Scaffolding Game Design: Towards Tool Support for Planning Open-Ended Projects in an Introductory Game Design Class

Alexander Card, Wengran Wang, Chris Martens, and Thomas Price
{acard, wwang33, crmarten, twprice}@ncsu.edu
Department of Computer Science
North Carolina State University
Raleigh, NC, USA

***Abstract*—One approach to teaching game design to students with a wide variety of disciplinary backgrounds is through team game projects that span multiple weeks, up to an entire term. However, open-ended, creative projects introduce a gamut of challenges to novice programmers. Our goal is to assist game design students with the planning stage of their projects. This paper describes our data collection process through three course interventions and student interviews, and subsequent analysis in which we learned students had difficulty expressing their creative vision and connecting the game mechanics to the intended player experience. We present these results as a step towards the goal of scaffolding the planning process for student game projects, supporting more creative ideas, clearer communication among team members, and a stronger understanding of human-centered design in software development.**

***Index Terms*—game design, game development, design documents, planning support tools, education, open-ended programming projects**

I. Introduction

With the growing popularity of digital games as a form of entertainment, an art form, and teaching tool [1], the need to educate students in game design has grown. Students from varying disciplines and fields of study take game design courses, meaning instructors cannot rely on students prepared by a single disciplinary foundation (such as Design or Computer Science). This increase in the number and diversity of students, especially those not in programming related disciplines, necessitates an improvement in the available support tools for students in computer science-based game design courses.

One method of teaching emerging in modern classrooms is project-based learning [2]. In one implementation of such a model, students engage in a semester-long project that requires synthesizing all key learning objectives for the course, including design, playtesting, and implementation. This course structure necessitates rapidly introducing programming concepts to students without prior experience, but without centering programming as the primary learning objective: students primarily need to learn how to *design* games in a way that allows them to map from player experience goals to low-level mechanics and their implementations.

In an effort to better support students we posit the following research question: What challenges do novice game design students encounter when planning game design projects?

In this paper, we present a study and subsequent analysis of student game design in an 200-level computing course. Our study was designed to find challenges that students face in the design process for creative, open-ended game projects. We designed the study starting with a digitized paper prototype, which we used to prompt students to describe game mechanics in a semi-formal "If-Then" structure and connect these mechanics to the game's dynamics and player experience phenomena. After analyzing the results of this prototype and corpus of student design documents, and conducting interviews about students' experiences with the class project, we identified weaknesses in students' ability to delineate player experience goals and connect them to their choices of game mechanics. We then collected and analyzed a corpus of games implemented in PuzzleScript (one of the main game engines used for class projects), which led us to a curated set of mechanics to use as examples for students to browse.

This paper describes our study design and subsequent analysis process, a participatory design procedure in which we collected assignment data and conducted student and instructor interviews from an introductory game design course at North Carolina State University. From the study we found students struggled with expressing their creative vision and connecting the game mechanics to the intended player experience.

II. Related Works

Project-based learning is defined by Thomas [2] as a model which organizes the student's learning around projects; complex tasks which have a basis in realistic and difficult questions or issues which give students the opportunity to synthesize the learning objectives of the course into a single, extended task. Such a model involves the student in the design and planning phase of creation which, in other models, may be overlooked [3], [4]. Investigating the importance of planning in student projects, tools such as the prototype design by

978-1-6654-4592-4/21/$31.00

Nelson and Mateas [5], assist game designers with planning their game space. Similarly, PlanIT! [6] is another planning tool which helps students create and plan projects in the Snap [7] environment by providing implementation and Snap specific assistance. However, at times students hold incorrect beliefs, as Gorson et al notes in [8], 15.42% of students in the research indicated the belief that planning before programming showed low expertise, yet the students negatively self-assessed when becoming frustrated from difficulties starting a project. Another study on undergraduate students found that students demonstrated a weakness in creating plans to be implemented as a functioning program [9]. Our research continues the spirit of these by focusing on ascertaining where undergraduate students in project-based game design courses struggle with design and planning, and why the struggle exists.

