fit the context of the week. For example, each team may have different SIG and office hours times, so the cues are sent relative to these feedback venue times. Further, the cues are designed to prompt students within shared Slack spaces that mirror the social structures of the studio. For instance, the mid-week check-in cue that is sent prior to office hours is sent in the team channel where the mentor is present. As such, mentors can jump in and give ad-hoc feedback, respond to student questions, or ask for more information about their status. As another example, the cues to update plans after SIG are sent out to the SIG channel, where coaches can easily address all teams to make the changes discussed in the meeting a few moments prior. Students can also send their revised plans here, prompting their peers to revise their own plans and do the same, engaging in socially shared regulation.

4.5 Compass Pilot Study

We tested Compass in an 8-week pilot study with six undergraduate design-researchers who led four HCI research projects, mentored by three graduate students. The primary goal of the pilot

study was to understand how Compass might better scaffold students to effectively manage their planning process throughout the week, as compared to their prior planning process management, sans Compass. Our core hypotheses were as follows:

1. Students will revise their plans more frequently throughout the week when using Compass (as opposed to the beginning and end of the week).

2. The plans that students compose and execute with Compass will be more structurally aligned throughout the week (i.e. their risks, deliverables, next steps will be aligned as the week progresses).

3. The plans that students compose and execute with Compass will be more aligned with mentor feedback throughout the week (i.e. their plans will incorporate the mentor feedback surfaced through ARS ecosystem supports).

We detail our pilot study methodology below.

4.5.1. Study Setting and Participants

To assess how Compass might support effective planning process management across available resources in a learning ecosystem, we tested the framework with an Agile Research Studio (Zhang 2017), where students learn to lead systems HCI research with the support of faculty and graduate student mentors. Student participants were undergraduate Computer Science majors, and had been working on their projects between 0 and 4 terms. Graduate student mentors (or SIG

heads) were HCI PhD students that had between 3 and 15 terms of experience mentoring HCI research.

	student researcher : # terms doing design research in an ARS	SIG head : # terms mentoring design research in an ARS
Team A	Student 1: 3 terms Student 2: 3 terms	Mentor 1: 11 terms
Team B	Student 3: 1 term Student 4: 1 term	Mentor 2: 14 terms
Team C	Student 5: 0 terms	Mentor 3: 3 terms
Team D	Student 6: 0 terms	Mentor 3: 3 terms

Figure 42. Compass Study Participants

Student teams were selected from three different SIG groups, with three different SIG head mentors to account for differences in mentoring process or style between different SIG mentors (see Figure 42). The selected student teams in these SIGs represent a spectrum of expertise with design-research and the ARS ecosystem -- 2 senior PhD students, 1 novice PhD student, 2 experienced undergraduate students (i.e. 3 terms in the program), 2 somewhat experienced undergraduate students (i.e. 1 term in the program), and 2 novice undergraduate students (i.e. first quarter in the program). Diversifying participants in this way helps mitigate against a risk of an “expertise effect”, where one team may show better process management simply because they are more experienced with design-research planning, or leveraging the ARS ecosystem supports.

4.5.2. Study Procedure

During the 8-week pilot study, students and mentors were instructed to use the Compass dashboard and cues to manage their planning process. To mitigate against any confounding factors related to starting the term, especially for students new and orienting to the studio and their project, we focused our study on weeks 3-9. All participants used Compass during weeks 3-5 (Sprint 2, Week 3 - Sprint 3, Week 5), as a way to train students and mentors on how to use the Compass dashboard and cues in their planning process. During weeks 6-8 (Sprint 3, Week 6 - Sprint 4, Week 8), all participants continued to use the Compass framework, with the exception of one week, where each team had the Compass framework removed from their practice to see how their process management would change in the absence of the scaffold (see Figure 43).

week	Team A	Team B	Team C	Team D
S2W3 (Apr 18)	compass	compass	compass	compass
S2W4 (Apr 25)	compass	compass	compass	compass
S3W5 (May 2)	compass	compass	compass	compass
S3W6 (May 9)	compass	compass	NO COMPASS	compass
S4W7 (May 16)	compass	NO COMPASS	compass	compass
S4W8 (May 23)	NO COMPASS	compass	compass	NO COMPASS
S5W9 (May 30)	compass	compass	compass	compass
FINALS WEEK (Jun 6)	compass	compass	compass	compass

Figure 43. Compass Study Procedure.

Student teams did not know which week they would have Compass removed until the day before their SIG meeting, in time for them to compose their plan for the week. Mentors were not

informed which week their students would not use Compass until the start of the SIG meeting itself, where they would present their plan for feedback. All participants returned to using the Compass during weeks 9-10 (Sprint 5, Week 9 - final exam week).

Each week that students used Compass, they were instructed to use the Compass dashboard to compose their plan for the week, prior to their SIG planning meeting. In the Compass condition, students were welcome to use existing planning tools in the ARS ecosystem (e.g. their Canvases, Polaris, Sprint Log). However, students and mentors were instructed to use the Compass dashboard to start their planning discussion at the start of each SIG meeting, and as it was useful through the rest of the meeting. Students were further instructed to take their SIG feedback notes within the Compass dashboard, using the Mentor Feedback sections.

Students and mentors also received Compass cues in their SIG and project channels at opportune moments to update their plan throughout the week (to share deliverables and compose their plan before SIG meeting, to revise their plan after SIG meeting, and to revise their plan before leveraging scheduled help venues (office hours and studio). During the week that students did not have the Compass dashboard or cues, they were instructed to use their usual planning tools to compose their plan for the week, as they previously used to, prior to their SIG meeting. Other than the reminder to not use the Compass dashboard a day before their SIG meeting, students did not receive any cues as part of the control condition.

In addition, students and mentors participated in weekly retrospective interviews to recount the details of how they executed their planning process from the previous week. As mentors and students tend to forget the details of each week by the end of term, these interviews gave us the opportunity to further investigate the details of what happened in each week in terms of plan execution, from the perspective of the students and their mentor. Mentors were further requested to assess the quality of their student's planning process at the end of each week, within 24 hours of the end of week (marked by the weekly SIG meeting). To minimize a risk of mentors revising how they planned to coach their students after formally assessing their student's planning process, we had mentors complete their assessments at the end of the week, rather than having them assess at different moments during the week.

4.5.3. Data Collection

To get detailed insight into how students managed the planning process week to week, we collected the following data.

1. Recordings of weekly SIG planning meetings were collected via Zoom. The primary author was present to conduct observations during the SIG meetings, but remained passive/non-participatory. To understand how students and mentors used the Compass dashboard as a conversational scaffold during the meeting, SIG members were instructed to screen-share any views that were displayed during their meetings (i.e. any documents, tools, etc that were used during the discussion).

2. Weekly planning tool usage data was collected from weeks 3-10, including detailed revision histories for each team's Compass dashboard, Planning View, Sprint Log, and Design Log when relevant (i.e. when students captured mentor feedback in SIG the week they did not have Compass).

3. Weekly Slack interaction data was collected from weeks 6-10, including any posts, replies, and reactions from team and SIG channel, as well as relevant planning discussions over direct messages that were shared by participants in their weekly retrospective interviews (see below).

4. Weekly retrospective interviews were conducted with students from weeks 6-9, where students were guided to recount their planning process from the past week. Interviews were approximately 45 min long. The primary author asked students to replay the week by visually walking through the revision histories of their planning tools, relevant Slack interactions, and discussions from their feedback sessions at different venues throughout the week.

5. Weekly retrospective interviews were also conducted with mentors from weeks 6-9, where mentors are guided to recount their coaching as it related to planning from the past week. Interviews were approximately 45 min long. Mentors were asked to replay the week by visually walking through any planning tools and relevant Slack interactions with

their students. They were also prompted to recount their feedback sessions as they monitored and coached students throughout the week.

6. Each week, mentors were asked to complete mentor assessments (from weeks 6-9), where mentors evaluated the quality of the plans, plan revisions, and plan execution of their students at the end of each week. To standardize the assessment process across mentors, they were asked to spend no more than 30 minutes reviewing each student team, shortly after each SIG meeting.

7. Finally, mentors and students completed end-of-term surveys and 20 minute final interviews to reflect on their overall experience with planning process in both conditions.

4.5.4. Data Analysis

To analyze how frequently students are revising their plans during the week, we review how their revision activity was distributed across the week, based on logged revisions in their planning tools. Specifically, we define a "revision session" as a continuous block of time where students are revising one of their planning tools. For instance, 35 revisions done from 1-2p count as one revision session. We then plot the timing of these revision sessions relative to their weekly timeline, and the venues where they are likely to receive planning feedback (e.g. SIG planning meetings, office hours, mysore sessions, etc). We look at any and all planning revisions that may have occurred across planning tools each week (e.g. the Sprint Log, the Planning View, Compass, Design Log (when used to capture SIG feedback)). For each team, we analyze the frequency and distribution of revision sessions each week, and compare both conditions. We

coded individual revisions and revision sessions relative to when Compass cues were sent, and when planning feedback venues occurred (e.g. SIG meetings, office hours, and mysore). With or without the Compass, we expect that students will revise their plans before their SIG meeting, to prepare to discuss their completed deliverables, and their plans for the following week. Thus, we focus on "midweek" revisions that may occur relative to other planning feedback venues (i.e. during SIG, after SIG, before office hours, after office hours, before studio, after studio).

To analyze the quality of the revised plans each week, we use the weekly mentor assessments to identify whether plans are (a) internally aligned in structure and (b) externally aligned with mentor feedback they received. At the end of each week, mentors were given rubrics and asked to evaluate the progression of their students' plans that week.

