Planning to Iterate: Supporting Iterative Practices for Real-world Ill-structured Problem-solving

Daniel G. Rees Lewis, Jamie Gorson, Leesha V. Maliakal, Spencer E. Carlson, Elizabeth M. Gerber, Christopher K. Riesbeck, and Matthew W. Easterday
daniel.rees.lewis@u.northwestern.edu, jgorson@u.northwestern.edu, lmaliakal@u.northwestern.edu, S.C.@u.northwestern.edu, egerber@northwestern.edu, c-riesbeck@northwestern.edu, easterday@northwestern.edu
Northwestern University

Abstract: Solving real-world highly ill-structured problems involves iteration: gathering information, building, testing, and revising products, experiments, and theories. However, we do not know how to create learning environments to teach iteration for highly ill-structured problems. *How might we help student teams effectively iterate for highly ill-structured design problems?* In this design-based research study we built on learning sciences research to implement *Planning to Iterate*—a weekly planning session in which teams create problem and planning representations. The study took place in a 6-week extracurricular undergraduate design program with five undergraduate project teams working on highly ill-structured problems. To understand team iterative practices, we analyzed videos of teams' weekly planning sessions, and teams' artifacts. Students significantly increased iterative practices, but infrequently integrated the practices together, suggesting re-design with additional coaching.

Keywords: Iteration, problem-solving, planning tools, design research, project-based learning

Introduction

Iteration—building, testing, and refining products, experiments, and theories—is an essential practice to understand problems and create solutions (Schön 1993). Scientists iterate on experiments to create new knowledge, engineers iterate on products for technical and social outcomes, and policy analysts iterate upon policy briefs to create arguments for how policies can achieve desired public outcomes.

Despite the importance of iteration across disciplines, novices struggle to iterate and so waste valuable time pursuing ineffective solutions. Consider the following undergraduate design team working on a highly ill-structured problem. The team partnered with a chain of non-profit stores that sell recycled building materials to reduce Co2 emissions. In the first week, the team identified a problem: most customers left shortly after entering stores without buying materials. By week 2, the team proposed signage to help customers better navigate the store and spent the next 4 weeks designing the signage and an online platform for distributing the signs. In week 5, the team deployed the signage in a local store and found that the signs didn't change customer behavior because they added to the clutter and went unnoticed. With only one week remaining, the team had no time to shift to a new solution and the problem went unsolved. The team *could* have tested and learned from a lower fidelity version of the signage earlier to uncover this unknown. The team's choice to avoid iterating is common; this team was the first of the four teams in their class to test solutions. So why was iterating so challenging?

Iteration was challenging because the teams were taking on real world highly ill-structured problems—problems with unclear parameters that require defining the problem before it can be solved (Jonassen & Hung, 2015; Simon 1973). The nature of highly ill-structured problems mean (a) goals for iterating are unclear and shift, (b) few tests can be quickly conducted in the classroom, and (c) educators cannot tailor problem-specific scaffolds as they do not know the solution. In this paper, we focus on iteration in design problems: design problems are a broad category of highly ill-structured problems in fields such as education, science, and engineering in which problem-solvers change "existing situations into preferred ones" (Simon 1996, p. 111). Conducting effective iterations for highly ill-structured problems is so complex that some argue it is too challenging for K-16 students (Crismond & Adams, 2012). In this study, we draw on seminal learning sciences (LS) research that support iteration (e.g. Brown & Campione, 1994; Kolodner et al., 2003; Hmelo-Silver 2004) to ask, *how can we help student teams to iterate as they take on highly ill-structured problems?*

Background

The shifting nature of highly ill-structured problems makes it difficult for problem solvers to set iteration goals, conduct tests, and for educators to scaffold. As Bardach describes "you will probably keep changing your problem definition, as well as your menu of alternatives, your set of evaluative criteria, and your sense of what evidence

bears on the problem. With each successive iteration you will become a bit more confident that you are on the right track, that you are focusing on the right question." (Bardach & Patashnik, 2015, p. xv). Unsurprisingly, the more ill-structured the problem, the more experts iterate (Jin & Chusilp, 2006).