III. Research Context

We conducted our research in the context of the introductory game design course at our university, North Carolina State University, CSC 281, a 3-credit undergraduate course. In a typical semester, this course enrolls 100-125 students with a variety of programming backgrounds (including no programming experience). There are no prerequisites for this course, but most students who take it are sophomores. The course satisfies a university requirement for "Interdisciplinary Perspectives," drawing students from a variety of majors and departments, including but not limited to STEM.

CSC 281 aims to teach students how to analyze and build games, both individually and as a team. We have collaborated with the faculty teaching this course over multiple semesters, spring and fall 2020, to ensure that our research goals align with the course's learning objectives. The course's contains a semester project in which students are organized into teams and asked to design, implement, and playtest a digital game.

Successfully realizing their designs as implementations requires students to understand the "MDA" framework proposed by Hunicke et al [10]. MDA is comprised of Mechanics, Dynamics, and Aesthetics, which the framework weaves together to describe games. From a designer's perspective, mechanics feed into the dynamic system behavior, which sets the aesthetic experience. The player's perspective is reversed, the aesthetic tone is born through the observed dynamics implemented though mechanics. Thusly, a game's mechanics, dynamics, aesthetics are mutually constrained. The course intends for students to be human-centered in their design, to put aesthetics first and understand how choices of low-level implementation details affect high-level experiential qualities. Where the aesthetics of the game are the "fun", the dynamics are how the mechanics facilitate the conveyance of that "fun".

In order to design from a player-centered perspective, students also need to have a more nuanced understanding of player experience than "fun," a typical starting point for novice designers attempting to articulate their design goals. GAME 101 introduces students to *typologies of pleasure* via the game design textbook *Rules of Play* [11]. For example, Marc LeBlanc describes categories of pleasure experienced from gameplay that include sensation, fantasy, narrative, challenge, fellowship, and discovery.

A. The Design Document

One major component of the project is the **design document**, which describes their game's backstory, player goal, mechanics, art and sound motifs, and level design, and how these elements contribute to the intended player experience. Students are asked to draft the design document as a first assignment for the project and encouraged to maintain and revise it throughout implementation. Our research focuses on this assignment, seeking to understand its role in practice for teams of student game designers, where it helps them meet learning objectives, and where it fails.

B. PuzzleScript

PuzzleScript is an open-source web-based puzzle game engine designed to help users make tile-based puzzle games. PuzzleScript uses a turn-based approach alongside rule-based programming to allow for succinct rules to encode games in. Rules in PuzzleScript have a pre-condition and post-condition, and are iterated as often as can be. Player movement triggers the rule iteration process for this stage, with an optional extra stage which allows for rules to be applied at the end of a turn.

In both spring and fall 2020 semesters, for pedagogical reasons, students were heavily encouraged to develop their semester project games using the PuzzleScript game engine, however were not required to do so.

IV. Data Collection

On our quest to address our research question and understand where novice game design students struggled, we started by staging interventions in the course CSC 281 for data collection. We developed a digital paper prototype of a planning assistance tool intended to assist the students in their creative design process in the introductory game design course. These documents provided a structured lexicon of nouns, verbs, and miscellaneous words, allowing student extension. In "Natural Programming" [12], Myers and Pane found that the students used event-based or rule-based structures to describe the mechanics of Pac Man, and we used a similar semi-formal "if-then" syntax to define gameplay mechanics. These were intended to encourage the students to consider what conditions were necessary for the mechanic to trigger, as well as how this would change and affect the game state at a higher level than implemented code. This extensible vocabulary provided generic keywords alongside default terms from PuzzleScript. We had two primary anticipations of these interventions: one being that restrictions breed creativity, and the other being the students more completely addressing the provided mechanics to prevent underspecification.