Mentors evaluated the structural alignment of student plans at 6 different assessment moments -- (1) the alignment of risks and goals before SIG, (2) the alignment of goals and next steps before SIG, (3) the alignment of risks and goals after SIG feedback, (4) the alignment of goals and next steps after SIG feedback, (5) the alignment of goals to deliverables at the end of the week, and (6) the alignment of risks to deliverables at the end of the week. Plan contents were extracted out of the tools and put into a standard table so as not to bias mentors in their weekly evaluations. Mentors reviewed structural alignment of plans by rating the alignment between two components on a scale of 1-5, where 1 meant "not aligned at all" and 5 meant "highly aligned". Using this data, we determined the degree to which the revised plans of a team align with the planning feedback they received from their mentor earlier in the week.

Mentors also assess the plans to see if the revised plans were aligned with the feedback that mentors gave students in their SIG meeting. For example, are the critical risks students identify aligned with what the mentor believes are the critical risks; are the planned deliverables that students identify aligned with what the mentor thinks their deliverables should be that week?; or are students enacting the planning strategies discussed with their mentors during SIG and Mysore (e.g. slicing or scoping their work). At the end of each week, mentors also evaluated the the degree to which student plans aligned with their feedback at two different assessment moments -- (1) the alignment of the team's revised plan with the mentor's planning feedback after SIG meeting and (2) the alignment of what the team actually delivered with what the mentor and team initially agreed they should deliver at the end of each week. Plan contents and feedback notes were extracted out of the tools and put into a standard table so as not to bias mentors in their weekly evaluations. Mentors rated alignment of revised plans to the mentor feedback captured by students on a scale of 1-5, where 1 meant "not aligned at all" and 5 meant "highly aligned". Further, mentors qualitatively documented the ways in which a team's completed deliverables aligned or misaligned with what the mentor and students agreed they should deliver by week's end.

Through this data, we will be able to measure the frequency at which students are able to revise their plans, the quality of those revisions in terms of structural alignment of students plans, and alignment of the plan revisions with mentor feedback. In doing so, we will be able to see how a student's understanding of their planning process flows within and across ARS planning supports

that already exist in the ecosystem -- both in the absence of planning process management scaffolds and with the scaffolding in place.

4.6 Compass Findings

4.6.1 Students revise plans midweek when using Compass

	compass weeks (average)	non-compass week
Team A	3.00	1.00
Team B	3.17	0.00
Team C	3.33	1.00
Team D	2.17	3.00

Figure 44. Summary of Midweek Revision Sessions with and without Compass

We found that on average, students had more midweek revisions the weeks they used the Compass (see Figure 44). By examining revision sessions per team (see Figure 45), the data shows that in the week without Compass (in pink), teams generally engaged in little to no plan revision sessions throughout the week. However, the weeks where students used Compass (in blue), the data show teams engaged in revision sessions throughout the week, relative to the venues where they typically receive planning feedback. Notably, the data also shows that these midweek revisions occurred before and after these venues.

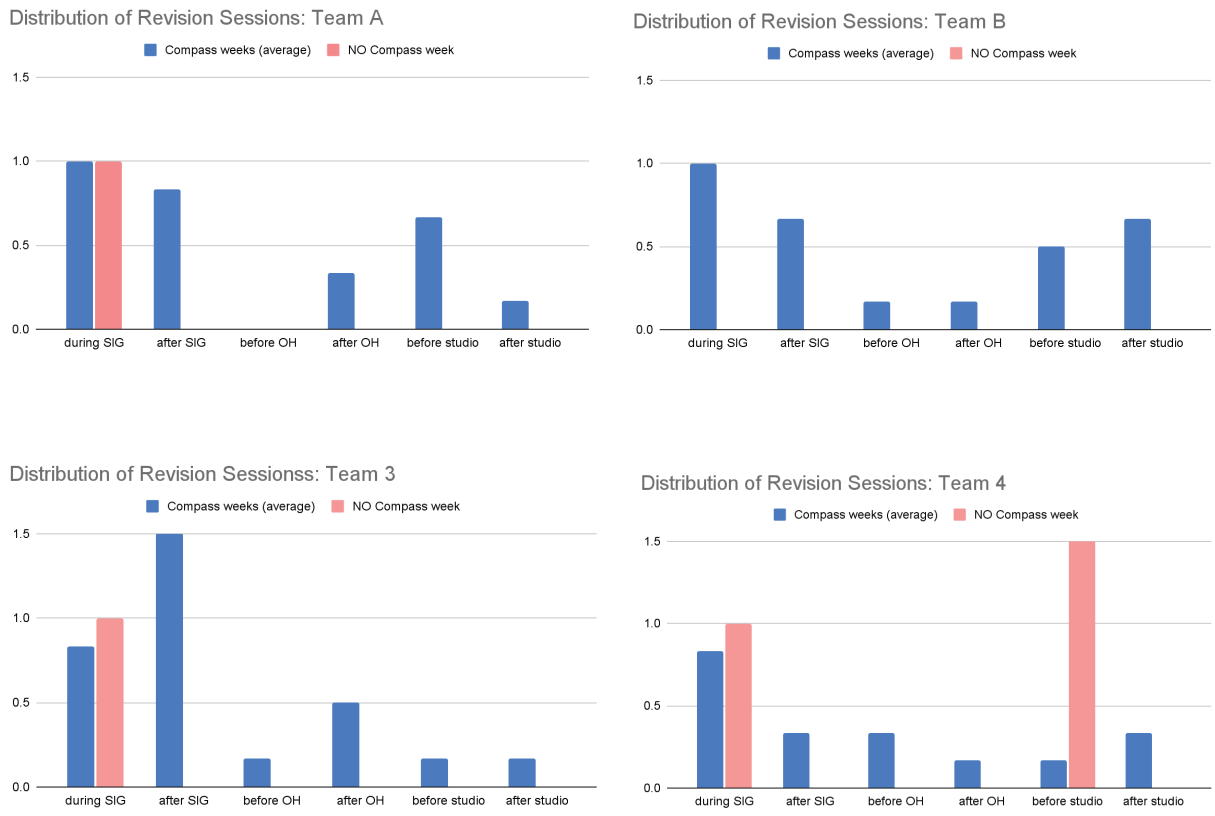


Figure 45. Per-Team Midweek Revision Sessions with and without Compass

Likely, these pre- and post-venue plan revisions reflect students replanning to better leverage the feedback opportunity, and students incorporating feedback from the venue into a revised plan afterwards. These findings suggest that using the Compass process framework may have prompted students to monitor their plan execution during the week using the dashboard, and to revise their plans after receiving cues around key venues where they likely received some planning feedback.

It is important to note that on the week without Compass, Team 4 engaged in several revision sessions before the studio venue. In the weekly retrospective interviews with both the team and mentor, we discovered that this week also happened to be the week that the team was presenting their quarterly status update and running a user test with the whole studio. Typically the week where students have a status update in studio, students heavily leverage mentor office hours and revise their plans to prepare for and make the most use of this one-a-term feedback opportunity. Thus, the peak in revision sessions for Team 4 is due to their studio presentation the next day.

	total	compass weeks (average)	non-compass week
revision sessions:	179	5	4
revisions:	1584	58.04	59.00
duration (min):	1405	50.04	55.75

Figure 46. Summary of Revision Data

Overall, the four student teams engaged in a total of 179 revision sessions over the 7-week period, totaling 1584 individual plan revisions, and 1405 minutes (23.43 hours) of time revising plans (see Figure 45). Interestingly, when examining plan revisions in terms of quantity (i.e., number of revision sessions, revisions, duration of revising per week), the data shows little difference between the weeks students used the Compass framework vs the weeks they did not have access to it.

	S2W3	S2W4	S3W5	S3W6	S4W7	S4W8	S5W9
Team A							
revision sessions	4	5	7	4	6	4	4
total revisions	34	36	52	30	31	31	26
total duration	44	43	54	24	56 min	17	26
Team B							
revision sessions	4	4	8	9	3	7	4
total revisions	42	46	77	100	62	103	37
total duration	25	43	62	147	53	81	29
Team C							
revision sessions	10	8	8	7	13	7	6
total revisions	59	131	79	56	79	60	50
total duration	62	76	38	57	70	48	41
Team D							
revision sessions	8	9	8	3	6	6	7
total revisions	86	67	52	29	42	42	45
total duration	61	44	14	67	39	77	63

Figure 47. Detailed Revision Data Per Week, Per Team

This is true when averaging weeks with Compass, but also in the week-to-week activity per team (see Figure 47). In reality, students who are regularly monitoring and revising their plans may not necessarily revise their plans *more.* This data underlines that a more precise measure is the distribution of revision sessions across the week, relative to the venues where students typically receive planning feedback from the supports in the ARS ecosystem.

4.6.2 Students plans were more structurally aligned when using Compass

	Team A	
	compass weeks (avg)	non-compass week
risks to goal (before SIG)	4.00	3.00
goal to next steps (before SIG)	3.00	3.00
risks to goal (after SIG)	4.00	3.00
goal to next steps (after SIG)	4.00	3.00
deliverables to goal (before next SIG)	4.67	3.00
deliverables to risks (before next SIG)	4.33	4.00

	Team B	
	compass weeks (avg)	non-compass week
risks to goal (before SIG)	4.67	3.00
goal to next steps (before SIG)	4.00	3.00
risks to goal (after SIG)	4.67	3.00
goal to next steps (after SIG)	4.67	3.00
deliverables to goal (before next SIG)	4.00	3.00
deliverables to risks (before next SIG)	4.67	4.00

	Team C	
	compass weeks (avg)	non-compass week
risks to goal (before SIG)	3.17	2.00
goal to next steps (before SIG)	3.00	2.00
risks to goal (after SIG)	3.83	2.00
goal to next steps (after SIG)	3.00	2.00
deliverables to goal (before next SIG)	4.67	4.00
deliverables to risks (before next SIG)	4.17	3.00

	Team D	
	compass weeks (avg)	non-compass week
risks to goal (before SIG)	2.00	2.00
goal to next steps (before SIG)	1.00	2.50
risks to goal (after SIG)	2.33	2.00
goal to next steps (after SIG)	1.00	2.50
deliverables to goal (before next SIG)	2.67	3.00
deliverables to risks (before next SIG)	2.67	2.00

Figure 48. Structural Alignment of Plans with and without Compass, per Team

Findings show that Mentors of teams A, B, and C considered student plans to be more structurally aligned the weeks they used Compass to manage their planning process. Figure 48 compares alignment data with and without Compass for each team, color-coded to reflect the alignment scale (red = not aligned at all, blue = highly aligned). For teams A, B, and C, the figure demonstrates that plans were consistently more aligned in the weeks students used Compass. For teams A and B (which were composed of more experienced students), mentors

generally considered components of student plans to be “aligned" or “highly aligned” with Compass, and “somewhat aligned” or “aligned” without Compass. For team C (which was composed of novice students), the mentor generally considered plans to be “somewhat aligned” or “aligned” with Compass, and “not aligned” or “somewhat aligned” without Compass. Notably, we see slightly higher alignment scores for Team D in the week they were without Compass. In retrospective interviews with Team D and their mentor, we learned that the student struggled to proactively plan and use planning supports throughout the term, with the exception of the week of their status update presentation to the studio, which was also their week without the Compass framework. As a result, we see higher alignment scores that week.