Real world, highly ill-structured problems: involve stakeholders outside of the classroom who desire a solution; require solutions unknown to students, teachers, and stakeholders (Simon 1973); and are embedded in real world complex systems. While challenging for students and teachers, highly ill-structured problems provide authentic learning opportunities in the practices and modes of thinking of many disciplines. We present an example project to further define and illustrate highly ill-structured problems, and how they pose difficulties for teaching and learning iterative practices. A student design team partnered with a disability advocacy group, and worked with airline staff, baggage handlers, and wheelchair users to reduce damage to wheelchairs transported in the hold of passenger aircrafts. This problem is ill-structured along several dimensions (Jonassen & Hung, 2015). First, this challenge has high *intransparency*: there are many important unknowns (e.g. airline policy, baggage handler training); the instructor and students do not know these unknowns or the form the solution might take. Second, this challenge has many *legitimate competing alternatives*: there are many possible solutions. At the outset, the solution could feasibly include graphic stickers to guide baggage handlers, lifting devices for baggage handlers, airline policy changes, straps for securing wheelchair in the hold, and training. High intransparency and many legitimate competing alternative solutions makes it challenging to know what to iterate on, so the goals and criteria for the iteration are unclear, and educators cannot create solution-specific scaffolds. Third, this challenge has high *heterogeneity of interpretations*: the multiple stakeholders have different constraints and definitions of success that need to be considered and uncovered (e.g. airlines, advocacy groups, baggage handlers). It is difficult to rapidly iterate as stakeholders are outside of the classroom. Due to the combination of *high intransparency, legitimate competing alternatives,* and *heterogeneity of interpretations*, iterating is simultaneously more important and more difficult when solving highly ill-structured problems.

Expert iterative practices to inform learning environments

Schön's theory of reflective practice (1993) emphasizes experts' reflection while iterating to inform both their conception of the problem and solution. Learning scientists have built on Schön to operationalize core expert iterative practices and establish expert-novice differences when iterating to structure (Simon 1973) highly ill-structured problems. First, expert designers dedicate time to *representing and revising* their understanding of the problem (Atman et al., 2007; Jain & Sobek, 2006). In comparison, students dedicate less time to revising their understanding (Atman et al., 2007). Second, experts *prioritize*, which involves identifying the unknowns most needed to understand the problem, and then iterating to uncover these unknowns first (Adams et al., 2003; Guindon 1990). Representing and prioritizing allows experts to overcome the ambiguity caused by the plethora of options in highly ill-structured problems, by identifying and selecting the important unknowns to uncover that set the goals of the iteration. In comparison students do not uncover unknowns because they iterate without a specific goal (Crismond & Adams, 2012), or respond with inaction in the face of problem ambiguity (Lande & Leifer, 2010). Third, Adams et al. (2003) found experts conduct *coupled iteration*. Coupled iterations involve learning about both the problem and the solution through creating, testing, and revising prototypes. Fourth, experts conduct *cascaded iterations*. Cascaded iterations involve rapidly cycling through a problem solving process, including developing solutions, gathering information about the problem or solution viability, and revising solutions (Atman et al., 2007). In comparison to experts, students often do not view rapid iterations as a useful practice (Crismond & Adams, 2012), and so gather less information, and develop and test fewer solutions (Atman et al., 2007). Coupled and cascaded iterative practices allow experts to understand highly ill-structured problems by quickly uncovering unknowns, discarding non-viable solutions, and developing viable solutions.

Teaching iteration

Despite the importance of learning iteration, many educators plan iterations *for* students through common waterfall (test solutions at the end of the project) and stage-gated (e.g test solutions by a given week) rather than teaching students to plan iterations (Crismond & Adams, 2012). We build on seminal LS approaches for teaching iteration for moderately ill-structured challenges, be that testing medical diagnostic theories (Hmelo-Silver 2004), scientific theories with peers (Brown & Campione, 1994), or model car designs and physics theories (Kolodner et al., 2003). However, these effective approaches are not designed for highly ill-structured problems, as they (a) provide students with clear goals for iterating (e.g. model car performance), (b) can be quickly tested in the classroom, and (c) allow educators to tailor problem-specific scaffolds as educators know the problem solution (e.g. cases relevant to the solution). Areas such as Maker education also engage students in iterative making (Kafai, Fields, & Searle, 2014), by lowering the barrier to engagement. Extending seminal LS approaches to highly ill-structured challenges, where students solve real-world problems for others, is a demanding task;

Jonassen and Hung noted "very ill-structured and complex problems, like design problems, may be too difficult to learn in a PBL [problem-based learning] setting" (2015, p.22).