We conducted three classroom interventions: an in-class activity, an extended game design assignment, and an augmented design document. For the in-class activity, students analyzed Sokoban, a game in which the player pushes boxes onto triggers to complete levels. The students briefly played the

game, then were asked to describe the game using the lexicon and "if-then" structure for mechanics. The second intervention tasked the students with extending Sokoban with a new level and mechanic. The final intervention was a modification of the Design Document assignment for the students final project: a culmination of the other interventions expanded by asking the students to explicitly connect their mechanics and design choices to their intended gameplay experience. The paper prototype provided a large amount of scaffolding to address components of the documentation expected by course staff.

To understand students' own perspectives of their design experience, at the conclusion of the semester we conducted one-on-one interviews with 4 students from the Game 101 class, in which we asked them about their design experiences and creating games in class, as well barriers and struggles which arose. Additionally, at the end of the course, instructors selected some student teams to present their games at a showcase.

Out of the 116 students, 94 agreed to participate in the research. However, as the design documents were part of group projects, we were unable to collect data from all participants in the case of groups which had students who declined to participate in the research. Of the 18 groups composed solely of consenting students, course staff provided us with anonymized Design Document assignments for analysis.

V. Analysis

We analyzed the 18 design documents by identifying the mechanics which were provided, how thoroughly these addressed potential edge cases, mechanical coverage: how completely and thoroughly mechanics described possibilities in the proposed gameplay, the detail of the player experience description, and the cohesion with which the mechanics addressed the player experience. We also examined the documents for references to existing games and how the intended experience was described as taking inspiration from the referenced media.

For mechanical coverage of the proposed gameplay, we considered mechanics in two sets: first level mechanics which are immediate consequences of the player's action, and incidental mechanics which are indirect results of player actions. An example first level mechanic is player movement: a key is pressed, and the player character moves. An example incidental mechanic is crate removal: when three boxes are in a row, the boxes are removed from the game world.

We analyzed the interview data using the 6-phase thematic analysis by Braun et al. [13]. In this process we first "familiarize ourselves with data" by reading, transcribing the data (Phase 1); next "generating initial codes" by having one researcher coded all interview data, noting down codes that describes students' experience with planning and implementation while receiving feedback from the other researchers (Phase 2); we next "search for themes", where the researcher presented the initial codings with quotes to the other researchers to find high-level themes (Phase 3); After that, the researchers reviewed and defined themes over 3 iterations by summarizing the core meanings and rejecting themes which were not expressed sufficiently in the data set (Phase 4 and 5). We present the result in the following section (Phase 6).

VI. Results

We organize our results by emergent themes from our interviews, identified in our thematic analysis. In each section, we discuss the challenges that students reported in interviews, and present the summaries relevant to each theme in table I.

A. Expressing Mechanics

In our interviews, students expressed challenges describing game mechanics in a way that communicated their ideas and set them up for successful implementation. One student noted difficulties identifying the core mechanics of their game, and were "overwhelmed with the amount of things that I can put into it" [P1]. They noted that this large possibility space caused problems during implementation: "it was easy to list the amount of mechanics we wanted to include. It's hard to just implement them all within this time span" [P1].

Our analysis of the design documents corroborates the challenge of expressing mechanics. Students struggled expressing their game's mechanics, which did not occur when describing Sokoban. When linking mechanics to the gameplay experience, students often used vague language in their descriptions.

Mechanics were lacking in the coverage of proposed gameplay. 5 of the 18 documents did not cover how the player could move or interact with the world in any detail, only including one or two mechanics. Another 5 of the remaining 13 documents covered the mechanics which were mentioned in the document, and the remaining 8 implicitly contained mechanics which were mentioned elsewhere in the document but were unspecified. Of the 18 design documents collected, only 9 contained any mechanics which considered edge cases or results of player actions.