A closer look at student plans week to week demonstrates a similar story, where teams A, B, and C generally maintained plans that were more structurally aligned the weeks they used Compass (see Figure 49). The following graphs depict how plan alignment looked for each team, across the 6 assessment moments during the week (i.e. (1) the alignment of risks and goals before SIG, (2) the alignment of goals and next steps before SIG, (3) the alignment of risks and goals after SIG feedback, (4) the alignment of goals and next steps after SIG feedback, (5) the alignment of goals to deliverables at the end of the week, and (6) the alignment of risks to deliverables at the end of the week). Each line represents one week of student planning, with the week without Compass highlighted in pink. Teams A, B, and C all composed and maintained plans that were generally more aligned in the weeks where they used the Compass framework to scaffold their planning process management. These findings demonstrate that, irrespective of the level of

experience of students, Compass was able to help students structure plans that were better aligned. Notably, the findings not only show that students could compose structurally aligned plans prior to any feedback, but that their plans remained aligned throughout the week, in the weeks they used Compass. These results suggest that process management frameworks like Compass can not only help students come up with sound plans (i.e. next steps that move them towards a deliverable that addresses a critical risk). They also suggest that such scaffolding can help students revise their plans throughout the week, such that their next steps still move them towards the deliverable and focus on their critical risks.

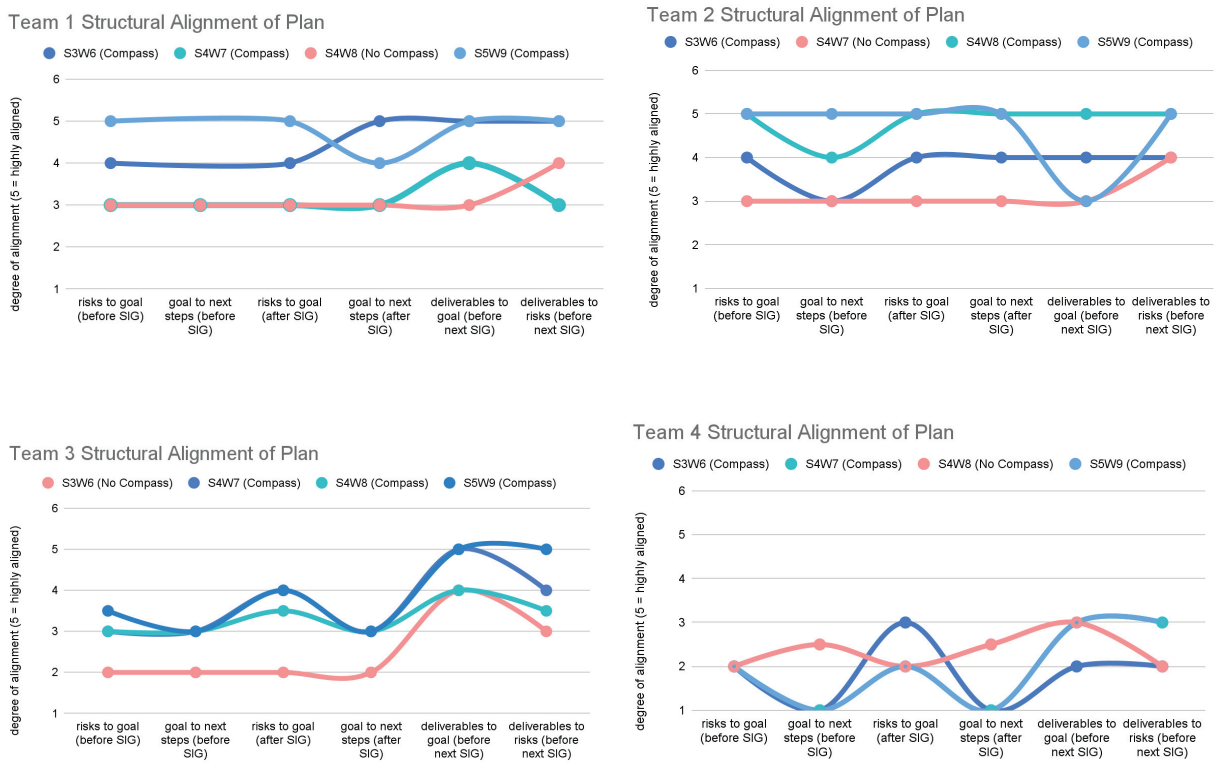


Figure 49. Structural Alignment of Plans Across Each Week, per Team

To understand the ways in which mentors recognized structural alignment within a team's plan, we asked mentors to also qualitatively assess the structural alignment of student plans each week. Mentors documented any alignment, and any misalignment they noticed in the initial plan. Below, we present a vignette from Team B, a team composed of senior students, to illustrate how alignment in their plan shifted with and without the Compass framework in place.

4.6.2.1 Structural Alignment of Plans for Team B with and without Compass

Team B is working on a system that seeks to build social connection between individuals by detecting and surfacing shared situations between them (e.g. connecting students who are all approaching midterm season). Team B has practiced the planning process on a weekly basis for multiple terms. Their team used Compass scaffolding to manage their process for the last four weeks, and had Compass scaffolding removed for one week near the end of term, Sprint 4 - Week 7. Figure 50 shows a snapshot of the team's plan the week without Compass, prior to their SIG planning meeting. This plan was extracted from their planning tools (i.e. Planning View, Sprint Log) and presented to mentors for assessment:

Risks	**Sprint Focus**	**Sprint Stories + Tasks**
Vague obstacles / terms / arguments in the Mysore Problem Statement and Design Arguments. Also a lack of verification on the practical side in terms of our new ideas for what the connection goals are. Also, updating prompts to be parallel doesn't really sound	We are going to do a lot more research when it comes to understanding which prompts might be more meaningful for the community, and also consider how we might be able to leverage context with that	Revise/Defend Problem Statement & Finalize Design Argument -- Make sure that we are moving forward on solid ground (e.g. there is alignment across our arguments) Identify vague terms in Problem Statement & Obstacles (e.g. what constitutes a community? what truly constitutes shared context? Is it just time/location or is there something more

super conceptually interesting yet.		conceptual?), research, define, and justify them Update the design argument, change design argument to include citations and better examples Conduct Lo-Fi Preliminary Testing -- Make sure our updated idea about shared community reflection actually achieves our user outcomes (parallel prompts + relevant context) -- A user test complete with intentional follow up questions ready to run by next week. Research more about prompts that provoke deep group reflection (e.g. what conceptually makes Before I Die relevant?) Come up with a set of parallel prompts and rationale for why they might be grouped into certain contexts that are relevant to potential test users Design a LoFi, manually controlled Slackbot Prototype Test (Complete with procedure, intentional follow up questions, etc.) for DTR community Research into why asking certain reflection questions at certain locations/contexts is more meaningful Researching More Artifact Methods -- We want to make sure that our contribution is beyond "better prompts," even if it works. Research why better questions may be insufficient / think of more technical ways to extend the conceptual contribution (e.g. auto-generation of deep questions based on context...!? Not the most important thing right now.)

Figure 50. Team B Example Plan Prior to SIG Meeting (Without Compass)

Upon reviewing this plan, their mentor noted the following alignments: “In the risks and sprint stories, they are aligned on the idea of making their problem statement and desired outcomes less

vague. They are also aligned on having a plan to test/verify their updated idea (design argument) about shared community reflection actually achieves our user outcomes." Here, the mentor noted the risk of a vague problem statement and design argument aligned with sprint stories and tasks related to revising these arguments. Further, the mentor noted that a plan that includes user testing after revising their design makes sense.

However, their mentor also noted the following misalignments: "(1) While [team name] hoped to investigate on the practical side what their new ideas for connection goals are, their stated focus doesn't explicitly describe a user test. (2) While they mention their focus to be "how we might be able to leverage context with [prompts]", it's not clear what risk this is addressing. (3) There are several sprint stories + tasks which do not seem immediately aligned." Their mentor's assessment highlights a critical misalignment: even if the student stories/tasks lay out building and testing a revised prototype that reflects new argumentation, the focus of their sprint does not capture the specific issues in their argumentation -- that they are not reflecting on whether the prompts in their design might build connection between people, and why it's critical that their conceptual approach integrates surrounding context.

Without this understanding, students may go through a design sprint, moving from revised argumentation to revised prototype to testing, but they will not focus their improvement on addressing these conceptual gaps in their work. Further, as their mentor points out, while some sprint tasks may be aligned with this narrative (e.g. "Research into why asking certain reflection questions at certain locations/contexts is more meaningful"), there is a long list of additional

tasks, many of which are not aligned with the core risk identified above. In such a situation, our needfinding suggested that it's likely that the student may get distracted by other tasks, derailing them from the intended mission of their sprint.