Planning to Iterate: Design argument for supporting student-led iteration

How might we help student teams to practice cascading and coupled iterations as they take on highly ill-structured problems? Our *Planning to Iterate* design argument in this design-based research study (Easterday, Rees Lewis, & Gerber, 2017) proposes that we can support iteration by (a) facilitating discussions and prompting teams to conduct a planning process of representing the problem, prioritizing unknowns, and making iteration plans; (b) using two templates, the *design canvas* and *iteration plan*; (c) providing and prompting use of guiding questions that drive students to consider common pitfalls, and examples (Table 1). The Planning to Iterate process we created supports expert iterative practices of representing and revising problem understanding (Atman et al., 2007; Jain & Sobek, 2006) and identifying and prioritizing the unknowns (Guindon 1990) that guide the goals of a coupled and cascaded iteration (Figure 1; Table 1). The *design canvas* and *iteration plan templates* help teams consider the important aspects and complexity of the problem (problematizing, Reiser 2004). The supports include the examples and guided questions to help the teams conduct the process and use the templates (Kolodner et al., 2003). Facilitators lead discussions of cases and prompt the process and use of tools, but each team manages their own planning (Brown & Campione, 1994).

Table 1: Features of the *Planning to Iterate* design argument for supporting expert iterative practices

1. Iterative practices	**2. Learning environment features**	**3. Principle**
Dedicate time to updating what they know about the problem, solution, and which aspects of the problem are important unknowns. (Atman et al., 2007; Jain & Sobek, 2006).	A. Weekly 1-2hr facilitated team planning sessions where teams revise the design canvas, identify important unknowns, & plan iteration	Regular prompted routines help students to learn practices (Brown & Campione, 1994; Puntambekar & Kolodner, 2005)
	B. Teams update the design canvas problem representation template (figure 1, figure 2b), each session, including noting unknowns	Surfacing complexity (Problematizing; (Reiser 2004); surface project aspects for collaboration (project board, Kolodner et al., 2003)
	C. Scaffolds representing problem: guiding questions, example design questions & facilitated discussion on A4 sheets	Cases & guiding questions for problem-solving decisions (Hmelo-Silver 2004; Kolodner et al., 2003; Puntambekar & Kolodner, 2005).
Prioritize most important unknowns in the project to define the goals for the iteration (Guindon 1990)	D. Prioritization facilitation: Teams prompted to note which sections of design canvas are the most important unknowns using cases and guiding questions on A4 sheets	Cases of team prioritization and guiding questions (Kolodner et al., 2003; Puntambekar & Kolodner, 2005)
Plan to conduct coupled and cascaded iterations. (Adams et al., 2003; Atman et al., 2007; Guindon 1990)	E. Teams use the iteration plan, a planning representation template, to articulate goals, criteria, and tasks based on prioritization of unknowns (figure 1)	Surfacing complexity (Problematizing, Reiser 2004) surface project element for collaboration (project board, Kolodner 2003)
	F. Example iteration plans guiding questions, & example cases and scaffolds for coupled & cascaded iterations	Cases & guiding questions for problem-solving decisions (Hmelo-Silver 2004; Kolodner et al., 2003; Puntambekar & Kolodner, 2005)

Methods

In this design-based research study (Design Based Research Collective 2003) we implemented Planning to Iterate (figure 1) in a 6-week, full-time (9am-5pm) extracurricular (student-led, no grades/credits) university human-centered design program where 4-5 member student teams designed novel products and services to solve a real-world, highly ill-structured problems. Students included 21 undergraduates (4 first-, 12 second-, 5 third-years; 57% female) ages 18-22 years old at a US university. Students majored or double-majored in engineering (15), sciences (3), social sciences (3), art (3) and journalism (3), and had between 0-2 years of design training.

Before the program, an undergraduate student organizer worked with local partner organizations to scope pro-social problems that the partner organizations needed solving, including: (a) reducing teen depression with a medical research center, (b) increasing mental health first responder support for youth with a medical research center, (c) improving airport accessibility for autistic travellers with a disability advocacy group, (d) reducing air travel related wheelchair breakages with a disability advocacy group, and (e) reducing dementia related stigma with a dementia advocacy group. The undergraduate student organizer also coordinated 2-4 hours of methods workshops (e.g. user interviewers) each week led by design educators.

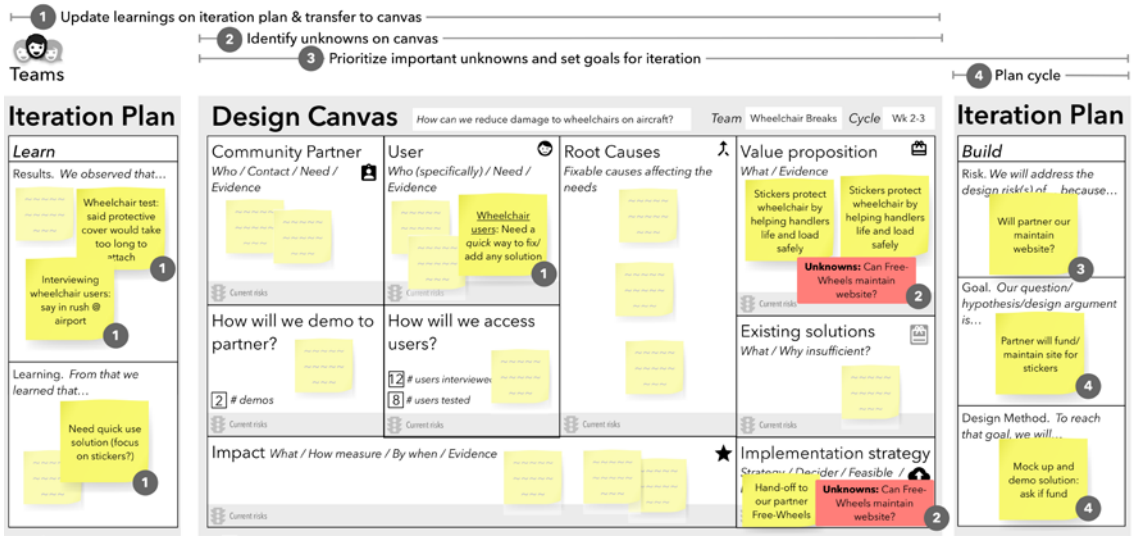


Figure 1. *Planning to Iterate* is designed to support teams to revise their problem representation and identify important unknowns that give them goals for their iteration. Simplified example from week 2-3 cycle.

Two of the authors facilitated 1-2 hour weekly planning sessions in weeks 1-5, in which teams update their findings (forthwith 'learnings') from the previous cycle on the iteration plan and transfer it to the design canvas (20-35 minutes), engage in a facilitated discussion of cases as a whole class and further revise their canvas with guided question sheets (20-30 minutes), identify unknowns ('risks') on the canvas (10-20 minutes), prioritize their most important unknowns/risks (5-10 minutes), and plan their next cycle on the iteration plan (15-25 minutes). Each team had a design canvas and iteration plan fixed to the wall (Figure 2). Teams work on their projects for the next week or 'cycle', the result of which they record at the start of the planning session the next week. Updating the design canvas is both the end of one cycle, and the starting point of the next cycle.

Data collection and analysis

The first analysis examined artifacts and video extracts for differences in coupled and cascaded iterations between the Planning to Iterate implementation and the previous year of the program (an earlier iteration of our DBR project). After finding a significant increase in iterative practices between the Planning to Iterate and previous year, we conducted a second analysis of video data of two planning sessions to examine how the implementation supported iteration. We collected videos of planning sessions of all teams (one camera per-team, 23 sessions, 34 hours), and photos of the iteration plan and design canvas (pre- and post planning session).

Analysis 1: Comparing iterative practices between Planning to Iterate and benchmark

We first examined weekly planning sessions to see if teams conducted coupled and cascaded iterations. In each session, teams summarized learning from the previous week and planned design activities for the current week. We collected data on 4 cycles from 5 teams for a total of 18 sessions (because 2 teams missed 1 session each). For each session we analyzed: (a) team learnings captured on the iteration plan (photos), (b) team learnings during the planning session (5-15 minute videos), and (c) the design canvases before and after the session.

To see if teams conducted coupled iteration—gathering information on problems and solution by building and testing to inform revisions—we developed three sub-codes: (a) *problem learning* to capture team reports of learning about aspects of the problem through testing prototypes, (b) *solution learning* to capture reports of learning about the solution through testing prototypes, and (c) *solution revision* to capture reports of plans to change the solution based on reported learning. We coded a cycle as involving coupled iteration if problem learning (code a) and solution learning (code b), informed solution revisions (code c).

To examine cascading iterations—the expert practice of rapidly cycling through a problem solving process—we coded iteration plans, design canvases, and discussion of learnings. Because students were practicing human-centered design, we used user interviews as a measure of gathering information, prototypes as a measure of the number of solutions created, and user tests (with a prototype) as a measure of testing solutions. For each cycle we counted *prototypes* for each generated or altered prototype. We counted *user interviews* for each unique user teams interviewed about the problem in the given cycle. We counted *user tests* for each test of a prototype with one user; if teams tested three prototypes with two users, we counted six user tests.

To provide a comparison of coupled and cascaded iterative practices, we analyzed data from the previous year's program, which followed a stage-gated approach; teams presented prototypes in weeks 4 and 6. Teams had 80-120 minute weekly coaching sessions, in which teams discussed and recorded their previous learnings and goals for uncovering unknowns. While teams were encouraged by their coach to iterate, the previous program differed from the current study as teams did not receive scaffolding on iteration, and did not engage in a Planning to Iterate process. The program otherwise had the same features, including multidisciplinary teams of 4-5, working full-time on real-world highly ill-structured problems with a client, and the same methods workshops. This study gave us parallel data to a different condition to the current study, namely: the reported learnings on the week's activities, and corresponding audio recording of teams reported learnings. We conducted the same analysis on the previous year's data as described above.

Analysis 2: Examining use of scaffolds

Analysis 1 showed a significant increase in coupled and cascaded iterations (see findings Table 2); as a result, our goal was to conduct a deductive video analysis (Derry et al., 2010) to understand if the design argument was functioning as we hypothesized. Analysis 2 examined video of two planning session from two Planning to Iterate teams: a team working on reducing wheelchair breakages in air travel (113-minutes), and a team working on improving service workers support of dementia sufferers (117-minutes). The planning sessions were taken from week 3 when teams were familiar with the planning sessions. We selected these two sessions because one team (wheelchair breakages) did conduct a cascaded iteration that cycle and one team (dementia stigma) did not. We selected these teams as initial analysis suggested they engaged the least and most in iterative practices. This allowed us to examine ways in which the design succeeded and failed to support student iteration.

We first identified macro-level codes, identifying sections of teams' planning sessions by sustained discussion and activity a given topic (Derry et al., 2010). We identified 30 (wheelchair team) and 34 (dementia team) sections (30-seconds to 11 minutes). Second, to see if the team engaged in iterative practices we identified which macro-level codes included events in which teams engaged in the iterative practices of: (a) revising aspects of problem representation, (b) identifying unknowns, (c) prioritizing unknowns as goals of the iteration, and planning (d) coupled iterations and (e) cascading iterations. Third, we examined whether teams connected these practices as predicted by the design argument: did teams revise problem understanding to inform their prioritization of unknowns, and then goals of coupled and cascaded iteration. We coded each significant event where team members referenced (verbal/gestures) previous discussions and artifacts (e.g. design canvas).

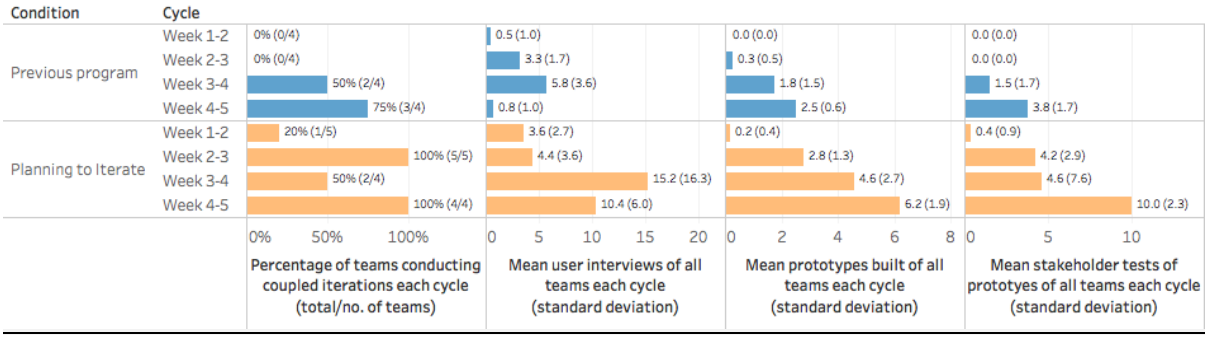
Findings and discussion

Analysis 1: Comparison of practices between Planning to Iterate and benchmark

The teams in the Planning to Iterate implementation showed more iterative practices compared to teams in the previous year. Analysis of student artifacts and video or audio data showed that teams were more likely to demonstrate coupled iteration through discussing and recording learnings about both the problem and the solution through creating and testing prototype solutions, and then suggesting the revision of future solutions (Table 2, column 1). In the Planning to Iterate implementation, teams demonstrated coupled iteration in 66% of cycles (12/18); four of the six cycles that did not involve coupled iteration were in the first week. In comparison, teams in the previous year demonstrated coupled iteration in 31% of cycles (5/16). In the Planning to Iterate implementation all teams conducted coupled iterations in the week 2-3 and 4-5 cycles, whereas in the previous year's program teams did not start conducting coupled iterations until the week 3-4 cycle (3rd cycle).

Consider an example of a coupled iteration from a team working on teen depression and suicide in the Planning to Iterate intervention. In week 2 the team tested three prototypes of an "emotions planner" designed to foster conversations between middle schoolers and parents building on work in child psychologists on "building emotional vocabulary" (design canvas). On the iteration plan the teams reported seeking information from "middle school students", "parents", and a "child psychologist". The teams wrote on the iteration plan "we got feedback that kids would only fill out the planner if someone enforced it", and discussed how this information emerged from showing their prototypes to a student and parent (video data). On the iteration plan the team had a post-it note, which read "look into 'digital' planner and how it could be implemented". Teams discussed creating email prompts, and getting teachers or parents to initiate conversations (video data). Finally, on the design canvas the team added a note reading: "weekly mental health related prompts aim to increase communication b/w students + parents → form habit". This exemplifies coupled iteration: the team discussed a) how problem understanding changed (students don't start conversations about emotions), b) current design (the design needed to prompt students), and c) how to change prototypes (add human/email prompts).

Table 2: Planning to Iterate teams showed more iterative practices compared to the previous year's program



In the current Planning to Iterate implementation teams conducted far more activity suggesting cascading iterations than in the previous year's program (Table 2, columns 2-4). As noted above, cascaded iterations involve rapidly cycling through problem solving activities. Based on analysis of video or audio recordings of team discussions, and team's design canvases and iteration plans, in each cycle teams conducted more user interviews, created more prototypes, and conducted more user tests. In the Planning to Iterate implementation, the mean of every single measure was higher than the previous year's program in every single corresponding cycle (Table 2, columns 2-4). What might explain these differences?

Analysis 2: Video analysis of two planning sessions to examine scaffold use

To understand why iterative practices increased, we analyzed video data of two teams in Planning to Iterate sessions in week 3 (end of week 2-3 cycle/start of week 3-4 cycle): The wheelchair team (WT) working on wheelchair breakages on passenger planes, and the dementia team (DT) working on helping service workers support dementia sufferers. In both planning sessions teams engaged in the process. Teams discussed and recorded learnings from the previous cycle on the design canvas (instances: WT=12, DT=14). Teams revised their design canvases using the guided questions and examples (verbal/gestural references: WT=7, DT=6). Teams also noted unknowns using guided questions (WT=6, DT=3). Teams set and recorded goals based on the prioritized unknowns in the design canvas (WT=2, DT=1). The teams verbally referenced the examples when planning on the iteration plan, and they planned to conduct coupled and cascaded iterations for the next cycle.

Figure 2. 2a. (left) Matt writes on the post-it notes as Aaron holds sheet with guiding questions. 2b. (right) Wheelchair breakages team design canvas and iteration plan (end of the week 3 planning session).

Analysis showed both effectiveness and fragility of Planning to Iterate in supporting iterative practices. The following extract exemplifies WT noting, and nearly missing, the unknown of baggage handler needs. The facilitator has just led a discussion (Table 1). Matt is standing at the design canvas, holding a pen and post-it notes, Jung, Sari, and Aaron are sat at a table facing the canvas. Aaron is holding the guiding questions sheet:

1. Matt: Problem 3. Haven't said the users needs slash do we know it well? (*guiding question*)
2. Aaron: I would say so. (*Sari: nods*)
3. Jung: We know it. We've had enough interviews, for now.
4. Matt: Have we added the most current… (*…evidence?'=next guiding question on sheet*).
5. Jung: …well we don't know the baggage handlers, needs yet. That's the one we don't know.
6. Matt: Oh, that's true (*adds a blank post-it note to user section of design canvas*) (*...off topic talk…*)

7. Jung: And a so a baggage handler needs….
8. Aaron: A process of work.
9. Matt: (*writing on post-it*) load…chair… and bags…quickly…lift…heavy…chair…load…onto… ramp…load…get…through…hold. (*turns to team*). Yeah?
10. Sari: Cool.

In this extract, Jung notes they do not know baggage handlers' (line 6), after Matt reads aloud a question. In lines 2-4 the all team give verbal or gestural indications they know the user needs. In line 4 Matt starts to read the next question, and Jung interrupts in line 5 to say they don't know baggage handler needs. The team then articulates and adds baggage handler needs to the canvas. Thirty seconds after this extract Jung says: "Wait, can we actually put risk on the baggage handler, because we actually haven't spoken to actual baggage handlers", to which Matt responded "yes we can", and adds an 'important unknown' sign to user needs section. The baggage handler need was referenced four more times over the session in relation to: (1) getting access to baggage handlers (27th minute), (2) wheelchair breakages (38nd minute), (3) solution description (52nd minute), and (4) goals (85th minute). At the end of the planning session Matt pointed at a note on the iteration plan reading *build 5 more prototypes, communicate our ideas with baggage handlers* and said "this is a crucial thing for us to get done". Over the next two cycles WT conducted 18 tests with baggage handlers. WT engaged in target practices, but also nearly missed an important unknown with only Jung noting they did not know baggage handler needs.

Video analysis highlighted weakness in supporting teams connect practices. DT raised two and WT raised one important unknown that were then never reference again. For example, in reference to their solution of a training for service workers, DT noted they didn't know whether organizations (e.g. grocery stores) would support service worker training. A team member, Kacee, noted "we haven't shown these (*pointed at design solutions on design canvas*) to any potential partners...so we don't know if they'll implement our design", to which Beth replied "yeah, that's bad." The team discussed the issue for 4 minutes, recorded the issue on the canvas, including adding the 'important unknown' sign, and did not mention the unknown again in the session.

Discussion

Taken together, the findings suggest Planning to Iterate supports student iterative practices in highly ill-structured problems—practices highly challenging for students enact and for educators to support (Crismond & Adams, 2012). The Planning to Iterate implementation saw more instances of coupled iteration—teams reporting on how their activities informed both problem understanding and solutions. It also saw more cascaded iterative practices—teams conducted more interviews, built more prototypes, and tested more prototypes.

Planning to Iterate might have only changed the distribution but not the aggregate iterative practices relative to a stage-gated approach. That is, we might have seen the same aggregate amount of user research, building and testing, with the Planning to Iterate seeing more balanced activity each week, and the other approaches seeing spikes of activity before deadlines. However, relative to the previous year, in the Planning to Iterate intervention teams conducted more of each activity each week. Our findings suggest Planning to Iterate helped teams surface more unknowns, motivating information gathering. Analysis 2 suggests teams generated multiple goals from unknowns. That both coupled and cascaded iteration increased from the benchmark suggests the design argument can support iteration for highly ill-structured problems.

While Analysis 2 supports aspects of the design argument, fails to help teams make connections between practices. The design argument functioned as hypothesized in that video analysis showed teams followed the Planning to Iterate processes. Furthermore, teams demonstrated iterative practices, such as noting and prioritizing unknowns (Atman et al., 2007; Guindon 1990). That teams articulated and recorded unknowns, and then planned activities to uncover unknowns (Atman et al., 2007), suggests that Planning to Iterate guided teams to set goals for iteration in the face of problems ambiguity (Jonassen & Hung, 2015). For example, WT noted baggage handler needs as an important unknown, and planned to reduce this unknown. However, selecting identified unknowns when planning goals appears to be fragile. For example, DT did not reference that they were unsure if organizations would want their solutions after they raised the unknown. Consequently, DT did not explore this important issue. Similarly, WT might not have noted that they did not understand the baggage handlers needs—an unknown that eventually set the goal for their next two cycles.

These findings also suggest re-designs. Teams can miss important unknowns, so we might add team questioning routines to check peer thinking (Brown & Campione, 1994). Furthermore, problem-based learning (PBL) style coaching (tutoring) could help teams connect the practices (Hmelo-Silver 2004). Such work would answer Jonassen and Hung's (2015) call to expand PBL to highly ill-structured problems.

Conclusion

This exploratory design-based research study tested Planning to Iterate, a design for supporting teams iterative practices in highly ill-structured problem solving. Findings suggest that Planning to Iterate supports iteration: teams conducted more coupled and cascaded iterations, and the scaffolds supported iterative practices. This work suggests design principles for supporting iterating when solving highly ill-structured problems: representing problem state and important unknowns to guide a plan for iterating. Problem and plan representations are scaffolded through templates, example representations and guided questions. However, fragile processes suggest adding peer support (Brown & Campione, 1994) and PBL coaching (tutoring, Hmelo-Silver 2004). While this work took place in design, highly ill-structured problems occur across domains, so this work could apply to many domains, such as policy analysis and science inquiry. Understanding how to teach students to iterate is critical if we want to prepare them to solve real-world problems that impact our daily lives.

References

Adams, R. S., Turns, J., & Atman, C. J. (2003). Educating effective engineering designers: The role of reflective practice. *Design Studies*, *24*(3), 275-294.

Atman, C. J., Adams, R. S., Cardella, M. E., Turns, J., Mosborg, S., & Saleem, J. (2007). Design processes: A comparison of students and expert practitioners. *Journal of Engineering Education*, *96*(4), 359.

Bardach, E., & Patashnik, E. M. (2015). *A practical guide for policy analysis: The eightfold path to more effective problem solving*. CQ press.

Brown, A. L., & Campione, J. C. (1994). *Guided discovery in a community of learners.* The MIT Press.

Crismond, D. P., & Adams, R. S. (2012). The informed design teaching and learning matrix. *Journal of Engineering Education*, *101*(4), 738-797.

Derry, S. J., Pea, R. D., Barron, B., Engle, R. A., Erickson, F., Goldman, R., ... & Sherin, B. L. (2010). Conducting video research in the learning sciences. *The Journal of the Learning Sciences*, *19*(1), 3-53.

Design-Based Research Collective. (2003). Design-based research: An emerging paradigm for educational inquiry. *Educational Researcher*, 5-8.

Easterday, M. W., Rees Lewis, D. G., & Gerber, E. M. (2017). The logic of design research. *Learning: Research and Practice*, 1-30.

Guindon, R. (1990). Designing the design process: Exploiting opportunistic thoughts. *Human-Computer Interaction*, *5*(2), 305-344.

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? *Educational Psychology Review*, *16*(3), 235-266.

Jain, V. K., & Sobek, D. K. (2006). Linking design process to customer satisfaction through virtual design of experiments. *Research in Engineering Design*, *17*(2), 59-71.

Jin, Y., & Chusilp, P. (2006). Study of mental iteration in different design situations. *Design studies*, *27*, 25-55.

Jonassen, D. H., & Hung, W. (2015). All problems are not equal: Implications for problem-based learning. *Essential Readings in Problem-based Learning*, 17-41.

Kafai, Y., Fields, D., & Searle, K. (2014). Electronic textiles as disruptive designs: Supporting and challenging maker activities in schools. *Harvard Educational Review*, *84*(4), 532-556.

Kolodner, J. L., Camp, P. J., Crismond, D., Fasse, B., Gray, J., Holbrook, J., . . . Ryan, M. (2003). Problem-based learning meets case-based reasoning in the middle-school science classroom: Putting learning by design (tm) into practice. *The Journal of the Learning Sciences*, *12*(4), 495-547.

Lande, M., & Leifer, L. (2010). Difficulties student engineers face designing the future. *International Journal of Engineering Education*, *26*(2), 271.

Puntambekar, S., & Kolodner, J. L. (2005). Toward implementing distributed scaffolding: Helping students learn science from design. *Journal of Research in Science Teaching*, *42*(2), 185-217.

Reiser, B. J. (2004). Scaffolding complex learning: The mechanisms of structuring and problematizing student work. *The Journal of the Learning Sciences*, *13*(3), 273-304.

Schön, D. A. (1993). *The reflective practitioner: How professionals think in action*. New York: Basic Books.

Simon, H. A. (1973). The structure of ill structured problems. *Artificial intelligence*, *4*(3-4), 181-201.

Simon, H. A. (1996). *The sciences of the artificial (3rd ed.).* Cambridge, MA: MIT Press.

Acknowledgements

We thank Delta Lab, Haoqi Zhang, Julie Hui, and Bruce Sherin for their feedback. We thank Alex Sher for help with implementation. This work was funded by the National Science Foundation (US) grants IIS-1530833 and IIS-1320693.