B. Understanding Possibilities

Students noted challenges being creative and designing game mechanics when they did not know what was possible or feasible in their game engine of choice. One student expressed disappointment at being unable to "come up with a more unique and original game" [P1]. They suggested that knowing more about the capabilities of the Stencyl game engine beforehand would have helped: "if I had at least known about Stencyl... it probably would have been going much smoother" [P1]. Another team noted that it was difficult to plan without knowing the affordances of the PuzzleScript game engine, and they ultimately had to cut ideas that weren't well suited to it: "a huge issue is the fact that like we were working with PuzzleScript... so we were limited in terms of 'what can we do with this?'... We had quite a few issues in the planning phase and we had to shoot down quite a few ideas" [P2]. The mechanics they did keep did not turn out as they expected: "[When planning], I was never thinking, 'how is this going to appear in PuzzleScript?'... it ended up becoming really tricky because there are some objects we have that are not exactly looking like what they are intended to be [in PuzzleScript]... They aren't bad mechanics

Theme	Supporting Evidence	Design Take-Away
Expressing Mechanics	Students struggled expressing their game's mechanics, which did not occur when describing Sokoban, an existing game.	Students need support narrowing the space of possible game mechanics, and expressing game mechanics with specificity. Without such support, they may plan games that are excessively ambitious or vague to receive high-quality feedback from peers.
Understanding Possibilities	Students had issues planning for mechanics and understanding how the mechanics would be implemented in their chosen platform.	Students want to create original games, but need support understanding the space of *possible mechanics* for a given platform, such as Stencyl or PuzzleScript. In the absence of such support, students' games may end up being homogeneous, with mechanics poorly suited to their medium.
Drawing Inspiration	Students indicated they took inspiration from existing games, however in some cases did not provide connections from the cited games.	Remixing and re-imagining existing games is a straightforward strategy for novice game designers to create their own games. However, without explicit support, novices may turn only to well-known games, which may not be suitable for their target platform. Therefore, students may benefit from additional support browsing examples of game mechanics that are diverse, appropriate and feasible.
Connecting Player Experience	Students did not provide links between mechanics and desired player experience, a core intent of the MDA framework.	Students may not intuitively link game mechanics to a desired player experience, or fail to express such connections if they exist. Students would therefore benefit from scaffolding to make these links more explicit when selecting game mechanics.

TABLE I
SUMMARY OF THE THEMES AND TAKE-AWAYS

per se—I don't regret including them, even if it was kind of like a stretch with what they're currently represented by" [P2].

Our analysis of students' design documents found similarities between mechanics. Of the 18 games, only 1 game was described as having different movement from the rest: 17 included keyboard input and 1 detailed grid-based mouse input. While PuzzleScript and Stencyl were officially supported in the course, 5 documents detailed using Unity. However, these 5 games used similar descriptions for mechanics.

C. Drawing Inspiration

Students also noted using other games as inspiration: "just taking ideas from different games because the first few levels kind of reminded me of jetpack joyride where you just have to survive as long as possible" [P1]. They mentioned this as a strategy for coming up with game mechanics: "I guess we kind of thought about other games that did well in the same area and got inspiration from there" [P4]. Taking a simple idea from an existing game was well-suited to teams without game design experience: "we're not super great at designing the actual game, the software that we used, so we didn't want to do something that was over complicated" [P4]. One student also noted that it would be helpful to have suggestions of games that might serve as inspiration, especially less well-known games: "if we had a more diverse [game] reference section, rather than just the most well known type of thing, we could look at more... mechanics or how the design works differently" [P1].

From the 18 documents, 8 cited other games as inspiration. However, students lacked specificity when explaining how such references shaped their own games: 4 of the 8 provided no details as to how the cited games connected to their project, nor was it obvious which aspects of that game were intended to be reflected in the student project design. The remaining 4 documents had a clearly linked theme to the referenced game.

D. Connection to Player Experience

One theme that was notably absent from our interviews, when we asked students how they went about designing their game mechanics, was any mention of linking game mechanics to a desired player experience. This is a core intent of the MDA framework: mechanics support dynamics which create aesthetics to define the player experience. When investigating students' design documents for justifying mechanics by player experience, we found that students used vague descriptions for their player experience, and did not connect the mechanics to the player experience. Of the 18 documents, 9 included "fun" or "feel challenged" in the intended player experience, without offering more specific descriptors or tying their mechanics back to these player experience goals. As discussed in Section 3, students are intended to avoid terms like "fun" and "challenge" when describing player experiences in favor of more specific emotions and experiential qualities.