Risks	Planned Deliverables	Next Steps
We haven't tested our hypothesis of parallel prompting using context to see if it is more effective at making people connectec to their community. This is risky because we need to know if this path of group connection is worth continuing to pursue	Built prototype... which we will used in Thursday's test. We will have tested everyone adn finally gotten some user test results when it comes to our project, thereby adding flame or water to our hypothesis.	SIG Day Debugging, figuring out deployment Discuss ways to deploy, if possible test and make sure it works on our phones. Onboard everyone onto the app, teach them how to do it for the test. Test Test Aggregate, review data

Figure 51. Team B Example Plan Prior to SIG Meeting (With Compass)

Next, we see a snapshot of Team B's plan the two weeks later (Sprint 5 - Week 9), prior to receiving feedback from their mentor in SIG (see Figure 51). This week, they are back to using Compass scaffolding. Here, their mentor noted several structural alignments: "(1) Their project risk is to test if parallel prompt using context will be effective for connecting, and so they will need to prepare by creating a prototype of their design idea and test that with users. (2) The next

steps are aligned with the goal of building a tech prototype that can be deployed for a test. (3) And the end steps are aligned with the goal of deploying and collecting feedback."

In terms of structural misalignment, their mentor noted the following: "Small – but they failed to mention the one underlying risk was not working on their learning goals of testing technology prototypes. They could have made a prototype that was not tech deployable to get at this risk." In contrast to the week without the Compass, the mentor considered the students' plan to be highly aligned before receiving any plan feedback, noting the ways in which the project risk, deliverables, and next steps focus on testing their specific hypothesis around a parallel prompts design that leverages shared context to socially connect individuals. The one misalignment the mentor found was minor, as they noted, and related to the personal learning goals of the students, which the Compass is not designed to incorporate. When comparing the plan alignment of Team B with and without Compass, this vignette suggests that when students lack alignment between their risks, deliverables, and next steps, they may not have a clear direction or focus for the way they will work that week, leading to a higher likelihood of getting off track in their work. As a consequence, students likely won't deliver what they initially set out to deliver, resulting in not addressing the risk they originally planned to mitigate. By having process scaffolds like Compass in place, students revise their plans within a structure that guides them to maintain the alignment of that plan, prompting them to find ways forward that still meet their deliverables and address their critical risks.

4.6.3 Students plans were better aligned with mentor feedback when using Compass

When students used Compass to manage their planning execution, mentors generally considered their revised plans to be more aligned with the mentor feedback captured by students, across all four teams. Figure 52 compares alignment data with and without Compass for each team, color-coded to reflect the alignment scale (red = not aligned at all, blue = highly aligned).

	Team A	
	compass weeks (avg)	non-compass week
revised plan to feedback (after SIG)	3.50	3.00

	Team B	
	compass weeks (avg)	non-compass week
revised plan to feedback (after SIG)	4.67	0.00

	Team C	
	compass weeks (avg)	non-compass week
revised plan to feedback (after SIG)	3.67	1.00

	Team D	
	compass weeks (avg)	non-compass week
revised plan to feedback (after SIG)	2.33	2.00

Figure 52. Alignment of Plans to Mentor Feedback with and without Compass, Per Team

For teams A, B, and C (which were a mix of both experienced and novice students), mentors generally considered the revised student plans to be “aligned" or “highly aligned” with their feedback the weeks they used Compass, and “somewhat aligned” or “not aligned at all” without Compass. For Team B, students did not capture any mentor feedback during the meeting the week without Compass, which we verified in their retrospective interviews. These findings demonstrate that when student teams used the Compass dashboard and cues, they were able to not only capture, but implement the feedback they received from their mentors into a revised

plan. Further, the ways in which students implemented the feedback and revised their plans aligned with the planning strategies the mentors had coached to them during their SIG meetings.

Mentors also qualitatively assessed the ways in which student plan execution was aligned with the feedback they gave students during the week. Specifically, to assess whether students could also deliver what they planned to deliver when using Compass, mentors documented the ways in which a team's actual deliverables aligned with what the student and mentor discussed as the planned deliverables for each week. Below, we present a vignette from Team C, that compares what the team planned to deliver to what they actually delivered in a week with the Compass, and a week without the process scaffold.

4.6.3.1 Plan Alignment to Mentor Feedback for Team C with and without Compass

Team C is working on a system that helps introductory CS students reflect on and improve ineffective mindsets and ways of working as they make progress on an assignment (e.g. procrastination behaviors driven by not believing in oneself). Team C is composed of one novice student who is joining the studio for the first time this term. Like all teams, Team C had 3 weeks of onboarding and practice with Compass to manage their planning process. Team C then had Compass scaffolding removed for one week in the middle of term, Sprint 3 - Week 6. Figure 53 shows a snapshot of the team's plan and completed deliverables at the end of the week. This plan was extracted from their planning tools (i.e. Planning View, Sprint Log) and presented to mentors for assessment:

Risks	Sprint Focus	Sprint Stories + Tasks	Learnings
Current problem statement and design argument don't get students to change their process–they can recognize their obstacles, but PATH doesn't simply recognizing them doesn't get them tothink of solutions to overcome them	Revise design & focus new interview guide towards getting students to recognize their ineffective behaviors, and come up with solutions	Consolidate all information into solid problem statement + design argument Based on all interviews/probes from past 3 sprints, create an updated design argument + problem statement Updated PRC canvas, updated design argument and problem statement in design log Re-write problem statement Re-write design argument Update design and interview users -- Based on problem statement and design argument, update the design and interview guide. Start recruitment Read literature on overcoming impostor syndrome **Create updated lo-fi prototype in Google Docs with additional features that allow students to identify blockers and come up with strategies to overcome** Recruit new users (CS211 + UPAL) **Interview new users**	-PATH should help students come up with solutions, not just recognize them -PATH should focus on repetitive ineffective behaviors that occur across CS classes

Figure 53. Team C Example Plan with Completed Deliverables (Without Compass)

Upon reviewing this plan, their mentor noted the following alignments between what the student completed, and the planning feedback the mentor gave throughout the week: “User testing with a design probe is consistent with the methods we discussed.” However, their mentor also noted a

core misalignment: “Design probe is focused more on offering general misc. solutions rather than being specific about what the characteristic being tested is - i.e. not considering how helping students deepen their understanding of their own process/link between underlying causes and their process can help them change their process.” Technically, the team did deliver a user test with a design probe, as planned. However, their mentor’s assessment highlights a critical misalignment: the design probe, and the design of the user test itself, is not focused on testing and building knowledge around the critical project risk at the time -- namely, a focus on promoting reflection on one’s process as a way to improve it. Due to this misalignment, the student left the risk for the week unresolved, and will need to conduct a more precise user test next week, focusing on their core hypothesis.

Risks	Planned Deliverables	Next Steps	Deliverables
Current design doesn't get user to CONNECT and RECOGNIZE how understand how some root cause leads them to certain bad behavior that prevent them from achieving a desired outcome Lack an understanding of how to support students once they recognize their root cause is linked to their obstacles	More targeted design with ways to get students to reflect on the root cause and describe how it impacted their process. What strategies did they do? What strategies do they wish they had done? Outlined strategies to support students after we get them to recognize their root cause	Recruit UPAL students Interviews with users, update design based on prior feedback Interviews with users Update design argument based on user tests	Revised design and interview notes that summarize if they could connect the root cause to undesired behavior Week 7: Interview 1 Notes from meeting with Heather Bacon & Joseph Holtgrieve PATH Design Log

Figure 54. Team C Example Plan with Completed Deliverables (With Compass)

Next, we see a snapshot of Team C's completed plan just a week later, Sprint 4 - Week 7 (see Figure 54). This week, they are back to using Compass scaffolding. Here, their mentor noted high alignment between what the student actually delivered, and what they discussed with the mentor throughout the week: "Basically exactly what we talked about - user testing with a new prototype that tries to get at students' process journeys, as well as learning more about how experts handle these problems." In terms of structural misalignment, their mentor noted the following: "Not much, really." In contrast to the week without the Compass, the mentor considered the students' completed deliverables to be highly aligned with the plan feedback that they gave to the team throughout the week, now planning a user test that not only focuses on testing the hypothesis around process reflection, but also exploring how expert programmers reflect and execute as they work through a programming problem.

This vignette illustrates how even with students who are novice to design research and an ARS ecosystem, the Compass framework enabled them to not only revise their plan to implement planning feedback from their mentor after a SIG meeting, but that they can continuously revise and re-align their plan to the coaching they receive at support venues throughout the week, resulting in completed deliverables that are highly aligned with what the mentor expected.

4.7 Discussion

Our findings demonstrate that the Compass process management framework supported students in managing their planning process effectively. First, when students used Compass to manage

their planning process, they successfully revised their plans throughout the week before and after key feedback venues. By identifying these opportune moments where students are likely to receive planning feedback, we can design subroutines that guide students to enact best planning practices in concert with these existing feedback venues. In doing so, we can train students to orchestrate their planning strategies in ways that leverage the supports already available to them in the ecosystem.

Further, when students adapted their plans in response to feedback throughout the week, our findings demonstrate that students maintained plans with sound structure, where their next steps moved them towards completing the deliverables that would mitigate their critical project risks. Literature describes how expert design-researchers set an initial objective, but then flexibly and continuously adapt their plans in response to changes in problem information (i.e. new risks, focusing on one of many deliverables, etc), so that they can still meet the objective. By providing students with a simple template that reflects the core components of a plan (the risks, the deliverables, and the next steps to complete them), Compass helped students compose a initial plan with alignment, and maintain alignment in their plan as they made revisions throughout the week -- maintaining focus on the original intention of their iteration.

Finally, our findings showed that Compass enabled students to revise their plans in ways that accurately implemented the planning feedback coached by their mentors during the week, through existing ARS supports. This is because Compass focused on *lightweight* interactions -- i.e. a dashboard that enabled students to capture feedback within the structure of the plan itself,

and cues within existing communication tools that model social structures (i.e. SIG channels in Slack) to implement that feedback at opportune moments. These findings suggest that process frameworks like Compass can guide students to successfully revise their plans in ways that reflect the planning strategies surfaced to them through their learning interactions within the planning subsystem.