VII. Future Work and Conclusion

In this paper we presented the data and analysis of a study in which we found novice game design students need support in understanding the possible space of mechanics, expressing their mechanics, and linking mechanics to the player experience.

Moving forward, we intend to implement a tool in the game design course to replace the current paper documents, allowing for data collection from consenting students. To this end we are working with the instructors to ensure that all the criteria desired for the design documentation are fulfilled in the tool. We also intend to conduct interviews with students who used the tool for the game design documentation to ascertain their involvement and the usability and helpfulness of the features and scaffolding we have added.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1917885. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] A. Repenning, D. C. Webb, K. H. Koh, H. Nickerson, S. B. Miller, C. Brand, I. H. M. Horses, A. Basawapatna, F. Gluck, R. Grover, K. Gutierrez, and N. Repenning, "Scalable game design: A strategy to bring systemic computer science education to schools through game design and simulation creation," *ACM Trans. Comput. Educ.*, vol. 15, Apr. 2015.

[2] J. W. Thomas, "A review of research on project-based learning," 2000.

[3] W. Jin and A. Corbett, "Effectiveness of cognitive apprenticeship learning (cal) and cognitive tutors (ct) for problem solving using fundamental programming concepts," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, SIGCSE '11, (New York, NY, USA), p. 305–310, Association for Computing Machinery, 2011.

[4] W. Jin, A. Corbett, W. Lloyd, L. Baumstark, and C. Rolka, "Evaluation of guided-planning and assisted-coding with task relevant dynamic hinting," in *International Conference on Intelligent Tutoring Systems*, pp. 318–328, Springer, 2014.

[5] M. J. Nelson and M. Mateas, "An interactive game-design assistant," in *Proceedings of the 13th International Conference on Intelligent User Interfaces*, IUI '08, (New York, NY, USA), p. 90–98, Association for Computing Machinery, 2008.

[6] A. Milliken, W. Wang, V. Cateté, S. Martin, N. Gomes, Y. Dong, R. Harred, A. Isvik, T. Barnes, T. Price, and C. Martens, "Planit! a new integrated tool to help novices design for open-ended projects," in *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*, SIGCSE '21, (New York, NY, USA), p. 232–238, Association for Computing Machinery, 2021.

[7] A. D. Johnson, R. E. Handsaker, S. L. Pulit, M. M. Nizzari, C. J. O'Donnell, and P. I. De Bakker, "Snap: a web-based tool for identification and annotation of proxy snps using hapmap," *Bioinformatics*, vol. 24, no. 24, pp. 2938–2939, 2008.

[8] J. Gorson and E. O'Rourke, "Why do cs1 students think they're bad at programming? investigating self-efficacy and self-assessments at three universities," in *Proceedings of the 2020 ACM Conference on International Computing Education Research*, pp. 170–181, 2020.

[9] K. Kwon, "Novice programmer's misconception of programming reflected on problem-solving plans," *International Journal of Computer Science Education in Schools*, vol. 1, p. 14, 10 2017.

[10] R. Hunicke, M. LeBlanc, and R. Zubek, "Mda: A formal approach to game design and game research," in *Proceedings of the AAAI Workshop on Challenges in Game AI*, vol. 4, p. 1722, San Jose, CA, 2004.

[11] K. Salen, K. S. Tekinbaş, and E. Zimmerman, *Rules of play: Game design fundamentals*. MIT press, 2004.

[12] B. Myers, J. Pane, and A. Ko, "Natural programming languages and environments," *Communications of the ACM*, vol. 47, pp. 47–, 09 2004.

[13] V. Clarke and V. Braun, "Thematic analysis," in *Encyclopedia of critical psychology*, pp. 1947–1952, Springer, 2014.