This research demonstrates a need within learning ecosystem design -- as we extend the supports and corresponding learning interactions that enable students to practice self-direction, it's necessary to also scaffold students to enact their practice as they execute within the ecosystem. By extending such ecosystems with process frameworks like Compass, we can increasingly help students and mentors recognize the ways in which students need to improve their metacognitive practice, and guide them in the actionable next steps they can take to improve their practice, as they practice.

Future work can explore the role of such process frameworks in supporting students to build and maintain a practice. For example, our empirical findings suggest that both senior and novice students showed improvements in their process execution the weeks they used Compass, and saw less effective process the weeks the framework was removed. This suggests that, irrespective of the level of expertise, students may benefit from process scaffolds that help them set up their practice, as they build their practice towards performing like an expert, or even as the work to maintain a practice when their surrounding contexts shift.

Chapter 5. Discussion

We first introduced Agile Research Studios (ARS), a learning ecosystem of socio-technical supports designed to scaffold students to make effective use of existing expertise and resources as they learn to self-direct complex work within this community. ARS integrates component supports -- i.e. effective work processes, social structures, and technologies -- into subsystems designed to help students focus on practicing core self-direction skills, such as planning, helpseeking, and reflecting. Our 2-year case study demonstrated initial evidence that the ARS ecosystem enabled a scalable and effective research studio. The ARS model supported 21 student researchers with less than 12 hours of faculty time each week, and led to core research outcomes such as 18 research projects led by 36 students, 9 published papers and extended abstracts at premiere HCI venues, and 3 ACM Student Research Competition awards. Perhaps most importantly, our pilot showed evidence that the studio could provide effective research training that fostered self-directed learning, where students reported engaging in planning activities and skills, and exercising help-seeking practices across the studio.

Despite a rather successful pilot program, over the years I observed that our ARS ecosystem had critical gaps in how the subsystems scaffold planning, helpseeking, and reflection skills, leading students to struggle in their self-direction practice. In order to maintain a scalable and effective ecosystem, it was necessary to support these additional skill gaps, while still working within the constraints of limited mentoring resources. For example, while students had access to an existing

subsystem of planning supports within the ARS ecosystem (i.e. sprint logs and SIG meetings), the ecosystem lacked supports for two subskills of expert planning -- structuring current knowledge of the problem space and assessing it for risks, on which to focus iteration plans. We introduced Polaris, a representation and risk-assessment tool to scaffold students in the construction and evaluation of the arguments underlying their design approach. Polaris embeds expert structural knowledge and diagnostic practice into templates and prompts that focus a novice's attention on three key structural issues in design argumentation: gaps, lack of depth, and misalignment. Our pilot study demonstrates that Polaris helped novice designers identify structural issues in their design arguments as experts do. Further, the mentors considered student issues to be critical to the design, and actionable for coaching students on next steps. Extending ecosystems with diagnostic tools like Polaris can enable mentors to focus their coaching efforts on conceptual issues in the domain, or discussing strategies and plans for mitigating critical issues in designs. Further, the introduction of new supports presents an opportunity to integrate those supports into the existing subsystem. For example, by coupling planning coaching with scaffolds like Polaris, we can make better use of limited mentor resources for more scalable and effective training environments for novice designers.

However, the introduction of more tools and processes means that the individual subsystems, as well as the overall ecosystem, grow in complexity. While well-intended, a complex learning ecosystem rich with supports can overwhelm a student trying to engage in desired learning interactions, inhibiting a learner trying to navigate the ecosystem as they practice self-direction.

Specifically, I observed that students had critical process breakdowns in their planning, such as extracting the planning feedback they receive via the planning subsystem, and implementing into a revised planning process. Consequently, students would continue to execute old plans, despite receiving new planning strategies as feedback. I introduced Compass, a process management framework that scaffolds students to strategically execute their planning process as they move across the planning subsystem. Compass integrates a dashboard and cues to help students manage their planning process by capturing and implementing the planning feedback surfaced through existing ecosystem supports during the week. Using the Compass framework, students are guided to practice *subroutines* that model expert practice -- namely, to reflect on and revise their plans continuously, at opportune moments. Our 8-week pilot study demonstrates that students using Compass revised their plans throughout the week, and composed and executed plans that were more structurally aligned, and more aligned with the feedback received from their mentor. These findings suggest a need to augment ecosystems such as ARS with process scaffolds, that use subroutines to guide students to strategize and execute effective processes as they practice, learning to leverage the many supports available to them.

In the following sections, I will discuss a conceptual approach that illustrates design principles for scaling effective ecosystems, the importance of situating the learning ecosystem within its community, my vision for future work on learning ecosystems.

5.1 An Approach for Scaling Effective Learning Ecosystems

In order to implement *learning ecosystems* that scalably and effectively train the many skills needed to lead design research work, I found that it is critical to take an approach that leverages existing community resources, and focuses on training self direction. Specifically, the ecosystem needs to *distribute* learning interactions across a community of resources, and then weave these supports and interactions back together into subsystems that enable the practice of core self-direction skills.

By taking a *dispersed control approach*, we can carefully study the composition of ecosystems like *Agile Research Studios*, and make intelligent design decisions about how members can engage in learning interactions across available supports. For example, ARS enables undergraduate students to leverage one another as they learn to debug their prototypes, to connect with their own SIG head or other graduate SIG heads for feedback on their study design and methodology, and to access the insight of the faculty advisor as they learn to shape the narrative of their conceptual approach within existing literature. By exploring the many ways in which members can interact across teams, roles, competencies, resources, tools, processes, etc., we can better understand the underlying ecology of our learning environments, and design them to leverage the rich supports that were present all along.

Further, by introducing *subsystems* that interweave these supports with the corresponding learning interactions between them, we can design ecosystems that enable the practice of specific

self-direction skills, like planning, helpseeking, and reflection. For example, we can design tools like the Sprint Log that guide a student to externalize their plan within a structure that models Agile best practices. Then, we can use these tools as shared views in feedback venues like SIG meetings, where they serve as conversational scaffolds between mentors and students on better planning strategies, like scoping down the tasks a student is trying to achieve in one week. By forming these subsystems, we can represent *routines* of practice that enable students to not only practice core self-direction skills like planning, but show them how to enact that practice by engaging the different ecosystem supports available to them.

Even with an ecosystem in place, there may be gaps in how we use them to scaffold expert practice for our learners. We must also build out our understanding of how experts practice these skills, and extend our subsystems to more fully support students in their own practice. For example, we can decompose the subskills that experts practice within a broader self-direction skill like planning. For example, expert design researchers simulate their design solutions and assess risks to inform their iteration plan. With this knowledge, we can expand our subsystems with new supports that embed these subskills, guiding students to practice in the ways in which experts are already familiar. For instance, when we recognized a gap in how we trained students in the subskills of planning, we extended the planning subsystem with *Polaris* -- a tool to scaffold them in representing and assessing risk in their design approach. By implementing tools that embed the structures that experts already use in practice, mentors can more clearly see the risks in how a student is practicing, and prescribe strategies that may help them improve.

Importantly, extending the ecosystem is not simply about introducing new tools and processes. Rather, it is essential to consider how these new supports will integrate into the existing subsystem, introducing new learning interactions as they connect to the other related supports.

For example, when we recognized a need to provide both tools and feedback venues for design argumentation and risk assessment, we further expanded the planning system with *linked canvases* to support visualization of different design problem structures. We then again expanded the subsystem by introducing the *Mysore* feedback venue, where students could bring in the riskiest areas of their argumentation, get mentor feedback that focused on improving their approach and practice implementing that feedback in the session itself. Importantly, when we layered in new support structures in response to the needs of our students, we strategically considered how they would functionally *integrate* into existing subsystems. We can thus enable students to engage in a deeper practice of expert skills, while still leveraging expert feedback.

Even still, an ecosystem rich with supports and corresponding subsystems designed to enable the practice of self-direction skills like planning may not be enough. I found that even when there is an extensive set of learning supports in place, and a corresponding set of prescribed learning interactions, all designed to work together as an ecosystem, that people can still struggle to build a fluid practice -- knowing when and how to execute an effective process across those supports. For instance, we found that on a weekly basis, students would receive valuable planning feedback from their mentors about enacting different strategies, and would fail to implement that feedback as a revised plan. As a result, many would continue to execute their former plans,

rather than maintaining focus on their deliverables and adapting their plans in order to meet those deliverables. In response, I introduced process management frameworks like Compass that guide students through the practice of planning by helping them design, execute, and monitor *subroutines*, or chains of learning interactions, at opportune moments. For instance, Compass used a combination of a dashboard and cues to guide students to manage and execute their plans as they practice. Compass would cue students to use the Compass dashboard to structure their plans prior to SIG meeting. Students would then capture SIG feedback within the dashboard itself, as they reviewed their plan with their mentor. Students would then be cued again to implement that feedback into a revised plan after their SIG meeting. Prior to key feedback venues (i.e. office hours and studio meetings), students would be cued again to review whether they were on track to meet their deliverables, and revise their plans to leverage the upcoming feedback venues to successfully complete their plan. By introducing process frameworks like Compass, ecosystems like ARS can further guide students to enact these subroutines -- structuring, monitoring, enacting, and revising self-direction strategies as they learn to build a practice within these highly supportive communities.

5.2 Situating Learning Ecosystems in Their Context

It is important to consider the way in which this ecosystem was situated within its community and culture of self-direction, and the influence this context may have on ecosystem success. Below, we discuss some of these potential contributing factors.

5.2.1 Participants Value Self-Direction

Whether they realize it or not, nearly all students and mentors that are part of the ARS studio valued, or grew to value self-direction. For instance, it is important to note that students who enter the studio go through a rigorous interview process, involving an application, pre-screening process, and a formal interview with students and faculty. This process evaluates students on their experience with self-directed practice. For example, we ask questions such as how much independent project work have they led, how did they overcome challenges that were new to them, how did they strategically plan out their work in the face of many deadlines, or how did they collaborate with others to achieve a goal. While we do not simply admit students who score high responses for all these questions, we evaluate their areas for improvement, and whether our studio structures can support that development. As a result, we consider most of the students admitted to be primed to practice as a self-directed learner.

Further, the mentors that orchestrate the studio are heavily invested in developing the value of self-direction in their students. Faculty mentors and graduate students who participate invest not only time, but mental and emotional energy when they commit to the mission of developing students as self-directed learners. For example, while the cognitive and metacognitive skills are parts of what students need to become self-directed, they also need other skills, like emotional skills. While we don't have formal structures for this, the culture of our ecosystem is such that mentors have coaching conversations that guide students through reflections around mindset (e.g. feelings of fear and how they relate to impostor syndrome), and how certain mindsets may

inhibit a student's ability to practice effectively. In doing so, it's possible these conversations also enable students to improve their metacognitive and cognitive practice. A mentor with such intentions uses their capacity to continuously monitor, evaluate, and refocus students in their own practices. While the ecosystem design surfaces some of this important information that allows mentors to do this work at scale, it may be essential that the work itself is done by mentors that truly value and prioritize the development of self-directed learners.

5.2.2 Participants Maintain a Culture of Self-Direction

While different students come and go through the studio and the ecosystem over the years, there are many practices in place that are designed to help carry a culture forward as the composition of enrolled students evolves.

For instance, when new tools or processes are introduced, students are already working with a well-established practice and ecosystem. As an example, when we introduced Mysore, senior graduate student mentors who had been leading their students in SIG meetings would follow and observe the faculty mentor for several terms before they could independently practice as a Mysore facilitator. Thus, skills and practices are slowly built up over time.

The community also has a culture of help and general reciprocity. Perhaps because the students care for one another, perhaps because they see it is a cultural norm when they join, but we see students are invested in helping one another, even as members come and go from the community. For example, students orchestrate an onboarding program that provides students with a guide that

explains some of the ecosystem acronyms and practices, and pairs newcomers to senior students, to provide a mentoring relationship outside of their SIG and project groups.

The community even holds a culture of shared responsibility for maintaining the community. For example, students volunteer to join and run service committees (much like how many faculty groups run) that share the responsibility of running the research group. Students lead side-projects like hosting a community hackathon every term, social outings like eating burritos by the lake, running interview committees to recruit new members, making swag like sweatshirts, stickers, hats, etc. All together, these volunteer efforts build a strong sense of camaraderie and community among students, one that we have seen persists as cohorts of students come and go.

Future work can characterize these cultural bits that may play a central role in holding a community together as they practice, and explore how ecosystems might foster such environments, or how they may need to adapt to different settings, cultures, and contexts.

5.2.3 Community Focus on Design-Research

It is important to acknowledge that the design of ARS was focused on the specific setting of training students to practice design-research. While there is a generalizable attention to self-direction skills, many of the specific tools and processes adapt to a design-research context specifically. For example, in a traditional management system, it may not make sense for a company to implement an ARS helpseeking subsystem, because their overall structure follows a strict hierarchy, and does not implement a distributed approach. In such a setting, help

interactions are not meant to reach across layers of the hierarchy. However, one could imagine using tools like Pair Research to connect individuals within a lateral layer, supporting their ability to get and receive help from one another. As another example, Polaris and the linked sets of canvases lay out the argumentation structures needed to argue for a novel conceptual and technical approach in design-research literature. However, the general idea of argumentation structures, and the risks in the rationale, may potentially translate to other areas of research. Similarly, we can examine supports like Compass, that give students a process framework to help them adapt and revise their plans in response to new information. However, it's possible that such a planning process framework may not transfer in certain settings where a precise, fixed procedure is prescribed, like laboratory sciences. Further work is required to understand how the ARS ecosystem model might transfer to other learning settings and organizational contexts.

5.3 Evaluating Learning Ecosystems Beyond "Expert Practice"

In this thesis, I primarily focus on evaluating ecosystems in terms of the learning and process outcomes of students. As an initial measure, a focus on students allows us to understand whether it is possible to have ecosystems that can train students in self-direction practice, like leading design research work. By demonstrating the possibility, we can motivate a need to continue designing sustainable learning ecosystems like Agile Research Studios, to create more opportunities for students. However, there may be many other factors to consider when

measuring the efficacy of a learning ecosystem like ARS. We discuss some of these possibilities below.

5.3.1 Evaluating Self-Directed Practice in Response to Student Needs

Self-directed practice may not need to look the same for every student. As we continue to develop ecosystem supports, we can be more flexible in what we choose to measure in terms of student success. For example, success may not mean that every student perfectly practices the skills of planning, helpseeking, and reflection like experts. It's arguable whether even experts maintain a "perfect" practice. Rather, we could imagine improving our measures to build profiles for individuals, understanding the areas in which they are struggling in their self-directed practice, and the areas in which they are proficient. And for more senior students, while at one point they may have been proficient in part of their practice, it is possible that new contextual factors disrupt their effectiveness temporarily (e.g. a wave of deadlines, or an unexpected family emergency). In such moments, students may need to focus their self-directed efforts on the areas of practice that they need most, such as re-scoping planned deliverables in the event of a family emergency. By more flexibly measuring what "successful practice" may mean for a student in a particular moment, we can ensure that our ecosystems are more responsive to the realities of student learning, and what a student really needs to practice at that time.

5.3.2 Evaluating the Efficacy of Mentors

There are also many other components of the ecosystem that we can consider when evaluating the overall ecosystem. For instance, we may evaluate the development and practice of mentors in

the community, such as measuring the many specific strategies that an expert mentor may enact while orchestrating feedback venues. For example, an expert mentor running a SIG meeting may not split the hour in half for two teams each week, and may choose to enact more responsive strategies, like asking the team struggling more to present first, and choosing to spend more of the session helping that team. Within the subsystems, we may also explore the mentor interactions that lead towards success, such as ad-hoc coaching on top of process cues, or a mentor coaching on project specifics, like guiding students to narrow in on which of the many project risks are most critical. The success of an overall learning ecosystem, in terms of effectiveness and scalability, may also depend on the degree to which mentors in that ecosystem enact successful orchestration strategies.

5.3.2 Evaluating the Performance of Subsystems

We may also need to evaluate the efficacy of how subsystems come together as a functioning ecosystem. For example, in this thesis, I identified limitations to the planning subsystem design that emerged from a mix of observed empirical breakdowns in student practice, and were then grounded in literature on planning as a practiced skill. Learning from these instances, we may define conceptual measures that help us evaluate when such subsystem breakdowns could arise. For example, what are the gaps in how our subsystems scaffold the sub-skills of helpseeking and reflection? Or, where are students failing to capture metacognitive feedback and implement revised metacognitive strategies as they work, such as a mentor coaching a student to seek help when they tend to struggle on their own more than they need to? By methodologically measuring

the performance of the subsystems, we can more strategically evaluate and improve the efficacy of the overall ecosystem.

5.4 The Future of Learning Ecosystems for Self-Direction

I look forward to extending my research by (a) expanding learning ecosystems to guide different skills; (b) expanding learning ecosystems to support mentor development and practice; and (c) adapting ecosystems to alternative learning settings.

5.5.1 Expanding Learning Ecosystems to Train Different Skills

Alongside planning, I am expanding our supports to better scaffold the subskills of helpseeking and reflection, and implementing corresponding process frameworks to guide students in their practice and execution. Helpseeking is also a composition of many subskills (e.g. expressing a structured help request vs. evaluating and identifying the resources that may fulfill the need). The same is the case for reflection (e.g. retroactively reflecting “on-action” vs reflecting “in-action, as you execute). With a fuller set of self-direction supports in place, we can explore how to guide students to enact their practice across these supports. For instance, mentors frequently coach students to enact particular helpseeking strategies throughout the week. such as routing a student to an expert in the community based on the expertise they need to overcome an obstacle in their work, or guiding students in framing help requests for upcoming feedback and practice venues. Similarly, mentors coach their students to enact particular reflection strategies as they work. For example, how a student’s progress on a project risk (e.g. “Work on defining measurable outcomes for your user testing this week”) might be inhibited by risks in their ways of working

(e.g. “You planned an ambitious user test and likely don’t have the hours to execute it. Rather than overcrank, consider focusing your user test on the core functionality you want to test”) I look forward to further exploring how we might improve helpseeking and reflection subsystems by expanding them with new supports and implementing corresponding process frameworks to better guide self-direction practice.

5.5.2 Expanding Learning Ecosystems to Support Mentor Development and Practice

Thus far, the discussion has focused on student outcomes, and better training students within the ecosystem to practice self-direction skills like planning, helpseeking, and reflection. I look forward to exploring more of the skills required to be an effective mentor in such an ecosystem, and the different support tools, processes, and social structures that can better train mentors in effective and scalable ways. For example, one could explore effective strategies of running a SIG planning meeting, where mentors may need to allocate time, adapt, and re-allocate time continuously as they hear the needs of their students. Similarly, one could explore expert strategies for recognizing when and how to route students to other ecosystem supports when appropriate, vs attempting to resolve all of the students’ needs oneself, which does not scale. Future work can build out the ecosystem to support the development of mentors, as a way to promote increased effectiveness, scalability, and resilience of the ecosystem.

5.5.3 Adapting Learning Ecosystems to Alternative Learning Settings

I am exploring how I might adapt these ecosystem models for alternative academic settings. For instance, by adapting many of these ecosystem approaches, how might these approaches apply to

single-course settings at scale, i.e. an introductory HCI course that trains 120+ undergraduate students in skills of design argumentation, risk assessment, and iterative planning? Along another vein, how might the ecosystem design adapt to settings where Ph.D. students are not readily available as mentors? Computer Science departments that focus on undergraduate and Masters education provide a unique opportunity to play with models where Masters students or senior undergraduate students practice as research advisors for novice undergraduate students. I am curious to further play with how I might adapt my conceptual approaches to the varying educational needs in academia.

Longer term, I plan to explore how metacognitive training environments might serve settings beyond academia, for those who aspire to design innovative solutions to the problems they face, but may not have access to formal research opportunities. For instance, small business owners and start-up ventures aspire to produce innovative products and services. However, many aspiring innovators face real-world constraints on the resources and supports available to them. In such circumstances, the lack of metacognitive skills in planning, helpseeking, and reflection can inhibit a person's ability to successfully tackle the real-world problems in their community. In future work, I plan to design learning ecosystems that focus on supporting communities of aspiring entrepreneurs and innovators to develop the metacognitive skills required to lead innovative work.

Conclusion

My focus is to develop *Agile Research Studios (ARS)*, a socio-technical ecosystem that trains learners in how to execute their metacognitive practices across available community supports to effectively self-direct research. To enable scalability despite limited mentoring resources, ARS implements a dispersed control approach -- an ecosystem composed of *component supports* (i.e. individual studio tools, agile processes, social structures) that distribute learning interactions across a community of practicing researchers. To enable effective research training, ARS fosters self-direction, where students focus on the metacognitive skills required to lead design-research work. To achieve this, ARS implements *subsystems* that interweave component supports via *subroutines* (i.e. sequences of learning interactions) that enable students to practice planning, helpseeking, and reflection skills.

My thesis contributes three iterations of Agile Research Studios, each motivated by emergent challenges in learning ecosystem design. In Chapter 2, I introduced ***Agile Research Studios*** through a pilot study that explores the potential of the ARS *learning ecosystem* to scale effective research training. In Chapter 3, I introduced *Polaris*, a case study of how to extend *subsystems* within the ecosystem with new *supports* and *subroutines* to address critical gaps in how the subsystems scaffold expert practice. In Chapter 4, I introduced *Compass*, a case study of how to augment the ecosystem with *process scaffolds* that train students to manage their process execution as they practice within a subsystem, and within the ecosystem more broadly.

In this thesis, I challenge us as readers, designers, researchers, and educators to ask ourselves how we can design more supportive learning communities that enable more learners to practice the skills they need to solve the problems of tomorrow. Real-world problem solving requires us to not only learn technical domain skills, but to develop a sense of being self-directed, within a community of support. Learners who enter the workforce quickly realize the need to know how to strategically plan out their work, seek help from surrounding resources as they encounter hurdles, and to continuously reflect on and improve their ways of working. While we as educators may be able to train these skills with individuals who are committed to apprenticeship-style practice with us for many years, I challenge us to think about how we can strategically leverage our limited mentoring capacity to be inclusive of more learners, while still maintaining highly effective training. By taking an ecosystems approach to designing these learning environments, I aspire to create more opportunities for students to foster the skills they need to tackle the problems of tomorrow, and perhaps more importantly, the skills they need to grow into more self-directed learners, driven to improve themselves and the communities around them.

References

1. Adams, Robin S., and Cynthia J. Atman. "Cognitive processes in iterative design behavior." *FIE'99 Frontiers in Education. 29th Annual Frontiers in Education Conference. Designing the Future of Science and Engineering Education. Conference Proceedings (IEEE Cat. No. 99CH37011*. Vol. 1. IEEE, 1999.

2. Robin S Adams, Jennifer Turns, and Cynthia J Atman. 2003. Educating effective engineering designers: The role of reflective practice.Design studies24, 3 (2003), 275–294.

3. Susan A Ambrose, Michael W Bridges, Michele DiPietro, Marsha C Lovett, and Marie K Norman. 2010. *How learning works: Seven research-based principles for smart teaching.* John Wiley & Sons.

4. Alan Bain and Mark Weston. 2012. The learning edge: What technology can do to educate all children. Teachers College Press.

5. Brenda Bannan-Ritland. 2003. The role of design in research: The integrative learning design framework. Educational Researcher 32, 1 (2003), 21–24.

6. Zygmunt Bauman. 1993. Postmodern ethics. (1993).

7. Molly Beisler and Ann Medaille. 2016. How Do Students Get Help with Research Assignments? Using Drawings to Understand Students' Help Seeking Behavior. The Journal of Academic Librarianship (2016).

8. Philip Bell and Elizabeth A Davis. 2000. Designing Mildred: Scaffolding students' reflection and argumentation using a cognitive software guide. InFourth international conference of the learning sciences. Erlbaum Mahwah, NJ, 142–149.

9. Philip Bell, Elizabeth A Davis, and Marcia C Linn. 1995. The knowledge integration environment: Theory and design. (1995).

10. Philip Bell and Marcia C Linn. 2000. Scientific arguments as learning artifacts: Designing for learning from the web with KIE.International journal of science education 22, 8 (2000), 797–817.

11. Katerine Bielaczyc and Allan Collins. 1999. Learning communities in classrooms: A reconceptualization of educational practice. Instructional-design theories and models: A new paradigm of instructional theory 2 (1999), 269–292.

12. Laszlo Bock. 2015. Work rules! Insights from Inside Google That Will Transform How You Live and Lead. Twelve.

13. Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). How People Learn: Brain, Mind, Experience, and School. Washington DC: National Academy Press.

14. Bransford, John D., and Barry S. Stein. "The IDEAL problem solver." (1993).

15. Ann L Brown. 1992. Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. The journal of the learning sciences 2, 2 (1992), 141–178.

16. Ann L Brown, Doris Ash, Martha Rutherford, Kathryn Nakagawa, Ann Gordon, and Joseph C Campione. 1993. Distributed expertise in the classroom. Distributed cognitions: Psychological and educational considerations (1993), 188–228.

17. Ann L Brown and Joseph C Campione. 1996.Psychological theory and the design of innovative learning environments: On procedures, principles, and systems. Lawrence Erlbaum Associates, Inc.

18. Spencer E. Carlson, Leesha V. Maliakal, Daniel G. Rees Lewis, Jamie Gorson, Elizabeth M Gerber, and Matthew Wayne Easterday. 2018. Defining and assessing risk analysis: The key to strategic iteration in real-world problem solving.Proceedings of International Conference of the Learning Sciences,ICLS 1, 2018-June (1 1 2018), 352–359.

19. Spencer E Carlson, Daniel G Rees Lewis, Leesha V Maliakal, Elizabeth M Gerber, and Matthew W Easterday. 2020. The design risks framework:Understanding metacognition for iteration.Design Studies70 (2020), 100961.

20. Jody Clarke, Chris Dede, D Jass Ketelhut, Brian Nelson, and C Bowman. 2006. A design-based research strategy to promote scalability for educational innovations. Educational Technology 46, 3 (2006), 27.

21. Paul Cobb, Jere Confrey, Andrea diSessa, Richard Lehrer, and Leona Schauble. 2003. Design experiments in educational research. Educational researcher 32, 1 (2003), 9–13.

22. Alistair Cockburn. 2006. Agile software development: the cooperative game. Pearson Education.

23. Mike Cohn. 2004. User stories applied: For agile software development. Addison-Wesley Professional.

24. Allan Collins and Manu Kapur. 2005. Cognitive apprenticeship. In The Cambridge handbook of the learning sciences, R Keith Sawyer (Ed.). Cambridge University Press, Chapter 6.

25. Allan Collins, Manu Kapur, and R. Keith Sawyer. 2014. Cognitive Apprenticeship (2nd ed.). Cambridge University Press, New York, 109–127.

26. David P Crismond. 2011. Scaffolding strategies for integrating engineering design and scientific inquiry in project-based learning environments. InFostering human development through engineering and technology education. Brill Sense, 233–255.

27. David P Crismond and Robin S Adams. 2012. The informed design teaching and learning matrix.Journal of Engineering Education 101, 4 (2012), 738.

28. Pierre Dillenbourg and Patrick Jermann. 2010. Technology for classroom orchestration. In New science of learning. Springer, 525–552.

29. Erin L Dolan and Deborah Johnson. 2010. The undergraduate–postgraduate–faculty triad: unique functions and tensions associated with undergraduate research experiences at research universities. CBE-Life Sciences Education 9, 4 (2010), 543–553.

30. Kees Dorst and Nigel Cross. 2001. Creativity in the design process: co-evolution of problem–solution.Design studies22, 5 (2001), 425–437.

31. Joanna C Dunlap. 2005. Problem-based learning and self-efficacy: How a capstone course prepares students for a profession. Educational Technology Research and Development 53, 1 (2005), 65–83.

32. Carol Dweck. 2006. Mindset: The new psychology of success. Random House.

33. Carol S Dweck. 2000. Self-theories: Their role in motivation, personality, and development. Psychology Press.

34. Matthew Easterday, Daniel Rees Lewis, and Elizabeth Gerber. 2014. Design-Based Research Process: Problems, Phases, and Applications. In Proc. of International Conference of Learning Sciences, Vol. 14.

35. Matthew W. Easterday, Daniel G. Rees Lewis, and Elizabeth M. Gerber. 2016. The logic of the theoretical and practical products of design research.Australasian Journal of Educational Technology 32, 4 (2016), 125–144.

36. Ertmer, P.A., Newby, T.J. The expert learner: Strategic, self-regulated, and reflective. Instr Sci 24, 1–24 (1996). https://doi.org/10.1007/BF00156001

37. Peggy A Ertmer, Donald A Stepich, Cindy S York, Ann Stickman, Xuemei Wu, Stacey Zurek, and Yuksel Goktas. 2008. How instructional design experts use knowledge and experience to solve ill-structured problems.Performance Improvement Quarterly 21, 1 (2008), 17–42.

38. Peggy A Ertmer and Krista D Glazewski. 2015. Essentials for PBL implementation: Fostering collaboration, transforming roles, and scaffolding learning. Essential Readings in Problem-Based Learning (2015), 89.

39. Thomas A Finholt and Gary M Olson. 1997. From laboratories to collaboratories: A new organizational form for scientific collaboration. Psychological Science 8, 1 (1997), 28–36.

40. Gerhard Fischer, Andreas C Lemke, Raymond McCall, and Anders I Morch. 1991. Making argumentation serve design.Human–Computer Interaction 6, 3-4 (1991), 393–419.

41. Mary Frank Fox. 1992. Research, teaching, and publication productivity: Mutuality versus competition in academia. Sociology of education (1992), 293–305.

42. Xun Ge and Susan M. Land. 2003. Scaffolding students' problem-solving processes in an ill-structured task using question prompts and peer interactions. Educational Technology Research and Development 51, 1 (01 Mar 2003), 21–38.https://doi.org/10.1007/BF02504515

43. Xun Ge and Susan M. Land. 2004. A Conceptual Framework for Scaffolding Ill-Structured Problem-Solving Processes Using Question Prompts andPeer

Interactions.Educational Technology Research and Development 52, 2 (2004), 5–22.http://www.jstor.org/stable/30221193

44. Elizabeth M Gerber, Jeanne Marie Olson, and Rebecca LD Komarek. 2012. Extracurricular design-based learning: Preparing students for careers in innovation. International Journal of Engineering Education 28, 2 (2012), 317.

45. Colin M Gray. 2018. Narrative Qualities of Design Argumentation. InEducational Technology and Narrative. Springer, 51–64.

46. Colin M Gray, Colleen M Seifert, Seda Yilmaz, Shanna R Daly, and Richard Gonzalez. 2016. What is the content of "design thinking"? Design Heuristics as conceptual repertoire.International Journal of Engineering Education 32 (2016).

47. Greeno, James G. "Trends in the theory of knowledge for problem solving." *Problem solving and education: Issues in teaching and research* (1980): 9-23.

48. Per L Halstrøm and Per Galle. 2015. Design as co-evolution of problem, solution, and audience.Artifact: Journal of Design Practice 3, 4 (2015), 3–1.

49. John Hattie. 2009. The black box of tertiary assessment: An impending revolution. Tertiary assessment & higher education student outcomes: Policy, practice & research (2009), 259–275.

50. John Hattie and Helen Timperley. 2007. The power of feedback. Review of educational research 77, 1 (2007), 81–112.

51. Hayes-Roth, Barbara, and Frederick Hayes-Roth. "A cognitive model of planning." Cognitive science 3.4 (1979): 275-310.

52. Michael Hicks and Jeffrey S Foster. 2010. Score: Agile research group management. Commun. ACM 53, 10 (2010), 30–31.

53. Sanna J¨arvel¨a and Allyson F Hadwin. 2013. New frontiers: Regulating learning in CSCL. Educational Psychologist 48, 1 (2013), 25–39.

54. David H Jonassen. 2004.Learning to solve problems: An instructional design guide. Vol. 6. John Wiley & Sons.

55. Jonassen, David H. *Learning to solve problems: A handbook for designing problem-solving learning environments*. Routledge, 2010.

56. Jake Knapp, John Zeratsky, and Brad Kowitz. 2016. Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days. Simon & Schuster.

57. Bryan Lawson and Kees Dorst. 2005. Acquiring design expertise.Computational and Cognitive Models of Creative Design VI. Key Centre of DesignComputing and Cognition, University of Sydney, Sydney(2005), 213–229.

58. David Lopatto. 2007. Undergraduate research experiences support science career decisions and active learning. CBE-Life Sciences Education 6, 4 (2007), 297–306.

59. Carol A Lundberg and Laurie A Schreiner. 2004. Quality and frequency of faculty-student interaction as predictors of learning: An analysis by student race/ethnicity. Journal of College Student Development 45, 5 (2004), 549–565.

60. Mariel Miller and Allyson Hadwin. 2015. Scripting and awareness tools for regulating collaborative learning: Changing the landscape of support in CSCL. Computers in Human Behavior (2015).

61. Robert C Miller, Haoqi Zhang, Eric Gilbert, and Elizabeth Gerber. 2014. Pair research: matching people for collaboration, learning, and productivity. In Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing. ACM, 1043–1048.

62. Sharon Nelson-Le Gall. 1981. Help-seeking: An understudied problem-solving skill in children. Developmental Review 1, 3 (1981), 224–246.

63. Newell, Allen, and Herbert Alexander Simon. *Human problem solving*. Vol. 104. No. 9. Englewood Cliffs, NJ: Prentice-hall, 1972.

64. A. Osterwalder. 2008. Business Model Canvas. (2008).

65. A. Osterwalder and Y. Pigneur. 2010.Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley.https://books.google.com/books?id=UzuTAwAAQBAJ

66. Parnell, John A., J. Douglas Barrett, and Edward M. Morrison. "Leveraging organisational hierarchies and networks: reassessing strategic planning and strategic doing." *International Journal of Management and Decision Making* 21.4 (2022): 426-442.

67. Paul R Pintrich. 2004. A conceptual framework for assessing motivation and self-regulated learning in college students. Educational psychology review 16, 4 (2004), 385–407.

68. Tjeerd Plomp. 2009. Educational design research: An introduction. An introduction to educational design research (2009), 9–35.

69. Polya, George. *How to solve it: A new aspect of mathematical method*. Vol. 85. Princeton university press, 2004.

70. Sadhana Puntambekar and Janet Kolodner. 1998. The Design Diary: Development of a Tool to Support Students Learning Science By Design. In the International Conference of the Learning Sciences ’98. ICLS, 230–236.

71. Daniel G. Rees Lewis, Jamie Gorson, Leesha V. Maliakal, Spencer E. Carlson, Elizabeth M Gerber, Christopher K Riesbeck, and Matthew WayneEasterday. 2018. Planning to iterate: Supporting iterative practices for real-world ill-structured problem-solving.Proceedings of InternationalConference of the Learning Sciences, ICLS1, 2018-June (1 1 2018), 9–16.

72. Daniel Rees Lewis, Emily Harburg, Elizabeth Gerber, and Matthew Easterday. 2015. Building Support Tools to Connect Novice Designers withProfessional Coaches. ACM Press, 43–52.https://doi.org/10.1145/2757226.2757248

73. Daniel G. Rees Lewis, Matthew W. Easterday, Emily Harburg, Elizabeth M. Gerber, and Christopher K. Riesbeck. 2018. Overcoming barriers between volunteer professionals advising project-based learning teams with regulation tools: Overcoming advising barriers with regulation tools.BritishJournal of Educational Technology(2018).https://doi.org/10.1111/bjet.12550

74. Allison M Ryan and Paul R Pintrich. 1997. ” Should I ask for help?” The role of motivation and attitudes in adolescents’ help seeking in math class. Journal of educational psychology 89, 2 (1997), 329.

75. Allison M Ryan, Paul R Pintrich, and Carol Midgley. 2001. Avoiding seeking help in the classroom: Who and why? Educational Psychology Review 13, 2 (2001), 93–114.

76. Donald A. Schön, *Educating the reflective practitioner: Toward a new design for teaching and learning in the professions*. Jossey-Bass, 1987.

77. Simon Buckingham Shum and Nick Hammond. 1994. Argumentation-based design rationale: what use at what cost? International journal of human-computer studies 40, 4 (1994), 603–652.

78. P Wesley Schultz, Paul R Hernandez, Anna Woodcock, Mica Estrada, Randie C Chance, Maria Aguilar, and Richard T Serpe. 2011. Patching the Pipeline Reducing Educational Disparities in the Sciences Through Minority Training Programs. Educational evaluation and policy analysis 33, 1 (2011), 95–114.

79. David Williamson Shaffer and Mitchel Resnick. 1999. ”Thick” Authenticity: New Media and Authentic Learning. Journal of interactive learning research 10, 2 (1999), 195–215.

80. Lee S. Shulman. 1986. Those Who Understand: Knowledge Growth in Teaching. Educational Researcher 15, 2 (1986), 4–14.

81. Natalia Smirnov, Matthew W. Easterday, and Elizabeth M. Gerber. 2017. Infrastructuring Distributed Studio Networks: A Case Study and DesignPrinciples.Journal of the Learning Sciences(2017), 1–52.https://doi.org/10.1080/10508406.2017.1409119

82. Suchman, Lucille Alice. Plans and situated actions: The problem of human-machine communication. Cambridge university press, 1987.

83. Jeff Sutherland and JJ Sutherland. 2014. Scrum: the art of doing twice the work in half the time. Crown Business.

84. Heather Thiry and Sandra L Laursen. 2011. The role of student-advisor interactions in apprenticing undergraduate researchers into a scientific community of practice. Journal of Science Education and Technology 20, 6 (2011), 771–784.

85. Heather Thiry, Timothy J Weston, Sandra L Laursen, and Anne-Barrie Hunter. 2012. The benefits of multi-year research experiences: differences in novice and experienced students’ reported gains from undergraduate research. CBE-Life Sciences Education 11, 3 (2012), 260–272.

86. Bernie Trilling and Charles Fadel. 2009. 21st century skills: Learning for life in our times. John Wiley & Sons.

87. Jan Van den Akker. 1999. Principles and methods of development research. InDesign approaches and tools in education and training. Springer, 1–14.

88. Etienne Wenger and Jean Lave. 1991. Situated Learning: Legitimate Peripheral Participation (Learning in Doing: Social, Cognitive and Computational Perspectives) by. Cambridge University Press, Cambridge, UK.

89. Anna Woodcock, William G Graziano, Sara E Branch, Ida Ngambeki, and Demetra Evangelou. 2012. Engineering students' beliefs about research: Sex differences, personality, and career plans. Journal of Engineering Education 101, 3 (2012), 495–511.

90. Haoqi Zhang, Matthew W Easterday, Elizabeth M Gerber, Daniel Rees Lewis, and Leesha Maliakal. 2017. Agile research studios: Orchestrating communities of practice to advance research training. In 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing.