01 Jan 2024

Project Wars: A Serious Game to Teach Decision Making in Project Execution

Cecil Eng Huang Chua
Missouri University of Science and Technology, cecq8z@mst.edu

Veda C. Storey

Linda Wallace

Recommended Citation

Chua, C. E., Storey, V. C., & Wallace, L. (2024). Project Wars: A Serious Game to Teach Decision Making in Project Execution. *Communications of the Association for Information Systems, 55*, pp. 110-153. Association for Information Systems.
The definitive version is available at https://doi.org/10.17705/1CAIS.05505

Communications of the Association for Information Systems

Volume 55 Paper 5

8-1-2024

Project Wars: A Serious Game to Teach Decision Making in Project Execution

Cecil Chua
Missouri University of Science & Technology, aehchua@gmail.com

Veda Storey
Georgia State University

Linda Wallace
Virginia Tech University

Recommended Citation

Chua, C., Storey, V., & Wallace, L. (2024). Project Wars: A Serious Game to Teach Decision Making in Project Execution. Communications of the Association for Information Systems, 55, 110-153. https://doi.org/10.17705/1CAIS.05505

Communications of the Association for Information Systems

Education Article DOI: 10.17705/1CAIS.05505 ISSN: 1529-3181

Project Wars: A Serious Game to Teach Decision Making in Project Execution

Cecil Eng Huang Chua
Business and Information Technology Department
Missouri University of Science & Technology
aehchua@gmail.com
0000-0001-9384-1535

Veda C. Storey
Department of Computer Information Systems
Georgia State University
0000-0002-8735-1553

Linda Wallace
Department of Accounting and Information Systems
Virginia Tech University
0000-0002-6705-3604

Abstract:

Project execution (i.e., the production, delivery, and deployment of an information system project's outputs) is a difficult concept to teach effectively because of the many real-time decision making variables facing a project manager. This paper introduces Project Wars, a web-based serious game designed to teach the project execution concepts of deferral of gratification, organizational mindfulness, and discipline in the face of adversity. Project Wars serves as both a platform to instill these concepts and as a basis for discussions on other critical concepts. The latter includes project team management and control, project leadership, the relationship between productivity, slack, and overtime, and project coordination. The paper also explains how to use Project Wars to teach project execution in a HyFlex classroom environment by developing a lesson plan around Project Wars. The game has been tested and demonstrated to be effective.

Keywords: Project Wars, Project Management, Pedagogy, Serious Game, Software Development, HyFlex Classroom, Decision Making.

This manuscript underwent peer review. It was received 01/12/2024 and was with the authors for three months for one revision. Craig Van Slyke served as Associate Editor.

1 Introduction

Project execution concerns the production, delivery, and deployment of a project's outputs (Zwikael & Smyrk, 2019). It is defined as "directing, managing, performing and accomplishing the project work; providing the deliverables and providing work performance information" (Project Management Institute, 2023, p. 329). The focus of project execution is the management of a project after the initial planning is complete, which is when the project interacts with the reality that exists within an organization. Much knowledge on project execution is tacit and difficult to codify. Furthermore, project execution often requires decision making under uncertainty, resulting in experiential methods being the preferred mode of instruction (Dantas et al., 2004; Hellström et al., 2023).

One way to teach project execution is to employ simulations and serious games rather than using solely lectures (Sharp & Hall, 2000; Srinivasan & Lundqvist, 2007). A serious game must have a challenging goal, be fun to play, be engaging, incorporate some concept of scoring, and impart to the user a skill, knowledge, or attitude that can be applied in the real world (Bergeron, 2006). Unfortunately, it is quite difficult to integrate existing serious games or simulations for project execution into a modern project management course for a multitude of reasons.

First and foremost, most existing project management serious games/simulations were designed under the assumption that the instructor is physically meeting with students in a classroom setting. In a more common HyFlex (Beatty, 2019) or online educational environment, it is difficult, for example, to play a board game. A HyFlex educational environment is one in which an instructor teaches across multiple modalities (Beatty, 2019). Some students are physically present, others attend virtually synchronously, and yet others attend virtually asynchronously.

Second, the vast majority of project management serious games or simulations assume a physically synchronous learning environment in which the instructor and students engage with each other at the same time. Most project management serious games require two or more players and are difficult to play in an asynchronous classroom environment, or when the instructor assigns the game as a homework assignment.

Third, it is generally assumed students have access to a standard educational infrastructure, such as a specific version of the Java Runtime environment. This may not be true in the HyFlex/online environment, where the specific IT infrastructure students utilize is unknown, or can change frequently. The authors, for example, have taught students who sometimes attend classes from offices where their IT department has blocked non-authorized software including the Java Runtime Environment.

Finally, most serious games/simulations have documentation describing how to use them, but little documentation on the expected learning outcomes and the development of the lessons around the game/simulation. The effective use of serious games requires the active participation of the instructor, especially during the debriefing stage when the simulation lessons, which are often tacit and difficult to codify, are elucidated to students (Certo, 1976; Wolfe, 1976). Most existing IT Project Management/Software Engineering serious games do not have accompanying teaching notes that explain the major learning points or instructions on how to integrate the game into a lesson plan.

The objective of this paper is to introduce Project Wars, a browser-based game for teaching project execution principles, and demonstrate how it can be integrated into an existing project management/software engineering class to teach concepts critical to the successful execution of IT projects. The main focus of Project Wars is to teach deferral of gratification, organizational mindfulness, and discipline in the face of adversity, all of which are important elements for successfully navigating the unknown unknowns of projects. The game can be played at https://cecilchua.online/stressgame/.

The paper proceeds as follows. Section 2 defines project execution, and reviews the key decision-based learning outcomes typically associated with the project execution module of an IS Project Management/Software Engineering course, as well as the role of serious games/simulations in the learning of these outcomes. Section 3 presents Project Wars. Section 4 outlines a pedagogical strategy for Project Wars, including teaching notes. Section 5 reports on experiences with Project Wars and Section 6 concludes the paper.

2 Project Execution and Serious Games/Simulations

This section defines and describes project execution and discusses the need for serious games in project management courses.

2.1 Project Execution

A project is a "temporary endeavor undertaken to create a unique product, service, or process" (Project Management Institute, 2021, p. 4). The combination of uniqueness and temporality shapes project management courses and differentiates them from other courses. Central to teaching project execution is the management of unknowns, especially unknown unknowns; that is, things we not only do not know about but also things we do not know we do not know about (Kvalnes, 2016; Loch et al., 2006; Ramasesh & Browning, 2014).

One cannot specifically prepare contingencies for any one particular unknown unknown (Loch et al., 2006). However, one can prepare a project such that it can buffer against shock and configure a project so it is flexible enough to respond when a specific, problematic issue arises. The challenge with teaching such concepts is that there are underlying attitudes that accompany the effective deployment of these concepts that must likewise be inculcated. These include the following:

Deferral of Gratification: Deferral of gratification refers to project managers who are willing to delay the taking of an immediate reward to obtain a greater benefit in the future. Deferral of gratification is not an explicit concept in project management. However, it is a concept implicit in much of project management work. Perhaps one reason for its omission is that the deferral of gratification generally has positive outcomes on a wide variety of measures of success (Baumeister et al., 2002; Converse et al., 2014; Mischel et al., 1989), so its specific impact on project management is not surprising.

Nevertheless, to be able to resist shock, a project must be built on solid foundations. For example, an IT project designed so constants are abstracted and not hard coded is more adaptable when an external shock (e.g., change in tax rate) necessitates changing the constants. Such efforts are often time-consuming and may result in foregoing immediate benefits. Projects require both the construction of the actual deliverable and the scaffolding to achieve that deliverable (Feng & Chen, 2014). Within our context, scaffolding refers to project elements that do not directly contribute to the final deliverable, but are important for achieving that final deliverable (Orlikowski, 2006). Requirements and design documents are examples of scaffolding. Explicit in IT project management thinking is the notion that requirements and design are necessary for project success and one cannot move directly to coding without them (Akkermans & van Helden, 2002; Plant & Willcocks, 2007; Remus & Wiener, 2010; Royce, 1970). However, requirements and design documents do not form part of the actual software deliverable.

Similarly, project accounting centers around the reality that projects are expenses that occur within a specific timeframe, with benefits projected into the future. Many introductory project accounting concepts such as payback period, internal rate of return, net present value, and return on investment focus on quantifying the amount of future gratification that makes a present project worthwhile (Lin & Pervan, 2003).

Information systems projects, generally, are fundamentally concerned with deferred benefits, so the financial management of a project normally focuses on managing expenses. Beyond these basic concepts are the cautionary tales where an excessive focus on expenses encourages project failure (Coombs, 2015; Lin & Pervan, 2003). For example, Mähring (2002) recounts how a strategic project was stymied because the project manager wanted to halt requirements gathering because the greatly expanded scope users demanded was causing the project budget to balloon. It was only when executives explained that the multiple hundred percentage cost overrun was still acceptable because the benefits were expected to be an order of magnitude greater than the costs that the project turned around and was completed successfully. Likewise, the change management literature highlights the importance of "quick wins" (Kotter, 1996; Kotter & Cohen, 2002) as a remedy to the fact most users are unable to defer gratification.

Deferral of gratification is something many students acknowledge. However, during actual project execution, many project managers feel pressure to take shortcuts. For example, during the requirements phase, project management at the failed London Ambulance Service Computer Aided Dispatch System ignored the concerns of ambulance staff and, further, did not put the system through rigorous testing before launch to meet a deadline (Finkelstein, 1993). This highlights the need for educational material that teaches students how and why they should resist such short-term pressures.

Organizational Mindfulness: Organizational Mindfulness comprises five parts (Butler & Gray, 2006; Swanson & Ramiller, 2004; Weick & Sutcliffe, 2006).

(1) *Preoccupation with failure*: The unique nature of projects means many unknowns are encountered. These unknowns generate multiple risks, which, in turn, must be effectively managed. A good project manager acknowledges the risks and plans and prepares for contingencies. Then, when risks materialize, they can be handled efficiently and effectively.

(2) *Reluctance to simplify*: There are often contextual nuances in projects that a project manager needs to understand. As a simple example, a naïve project manager would think of price as a single attribute. However, the concept of price (or value) in accounting, operations, and sales are distinct. An accounting price often considers an asset based on some form of depreciation. In operations, the price is often associated with costs. In contrast, a salesperson often views price as a relative judgment by the customer. In an information system that captures all of these ideas, it is often necessary to model distinctly the concepts of book value, current value, cost price, list price, and actual sales price.

(3) *Sensitivity to operations*: Project managers must obtain a nuanced understanding of how user operations work, and be sensitive to the project's environmental context, taking into account such environmental elements as politics (Pinto, 2000). For example, one of the key failures of the New Zealand NovoPay education sector payment system occurred when the project team did not understand the complex nuances of how the educational staff in New Zealand were paid. In contrast, in its successful successor EdPay, Project staff spent years discussing payment with educational staff and streamlining school payment processes prior to its launch (New Zealand Government, 2013; Novopay Staff, 2017).

(4) *Commitment to resilience*: The majority of IT projects consistently encounter problems and run over budget, over time, or else fail to achieve objectives (Standish Group, 2004, 2009, 2020). Project managers must therefore be willing to face and adapt to adversity. Turnover and turnaway (i.e., leaving project management as a profession) are serious problems in IT project management (Joseph et al., 2007; McKnight et al., 2009; Parker & Skitmore, 2005).

(5) *Deference to expertise*: Many IT projects require highly specialized skillsets project managers do not possess. Project managers thus cannot understand and interpret what workers are doing (Kirsch, 1997). However, most workers inherently want a project to succeed (Nidumolu & Subramani, 2003/2004), making it important for the project manager to leverage the expertise of both the project team and other critical stakeholders (e.g., subject matter experts, and involved executives).

Thinking in an organizationally mindful way is difficult for many project management students. It requires a specific mindset. For example, regarding preoccupation with failure, students often do not understand the difference between recognizing a risk (e.g., a car can have a flat tire) versus planning for and mitigating risk (e.g., one must know how to change a tire or have an active auto club membership). Fundamentally, a significant part of project execution training involves teaching students to: always think about what could go wrong; not rely wholly on shortcuts; recognize the context-dependent nature of managing crises; incorporate concrete action into a risk plan; and obtain correct assistance when necessary.

Discipline in the Face of Adversity: There is a substantial argument that project management is a distinct profession with its own disciplinary practices (Cooke-Davies et al., 2009; Hodgson, 2002; Roe & Elton, 1998). Professions possess their own unique bodies of knowledge and impose their own material ordering on the world (Hodgson, 2002). To be disciplined in a profession means that, rather than employing instinct, one employs that body of knowledge to address problems.

When projects proceed as expected ("go right"), there is often little positive feedback, because this is considered business as usual. Indeed, a project manager can still face negative feedback when a project goes well. For example, as a project matures, managers often face user resistance. Similarly, change requests often arise because users perceive value in a project that is proceeding smoothly. In contrast, there is often substantial (often unproductive) feedback when a project does not proceed as desired ("goes wrong"). Project managers must be able to remain calm and ignore negative sentiment to make the correct decision regarding the next course of action (Carmeli et al., 2021; Fey & Kock, 2022; Pavez et al., 2021). For example, when projects go wrong, they get delayed, leading to escalating costs. There is often pressure at that point to cut costs. It is important to recognize naïve cost cutting may not be the appropriate solution. Instead, it may be necessary to allocate more money to the project, because (for example) the project was initially inappropriately scoped (Mähring, 2002).

Teaching project management is a challenge because students often grasp only a superficial understanding of the body of knowledge. For example, one element of the project management body of knowledge is that good requirements and good design are important (concepts). Furthermore, codifying the requirements and design is important prior to implementation in all but trivially sized projects.[1] However, because of time pressure (deferral of gratification, discipline in the face of adversity), many project managers shortcut requirements and design, resulting in tragic consequences. Experiential learning has been demonstrated as one successful way to inculcate correct attitudes (Wozencroft et al., 2014).

2.2 Serious Games in Project Execution

A serious game is a game where the primary purpose is something other than entertainment- typically education (Abt, 1987; Anderson et al., 2024; Dallaqua et al., 2023). The entertainment value of a serious game exists principally to engage the player so the primary purpose can be achieved. Serious games are one form of experiential learning (Westera, 2019), with a demonstrated ability to change individuals' attitudes (Bergeron, 2006; de Vries & Knol, 2011). They provide a simplified and contrived situation that contains enough illusion of reality to induce real-world like responses (Keys & Wolfe, 1990). They are also a popular way of illustrating complex concepts and improving student motivation (Kincaid & Westerlund, 2009).

The concept of serious games is often confused with the concept of games, educational games, and gamification. A game is a form of structured play. Games are different from other forms of play in that games have rules (Merriam-Webster, n.d.). A serious game is a game where the principal goal (frequently educational) is not entertainment (i.e., the play exists to achieve some other purpose (Abt, 1987; Anderson et al., 2024; Dallaqua et al., 2023)). An educational game is a game that has educational value (Cermak-Sassenrath et al., 2023; Kim et al., 2023; Li et al., 2024; Meskill, 1990). An educational game may or may not be a serious game, depending on whether its goal is principally to educate or provide entertainment. For example, Sid Meier's Civilization is an educational game because it teaches concepts like how certain technologies like granaries and aqueducts shape civilizations. The game has been employed in classrooms (King, 2021). However, Civilization sacrifices real world truth for entertainment. For example, the game treats all civilizations including ancient and modern civilizations as having the same cultural mindset (Carpenter, 2021). The literature generally employs the term serious game to identify games designed for classroom use (e.g., Abt, 1987; Anderson et al., 2024; Dallaqua et al., 2023). Finally, gamification is the addition of game-like elements (e.g., scoring) into things not traditionally considered games (Kingsley & Grabner-Hagen, 2023; Mazarakis & Bräuer, 2023; Ofosu-Ampong, 2020). For example, giving prizes to the child who reads the most books in a week is a form of gamification. Table 1 summarizes the difference between these concepts.

Table 1. Differences Between Games, Serious Games, Educational Games and Gamification

Concept	**Definition**	**Example(s)**
Game	A form of structured play. Games have rules to which players must adhere.	Chess, Soccer, FreeCell Solitaire
Serious game	A type of game where the primary purpose is not play/entertainment. Play is used to achieve the primary purpose. Because play is not the primary purpose, the design of such games will often give up entertainment value when it weakens the primary value of the game.	Oregon Trail. The principal goal of this computer game is to teach about the life of 19th-century pioneers on the Oregon Trail. Hunting is often a fun part of the game, but hunting has limited value in the game because the hunter cannot carry all the meat of large game animals.
Educational game	A game that has educational value. Education need not be the primary	The Sid Meier's Civilization series of games were primarily designed for

[1] We argue that the need for documentation for requirements and design is a universal element of the project management body of knowledge. Even proponents of agile project management do not reject the need for documentation; they simply argue against excessive documentation. Attempts to scale agile to large projects such as the scaled agile framework introduce their own forms of documentation like epic templates and the lean business case. Even homework assignment projects need clear requirements as witness the instructor who deals with argumentative students after giving an unclear homework assignment!

	purpose of the game.	entertainment, but can and have been used in classrooms to teach anthropological concepts. Civilization sacrifices pedagogy for gameplay. For example, all civilizations play equivalently. There is no difference between playing as the Aztecs or the English.
Gamification	The introduction of game elements into non-games.	The awarding of weekly prizes to the child(ren) who read the most books.

Because serious games are so well-aligned with the needs of project execution teaching, a substantial number of serious games on project execution have been developed. Appendix A provides a summary of relevant efforts. See also (Caulfield & Xia, 2011; Hellström et al., 2023; Rumeser & Emsley, 2018) for surveys of serious games in software engineering.

However, an instructor teaching a modern online or HyFlex software engineering/IT project management class will discover there are few to no available serious games to use. First, a substantial number of published serious games have been abandoned. Second, many of the games are physical turn-based games, such as board or card games. These kinds of games cannot be used in an online or HyFlex environment because online students have no access to the board/cards. The app-based games are similarly often not procurable. The only two available app-based games are SimSE (Navarro & Hoek, 2004) and AMEISE (Bollin et al., 2011). The version of SimSE on the public website employs a deprecated version of Java. AMEISE employs a pseudo-command-line interface and requires the instructor to set up their own public server, making it difficult to employ. Group games (e.g., most card/board games) are also difficult to employ in asynchronous environments because of conflicting schedules.

There are significant problems when trying to facilitate app-based games in online/hybrid environments. Student infrastructure is not standard. Some students have poor computing resources; others are denied access to install software (e.g., working from the office with strict control policies). Still, others lack the skillset to install software. For example, Java-based games require the installation of the Java Runtime Environment.

Thus, purely web-based project management games would be ideal for this environment. Officially, there are a number of web-based project management games. However, the majority are used either to illustrate narrow concepts (e.g., how a Gantt chart or network diagram works) or are commercial and very expensive, making them impractical for use in the typical course.

3 Project Wars

We developed Project Wars, a web-based serious game for teaching project execution concepts in HTML/JavaScript/PHP. The objective of the game is for the player to produce (initially) 10 “implementation cards” of which a randomly selected 8 are to contain at most 2 minor bugs. Implementation cards are the game’s proxy for how far the actual project deliverable has progressed. These numbers can be adjusted in a settings file. Minimally, a player could complete the game in approximately 15 minutes. However, the typical player makes mistakes that can increasingly make the game more difficult to complete. It generally takes playtesters 1-2 hours to either complete the game (i.e., deliver the aforementioned implementation cards) or set the game to an essentially unwinnable situation. An example (common) unwinnable situation is for the project environment to become so toxic new hires quit immediately upon arrival. The entire game employs a point-and-click interface- students do not need to physically write code or specifications, etc.

The game is designed to be multiplatform. It is possible for a player to begin a game on their work PC and then continue the game on their tablet on a bus by simply logging in under that same username and password. The master data (i.e., data describing how the rules work) for the game are stored in a combination of a MySQL database and a configuration file. Transactional data (i.e., data created as the player plays the game) are stored in the same MySQL database.

The game is inspired by the card game Problems and Programmers (Baker et al., 2005). Many concepts in the game (cards representing requirements, design, implementation, concepts, delayed hiring) are similar to those in Problems and Programmers but adapted to a modern HyFlex teaching environment. The game introduces many new ideas such as separate skill statistics for specification, design and implementation, and logarithmic skill distribution. For example, in Project Wars, one never knows whether an unrevealed card has a problem. In Problems and Programmers, cards were flipped from the unknown to the known state. In other words, one can trace the lineage from Problems and Programmers to Project Wars, but the two are very different games. The game comprises five screens, which flow as Figure 1.

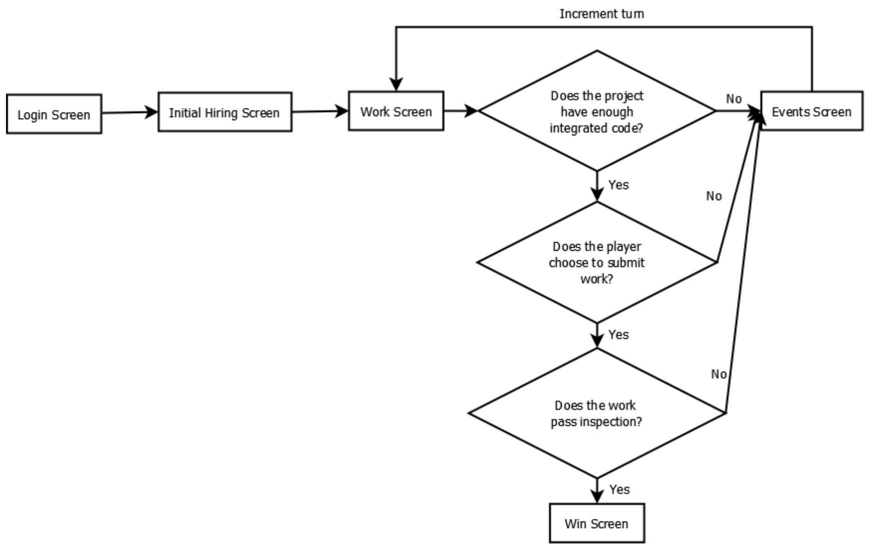
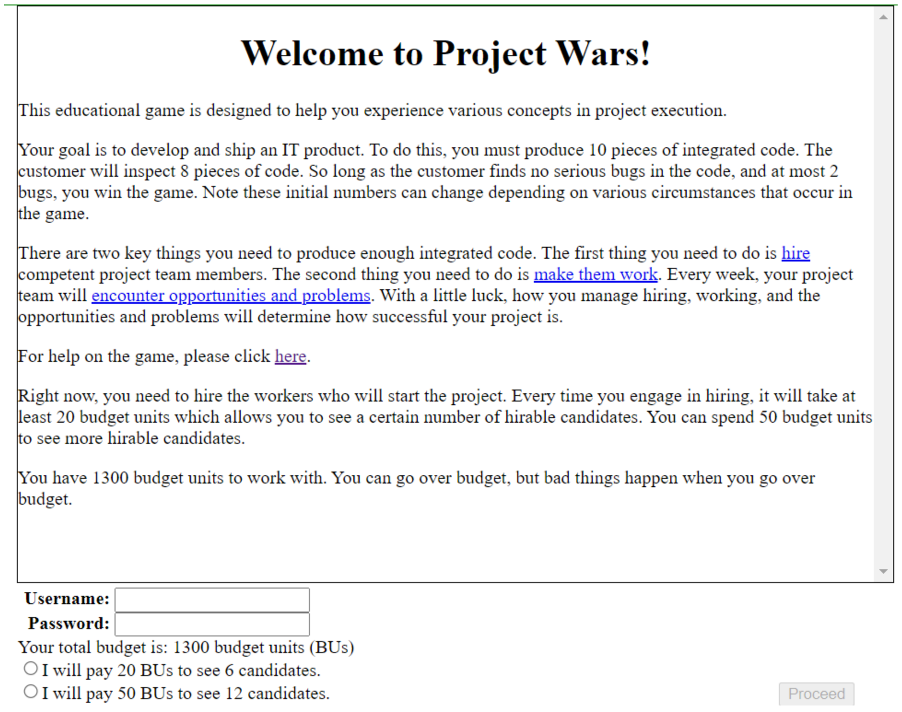

Increment turn
Login Screen
Initial Hiring Screen
Work Screen
Does the project have enough integrated code?
No
Events Screen
Yes
No
Does the player choose to submit work?
Yes
No
Does the work pass inspection?
Yes
Win Screen

Figure 1. Flowchart of the Game

3.1 Login Screen

In the login screen, the player provides as input a username and password and selects whether to engage in an expensive or cheap first hiring round. The game employs the username to track the player's progress and distinguish the player from other players. The password ensures that the player is unique by denying login for a second player with the same username until the first player has completed their run. A screenshot of the login screen is presented in Figure 2.



Welcome to Project Wars!

This educational game is designed to help you experience various concepts in project execution.

Your goal is to develop and ship an IT product. To do this, you must produce 10 pieces of integrated code. The customer will inspect 8 pieces of code. So long as the customer finds no serious bugs in the code, and at most 2 bugs, you win the game. Note these initial numbers can change depending on various circumstances that occur in the game.

There are two key things you need to produce enough integrated code. The first thing you need to do is hire competent project team members. The second thing you need to do is make them work. Every week, your project team will encounter opportunities and problems. With a little luck, how you manage hiring, working, and the opportunities and problems will determine how successful your project is.

For help on the game, please click here.

Right now, you need to hire the workers who will start the project. Every time you engage in hiring, it will take at least 20 budget units which allows you to see a certain number of hirable candidates. You can spend 50 budget units to see more hirable candidates.

You have 1300 budget units to work with. You can go over budget, but bad things happen when you go over budget.

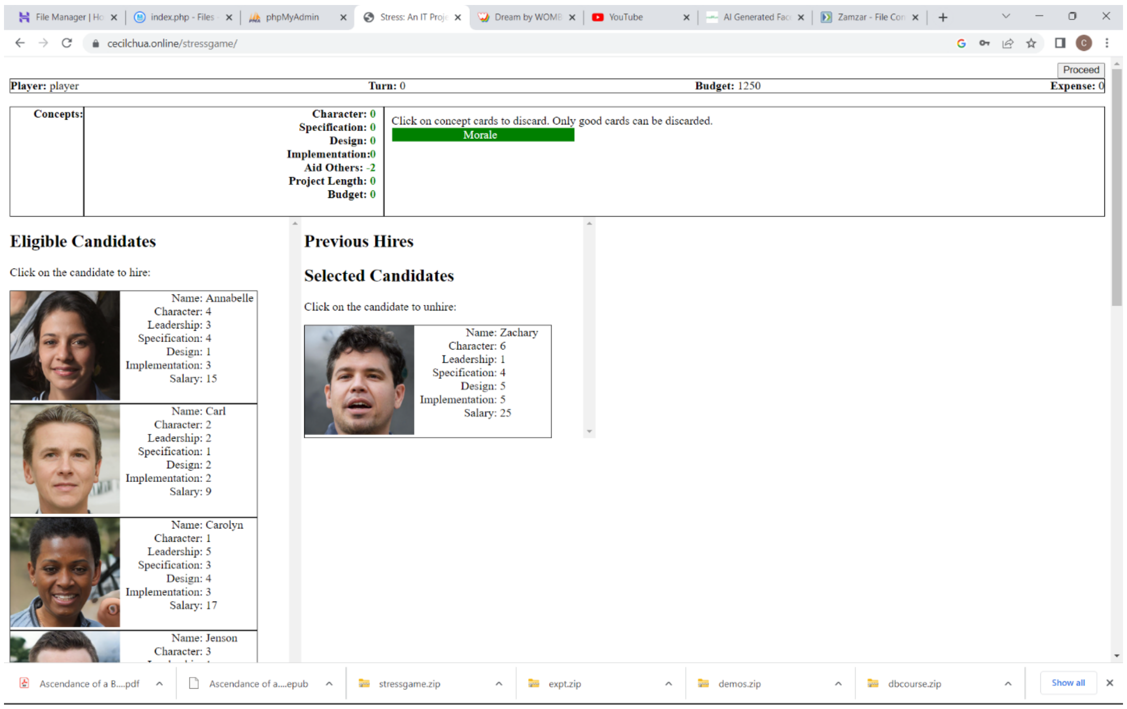
Username:
Password:
Your total budget is: 1300 budget units (BUs)
○ I will pay 20 BUs to see 6 candidates.
○ I will pay 50 BUs to see 12 candidates.

Figure 2. Project Wars Login Screen

3.2 Initial Hiring Screen

After logging in, a new player is brought into the initial hiring screen. Hiring in Project Wars is intended to simulate the generic onboarding of project team members, not just hiring. Except for the initial hiring, all project team members are onboarded two weeks after hiring, simulating the delay associated with onboarding a new person.

Every attempt at hiring is costly, reflecting real life hiring and onboarding (Brenčič, 2009; Howitt & McAfee, 1987). The player can attempt to minimize the hiring cost (which simulates constrained advertising, the hiring committee is uninterested, etc.), which results in fewer candidates, or expend more resources in hiring, which results in more candidates. Figure 3 shows a sample hiring screen.



Player: player **Turn:** 0 **Budget:** 1250 **Expense:** 0

Concepts:

Character: 0
Specification: 0
Design: 0
Implementation:0
Aid Others: -2
Project Length: 0
Budget: 0

Click on concept cards to discard. Only good cards can be discarded.

Eligible Candidates

Click on the candidate to hire:

Name: Annabelle
Character: 4
Leadership: 3
Specification: 4
Design: 1
Implementation: 3
Salary: 15

Name: Carl
Character: 2
Leadership: 2
Specification: 1
Design: 2
Implementation: 2
Salary: 9

Name: Carolyn
Character: 1
Leadership: 5
Specification: 3
Design: 4
Implementation: 3
Salary: 17

Name: Jenson
Character: 3

Previous Hires

Selected Candidates

Click on the candidate to unhire:

Name: Zachary
Character: 6
Leadership: 1
Specification: 4
Design: 5
Implementation: 5
Salary: 25

Figure 3. Initial Hiring Screen

All job candidates have six statistics, five of which are randomly generated:

- **Character:** The candidate's willingness to stick with the project and be a team player. The character statistic is probably the most important in the game because it allows the candidate to avoid certain kinds of negative encounters.
- **Leadership:** The candidate's ability to be a leader. The game allows the player to set one team member as a leader. This person performs no production work, but their leadership score is used to allow the team to avoid certain kinds of negative encounters and to benefit from certain kinds of positive encounters. If a project team has no leader, the project team is vulnerable to all negative encounters that can be avoided by leadership. Furthermore, while leaderless, the project team cannot benefit from positive events that require leadership and may lose the long-term benefits of any previous encounters that required high leadership to obtain. Naïve players may think they need only one candidate with high leadership. However, it is possible for the project team leader to become incapacitated in the game (or quit), necessitating someone else to step into that role immediately.
- **Specification, Design, Implementation:** The skill of the candidate in developing specifications, design, and implementation cards respectively.
- **Salary:** The cost per week of the candidate. This is calculated based on the total of the other statistics with high scores costing more than low scores.

Candidate statistics range from 1-6 distributed in a logarithmic distribution (i.e., a 6 is very rare, whereas a 1-3 is quite common). The numbers are intended as a linear measure of the candidate's strength in that area. Thus, a 6 means the candidate is 6 times more productive in that statistic than a 1 and 2 times more productive than a 3. This reflects how the labor market actually works, because there are usually few exceptional candidates (Ishikawa et al., 2022; Sackman et al., 1968). The faces of the candidates were generated using the AI face generator at BoredHumans.com (https://boredhumans.com/faces.php).

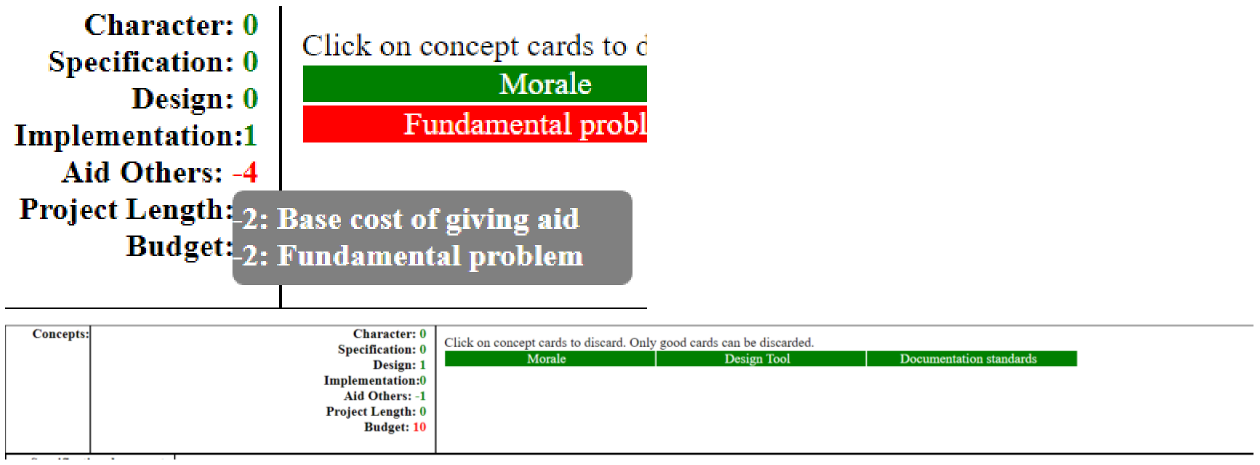
3.3 Work Screen

Once the initial set of candidates are hired and become project team members, the player is brought to the work screen. The work screen is divided into four sections: menu, concepts, specification, and design and work.

Menu Section. There are four gameplay options in the menu and two informational options.

- **Hire (Gameplay):** The player can hire new project team members. Team members appear two weeks after hiring.
- **Work (Gameplay):** Team members perform work (explained below).
- **Submit Deliverable (Gameplay):** This option is only made available to the player once they have 10 integrated implementation cards. On submission, the system checks 8 randomly determined integrated implementation cards for bugs. If the game finds two or fewer bugs and no major bugs, the player wins the game.
- **End Week (Gameplay):** The project team members perform work, the budget is decreased, and the player is brought to the events screen. In the game, it is possible for the budget to become negative. As in real projects, the game allows for budget overruns. Having a negative budget makes the project vulnerable to certain negative encounters such as executive meddling or the project team lead quitting because they understand the project is in a bad state.
- **About:** Credits
- **Reset:** Aborts the game. It is quite easy for novices to put the game into an unwinnable state where the project is over budget, all work is immediately undone by negative encounters, and new hires quit as soon as they join the project. Resetting allows players to try again.

Concepts Section. The concepts section describes the concepts (both positive and negative) currently in play in the project. Figure 4 presents the concepts section.


Character: 0
Specification: 0
Design: 0
Implementation:1
Aid Others: -4
Project Length:
Budget:

Click on concept cards to d
Morale
Fundamental probl

-2: Base cost of giving aid
-2: Fundamental problem

Concepts:

Character: 0
Specification: 0
Design: 1
Implementation:0
Aid Others: -1
Project Length: 0
Budget: 10

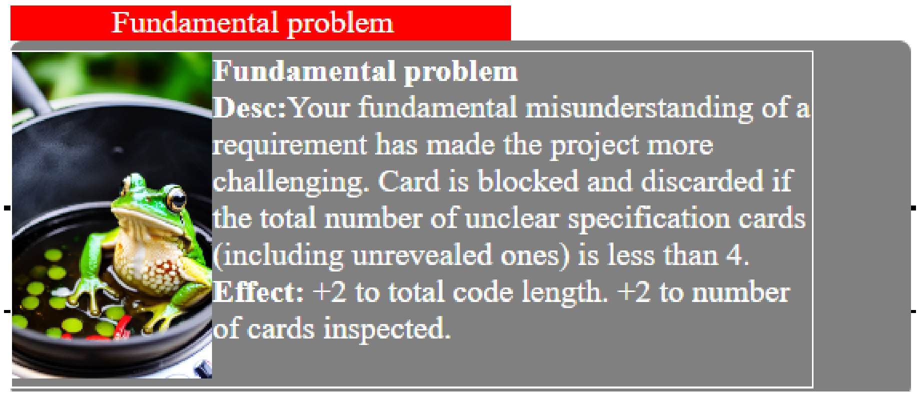
Click on concept cards to discard. Only good cards can be discarded.

Morale | Design Tool | Documentation standards

Figure 4. Concepts Section

Concepts are enduring positive and negative adjustments to the project team's performance obtained as a result of encounters (see Section 3.4) below. Concepts continue to affect the team until certain conditions are met. For example, a positive concept can be lost when the project leader's leadership falls below a threshold (which can occur if the project leader quits). Some positive concepts like "executive favor" can block negative concepts. Likewise, negative concepts can be overcome when the project conditions improve. For example, the concept of “misinformed design” is discarded when the number of unclear specification cards falls below a certain threshold.

On the left of the concepts section is the concept summary, which describes the overall effect of the concepts in play. Hovering over each concept summary number reveals which concepts are affecting the concept summary number. On the right, are the set of concept cards in play. Green concept cards provide benefits to the player while red ones are drawbacks. Hovering over each card provides a description of what the card does. Figure 5 presents an example of hovering over the concept “Fundamental problem.” The card explains what the problem is (your fundamental misunderstanding…) and also how to get rid of it. Once there are fewer than 4 unclear specification cards, the fundamental problem card will be removed from the project.


Fundamental problem

Fundamental problem
Desc: Your fundamental misunderstanding of a requirement has made the project more challenging. Card is blocked and discarded if the total number of unclear specification cards (including unrevealed ones) is less than 4.
Effect: +2 to total code length. +2 to number of cards inspected.

Figure 5. Hovering Over the Concept Fundamental Problem

Specification and Design Section. The specification and design section displays the specification and design documents in play. An example specification and design section is presented in Figure 6.

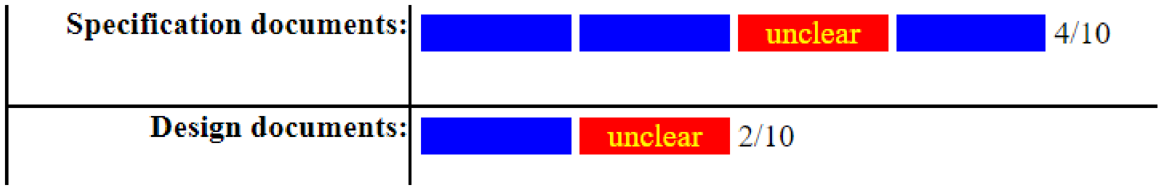


Figure 6. Specification and Design Section

Documents have three states: 1) clear, 2) unclear and revealed, and 3) unclear and unrevealed. Documents are displayed as red if they are unclear and revealed, and blue if they are clear or their state is unrevealed. Unrevealed means the player does not know whether the document is clear or unclear. As in real life, there are known issues, reflected in the red unclear cards, and unknowns, reflected in blue unclear cards.

The player can develop a total of 10 specification documents and 10 design documents. The player can also review these documents. A successful review reveals unclear specifications or design documents. An unsuccessful review does not reveal underlying problems.

Clear specification documents reduce the probability that design documents will be unclear and implementation cards will have bugs in them. Having clear specification documents also allows the player to avoid certain negative encounters. Clear design documents reduce the probability of bugs in the implementation cards and help avoid certain negative encounters.

Work Section.

Here, the player assigns work to each team member who can perform the following tasks:

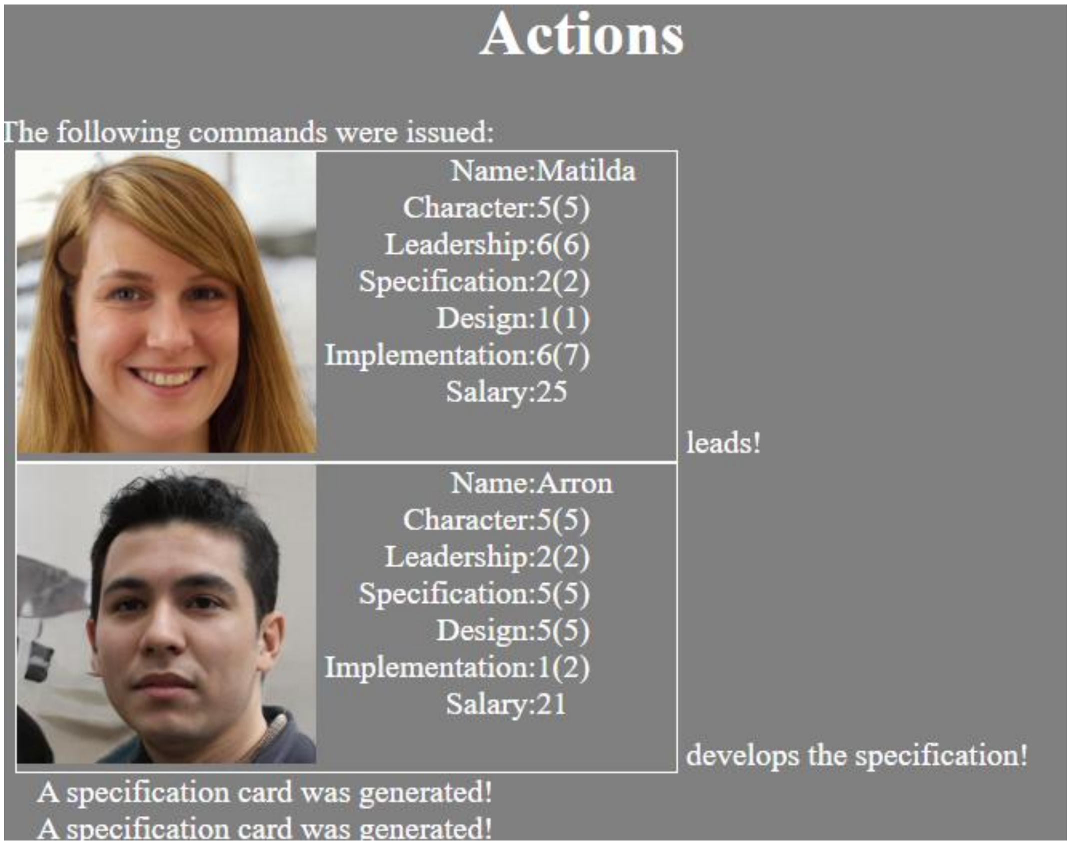
- **Do Nothing:** Performs no actions this turn. This can sometimes be forced on the player, because the project team member is, for example, sick, skiving (i.e., shirking work), fighting with another team member, or consumed with firefighting.
- **Lead:** Only one project team member can lead at a time.

- **Create/Edit Specification**: The team member creates specification documents based on their specification skill augmented by specification-based concepts and/or replaces unclear specification documents. The probability that a new or replaced specification document will be unclear is determined by the team member's skill. If there is a lower percentage of specification documents than design documents and/or implementation cards, the project loses a design document/implementation card for every specification document created. This reflects having to redo design/implementation as requirements are added.
- **Review Specification**: The team member has a chance to reveal unclear specification documents based on their specification skill.
- **Create/Edit Design**: The team member creates design documents based on their design skill augmented by design-based concepts and/or replaces unclear design documents. The probability a new or replaced design document is unclear is determined by the team member's skill and the number of existing clear specification documents. If there is a greater percentage of implementation cards than design documents, the project loses an implementation card for every design document created. This reflects that refining the design forces a rework of the implementation.
- **Review Design:** The team member has a chance to reveal unclear design documents based on their design skill and the number of existing clear specification documents.
- **Create Implementation**: The team member creates unintegrated implementation cards. Implementation cards are linked to a particular team member, reflecting that understanding someone else's implementation is more difficult than understanding your own. Four variables are tracked for each implementation card:
    - *Whether the card, if it has a bug, is revealed.* Cards with no bugs can never be revealed. Revelation means the bug is known to the player.
    - *Whether the card has no bugs, a simple bug, or a complex bug.* Simple/complex bugs denote the difficulty of fixing the bug. A simple bug is fixed by replacing the card with a new one. This takes one skill point. A complex bug is fixed by moving the card to the top of the code stack and then transforming it into a simple bug. It can take multiple skill points to replace a complex bug.
    - *Whether the card has no bugs, a minor bug, or a major bug.* Minor/major bugs denote how unhappy users are with the bug. When the player submits the deliverable, the game randomly checks the code. The game rejects the deliverable if the game discovers at least three minor bugs or at least one major bug.
    - *Whether the card is integrated.* Integration is an action each team member can perform. Integration packages the implementation cards for delivery. Once an implementation card is integrated, no further action can be performed on the card, but the card is still vulnerable to negative encounters.
- **Assist in Implementation:** This form of work is similar to implementation creation except the implementation card is created next to another team member. Also, assisting in implementation carries a penalty, where one produces fewer implementation cards than if one created the implementation. Assisting in implementation is an important strategy in the game. While the player takes a productivity hit, assistance makes the code less vulnerable to certain kinds of negative encounters. Also, assistance is the only way one can manage abandoned code -- when another team member quits.
- **Review Implementation:** Inspect implementation cards for bugs.
- **Debug Implementation:** Fix bugs. It is possible for a debug action to make the situation worse, where (for example) a simple minor bug is replaced with a complex, major bug.
- **Package Implementation:** Set all implementation cards for one team member to integrate.
- **Terminate:** Fire a project team member. In the game, this is generally a bad action to take because it reduces the overall morale of all team members. Morale reduces team member character across the board, making the team more vulnerable to negative encounters.

Team members can be assigned to do work carefully and to perform overtime. Doing work carefully halves the amount of work a team member performs but reduces the likelihood of error (unclear specification/design and buggy code). Overtime increases the amount of work done by 1.5 but permanently reduces the team member's character by 1.

3.4 Events Screen

The events screen summarizes the effect of the work the project team did and then presents five randomly determined encounters, one for every day of the work week. Figure 7 presents an example summary of the work.



Actions

The following commands were issued:

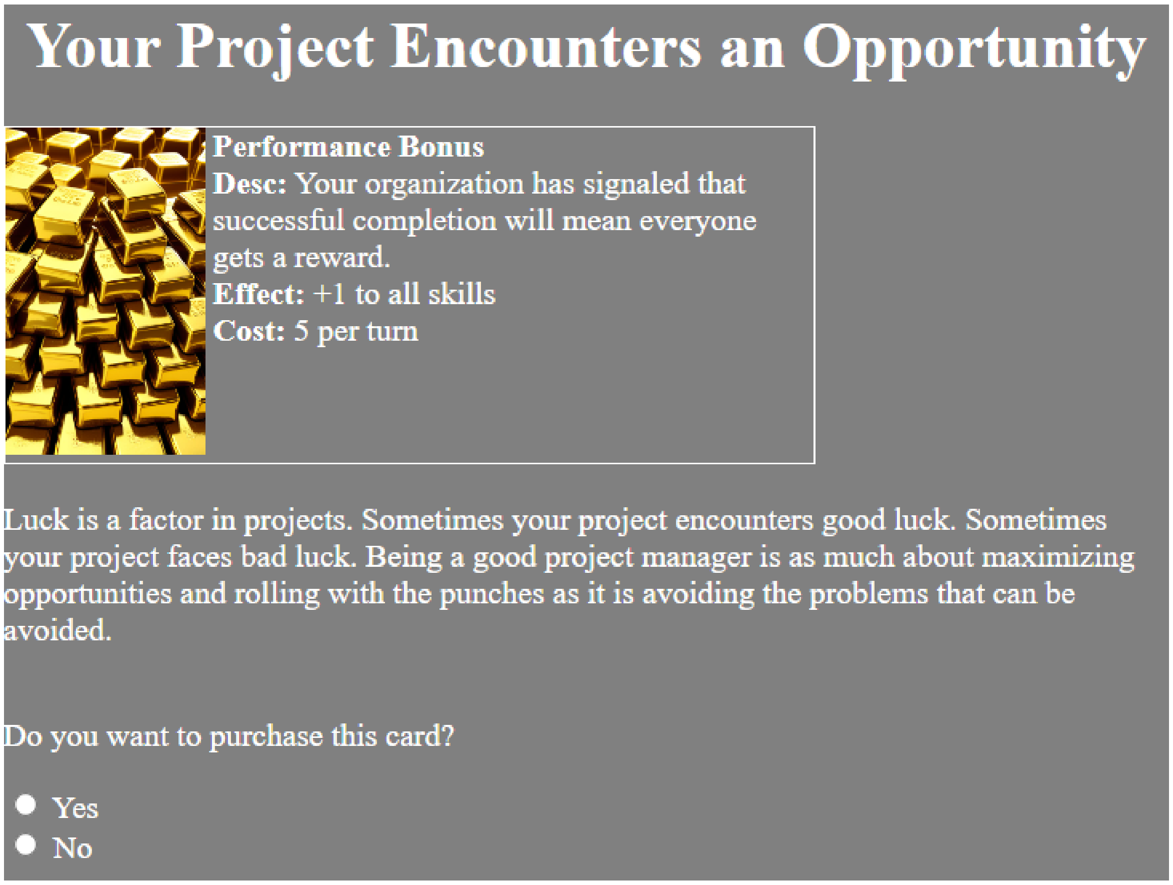
Name:Matilda
Character:5(5)
Leadership:6(6)
Specification:2(2)
Design:1(1)
Implementation:6(7)
Salary:25
leads!

Name:Arron
Character:5(5)
Leadership:2(2)
Specification:5(5)
Design:5(5)
Implementation:1(2)
Salary:21
develops the specification!

A specification card was generated!
A specification card was generated!

Figure 7. Action Summary

Encounters can be positive or negative. Many encounters will be blocked based on the current progress of the project (e.g., the number of clear specification and design documents in play) or the characteristics of the project team (e.g., the leadership score of the leader or character score of the team member). A positive encounter can be blocked for example because a leader has insufficient leadership. A negative encounter can be blocked for example because all team members had enough character or there were enough clear specification cards. Blocking is automatically performed. Figure 8 presents an example positive encounter whereas Figure 9 presents an example negative encounter. A description of all encounters is presented in Appendix B.



Your Project Encounters an Opportunity

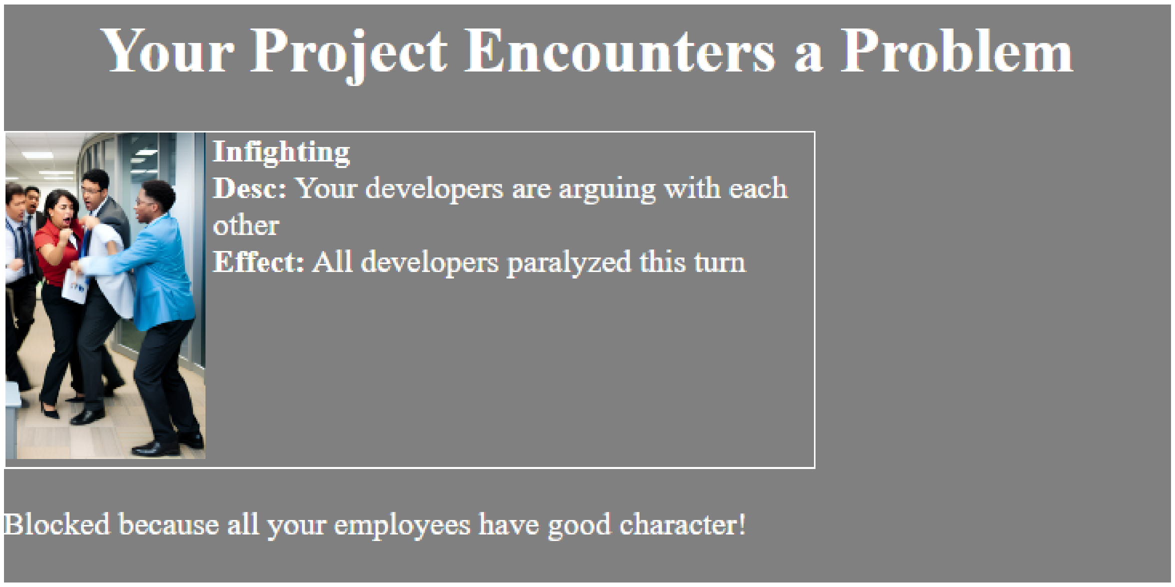
Performance Bonus
Desc: Your organization has signaled that successful completion will mean everyone gets a reward.
Effect: +1 to all skills
Cost: 5 per turn

Luck is a factor in projects. Sometimes your project encounters good luck. Sometimes your project faces bad luck. Being a good project manager is as much about maximizing opportunities and rolling with the punches as it is avoiding the problems that can be avoided.

Do you want to purchase this card?

- Yes
- No

Figure 8. A Positive Encounter



Your Project Encounters a Problem

Infighting
Desc: Your developers are arguing with each other
Effect: All developers paralyzed this turn

Blocked because all your employees have good character!

Figure 9. A Negative Encounter

Encounters can be enduring. An enduring encounter becomes a concept card that will appear on later work screens and affect the project team’s performance until removed.

The images for all encounters were generated with the AI artist Dream by Wombo (https://dream.ai/create).

4 Application of Project Wars

The literature on serious games has repeatedly emphasized that, while game quality is important, how the game is employed is also important for learning to occur (Keys & Wolfe, 1990). The instructor needs to play an active role (Wolfe, 1976), with debriefing an especially important part of successfully employing a serious game in the classroom (Certo, 1976).

Project Wars is intended to be employed in a manner similar to a Harvard case study. Students play the game/read the case on their own time, reflect on the game/case, and then the game/case is discussed in class over an extended time period (i.e., more than two hours) (Nohria, 2021).

The game assumes the student understands the concepts of specification, design, and implementation and knows that specification is important to design which is important to implementation. The game intentionally does not state this relationship so if students rush to implement a project, the negative results of doing so become visible. In other words, the game assumes the player knows at an intellectual level what the right thing to do is -- requirements first, then design, then implementation. However, under pressure, some players may attempt to shortcut this process. Students who are not taught these concepts can become very lost trying to understand why they keep losing.

Before the game is assigned, the instructor should show a brief live demo of the game and explain how it is played. Students are instructed to play the game for homework and are encouraged to play the game at least twice so they can learn from their mistakes. Students are then asked to write a reflection on the game. The reflection is marked as a homework assignment so students take the reflection seriously. The reflection must have at least four learning points, each learning point divided into four parts:

- The actionable learning point itself. The student must provide a one-sentence summary of the learning point structured in the form if <situation> I will <verb> or when <situation> I will <verb> or I will always/never <verb>. This forces the student to make the learning point concrete.
- A one to three-sentence explanation of the learning point.
- A negative example of the learning point. This is a situation in the game where the player's action made things worse.
- A positive example of the learning point. This is the reverse of the negative example where the player's action made things better.

An example lesson plan is presented in Appendix C. A key part of the lesson plan is for students to write reflections which the instructor reviews prior to class. The instructor quotes from the reflections and copies and pastes student writing into the themes below. The themes and student quotes are projected to students one theme at a time. Students are asked for each theme and quote to relate what happened in the game to the class. This serves as an icebreaker to get the discussion started. These themes are then employed for the class discussion.

The class discussion in turn serves as a springboard to other topics in project execution that the instructor may want to introduce in later classes. Such topics include:

- Management and Control of a Project Team - the role of self and input control is a topic the game allows a discussion on - see Section 4.2.
- Project Leadership - see Section 4.3
- The relationship between productivity, slack, overtime, and related concepts - see Section 4.4.
- Coordination Roles and Technology - see Section 4.5.

4.1 Disciplined Project Management

The main point of the game is to teach deferral of gratification, organizational mindfulness, and discipline in the face of adversity. The game is winnable under the following circumstances:

- The player hires team members of good character at the beginning; this is people with character of 5 or more. Later, as concept cards that create positive workplaces are obtained, a character of 4 becomes permissible. If such people are not available, the player is better off not hiring at all. Indeed, delaying the project because one does not have people of sufficient character provides a better outcome than hiring people with skill, but poor character. This is because people with poor character infect the project team, reducing morale and causing other team members to leave.

- The player dedicates resources to hiring and concept cards. As with real life IT projects, the core cost in Project Wars is salaries and while concept cards are "expensive," their material impact on overall cost is minor, but their ability to multiply productivity is large. Creating the right work environment (i.e., putting in place good concept cards, maximizes the ability of the team to do work, thereby allowing the team to complete work faster and more effectively (i.e., fewer bugs)).
- The player hires at least one leader with high skill (5 or more).
- The player focuses on specification first, then design, and then implementation. This is true for all IT projects. Even agile projects require clarity of specification and design; agile argues for minimal documentation, not no documentation.
- In the game, there are multiple unavoidable problems. One is "necessary new requirements" which causes a loss of specification documents. When this problem arises, the project team should focus on building clear specification documents, which causes a loss of design documents and implementation cards. The team should then focus on building clear design documents and clear implementation cards. Attempts to shortcut this effort can lead to a disaster.
- Project team members should be responsible for what they are good at. Having a team member who is not good at implementation perform an implementation will often make the problem worse – they create numerous bugs, resulting in work that must be redone. When they fix problems, they make the problems worse. In fact, it is often better that a team member with a poor skill level in something, simply be a reviewer.
- Project team members with poor skills can be used to review specifications, design, and implementation. At worst, they reveal no unclear/buggy cards.
- The player focuses on winning from the current situation rather than trying to recover from a loss. There are unavoidable problems in the game. Any attempt to shortcut those unavoidable problems (e.g., writing code after losing specification cards due to necessary requirements) makes the situation worse.

The mapping between optimal gameplay and the learning objectives is not one-to-one, partly because the learning objectives are not orthogonal. Nevertheless, a mapping can be discussed with students.

Deferral of Gratification: The player is penalized for making choices that trade off short-term benefits for long-term gains. A player who does not produce enough specification and design documents, who does not spend sufficient effort ensuring these are clear, and who does not spend enough time reviewing implementation, finds their work is often undone by negative concept cards. Furthermore, these problems accumulate until they affect morale which causes project team turnover. Deferral of gratification is also taught when the player hires the best candidate out of a mediocre pool, rather than hiring truly capable candidates. In the game, it is often better to accept the sunk cost of a poor candidate pool and to spend money to attempt to hire again, rather than to obtain a candidate who either creates vulnerabilities in the team because of poor character or creates problems for the team because of poor skill.

Organizational Mindfulness: Each of the subdimensions of organizational mindfulness is modeled in the game.

- *Preoccupation With Failure:* The game strongly rewards a conservative approach to project management, while punishing risk-taking in the form of shortcut taking. A player who makes sure they have 10 clear specification documents, and 10 clear design documents, always topping up as negative encounters remove these documents will face fewer negative encounters than one who ignores these losses to produce the final set of implementation cards. Indeed, it is a common scenario for one to have (for example) 10 clear requirements, 10 clear designs, and 8 bug-free implementation cards and be about to run out of budget only to encounter the unavoidable negative encounter "necessary requirements" that cause one to lose 2 requirements documents. At this point, it is very tempting to ignore the loss to produce and review the last 2 implementation cards and ship. Unfortunately, the loss of two requirements documents exposes the player to a number of negative encounters that can quickly undo the project. In other words, the player who worries about future negative encounters and their impact on the project schedule is more likely to succeed than one who thinks victory is just around the corner and at that point rushes to complete.

- *Reluctance to Simplify:* As mentioned previously, taking shortcuts in the game often hurts the player. A player who chooses not to assign a leader becomes overwhelmed by leadership-related negative encounters. A player who allocates someone with poor skill to perform a task gets saddled with unclear specifications, unclear design, or bugs. A player who shortcuts tasks finds negative encounters are not blocked. Basically, any failure to maintain project management discipline results in negative repercussions.
- *Sensitivity to Operations:* The game incorporates a number of elements designed to mimic politicking and morale in the workplace. First, the game explicitly includes a morale variable, and players who overuse the overtime option find their project team will quit. Second, there are various encounters like “executive meddling” and “executive favor” that explicitly model the intervention of others in the organizational environment. Finally, once the project shows signs of trouble (e.g., a budget shortfall), executives will become more involved, doing things like eliminating all positive concepts in an attempt to save costs.
- *Commitment to Resilience* is modeled as hiring candidates and concept cards. It is expensive to hire candidates with high character and to obtain concept cards because most carry some cost. Nevertheless, these costs pay for themselves because the player avoids negative encounters and obtains a boost in productivity from the concept cards.
- *Deferral to Expertise:* The game strongly penalizes the use of someone with low skill to perform a skilled task. Poor leaders will be unable to employ positive concepts and expose the project team to negative concepts. Poor implementers will create bugs that cost the team more time to fix than if a skilled implementer performed the coding. One particular skill-related issue in the game is the “bus factor,” where a temporary or permanent loss of a project team member can compromise a project if no one else has that skill (Cosentino et al., 2015). “Illness,” for example, is an unblockable negative concept in the game.

Discipline in the Face of Adversity: There are unavoidable negative encounters. Examples include “financial hardship,” which causes an unavoidable loss in the budget, and “illness” which causes a project team member to skip a turn. Random luck also plays a significant part in the game, if, for example, one can have no good candidates to hire. Because the budget shrinks at every turn, a player is put under pressure whenever unavoidable bad luck arises. In the game, the best outcome arises when the player accepts the loss and moves on. If a player acts in an undisciplined manner by, for example, forcing an ill team member to perform overtime to compensate, or hiring a bad candidate, the player absorbs substantial risk. Overtime decreases morale which incentivizes the specific team member and all team members to quit; bad candidates can create more problems for the project than they are worth.

4.2 Good Team Members

The game provides a foundation for a class discussion on what makes a good team member. Many of the negative encounters in the game center on different elements of poor character. Examples include negative encounters like infighting, slacking, and poor version control. These cards reflect a range of character flaws in project team members, including poor team engagement (infighting), laziness (slacking), and a lack of discipline (poor version control).

Self-control in the project team is an important discussion issue. One problem in IS projects stems from managing people with high expertise in areas in which the project manager has no expertise (Nidumolu & Subramani, 2003/2004). Substantial work in project teams is not always observable (Kirsch, 1996, 1997, 2004), so the project manager must trust the team, and the team member must act in a manner that is worthy of trust. This must be managed by selecting the right team members, known as input control (Wiener et al., 2016).

Another issue is turnover contagion (Bartunek et al., 2008; Felps et al., 2009). Every time a person leaves the project team, there is a negative impact on morale. Morale lowers the effect of every team member’s character, which makes it more likely that other team members quit. It is possible for one quitter to then create a cascade effect where more and more project team members quit in succeeding weeks, a phenomenon well documented in the literature (Porter & Rigby, 2021).

4.3 Understanding Leadership

An important issue in IT project teams is that team members are often highly skilled and know their jobs better than the leader. In such situations, it is pointless for the leader to instruct the team members. Much modern thinking, especially in the agile world, posits that the project leader is a facilitator rather than a traditional command manager (Tabaka, 2006).

In the game, the leader does not perform any production work. Nevertheless, the leader is absolutely necessary, because the leader blocks many negative encounters (e.g., executive meddling, frivolous requirements, poor inter-role coordination). These negative encounters frequently center around threats external to the project team (e.g., executive meddling, frivolous requirements) and coordination (e.g., poor inter-role coordination) (Kraut & Streeter, 1995).

The leader also enables certain good encounters (e.g., coordination mechanisms, social environment, standup meetings) (Plowman et al., 2007). Thus, in the game, effective leadership encapsulates someone whose role is to interface with both internal and external parties to overcome obstacles and maximize opportunities (Barber & Warn, 2005).

The game can be used to help the instructor reinforce the point that a busy leader is an ineffective leader (Bruch & Ghoshal, 2002; Lipkin, 2022). Busy leaders do not sense the environment for obstacles or opportunities. The best projects are those where the leader has nothing to do because the project is running smoothly.

4.4 Being Busy vs. Being Productive

Ironically, a good Project Wars player spends less time playing the game than a bad player. Furthermore, in successful games, many of the project team members will be idle (set to Do Nothing), because there is nothing for them to do. They might do things that make a minor impact (e.g., review each other's code), but, for example, a person with 6 specifications and 1 implementation rating has little to do once implementation is being performed. In contrast, if the project is not going well, the team loses specification documents, design documents, etc., it is quite common for all team members to be busy.

This allows the instructor to turn the class discussion to the concept of slack and idle time. Idle time and slack are an inherent part of projects and, if they are occurring, are indicators the project is proceeding well. Learning objectives include how to recognize the difference between being busy and being productive and how a good institutional structure enables project teams to be productive, but not busy (DeMarco, 2002).

One element of the game that captures the difference between busy work and productive work is an unclear document/bug. In the game, not performing tasks carefully and assigning people with low skills to perform tasks creates unclear specifications, unclear design, and bugs. These require further work to undo. Thus, just observing how many requirements documents, design documents, and/or implementation cards are generated per turn leads to poor management decision making. It is better to produce fewer requirements/design/implementation per turn if they are clear/bug-free, than to produce more that, subsequently, need to be corrected. This can lead to a discussion of the value of metrics like lines of code produced in IT projects. Relying on these metrics can lead to poor decisions. Should these metrics be captured?

A discussion of being busy versus being productive can then flow into a discussion of overtime in projects. Overtime is quite common in IT projects. However, overtime is not desirable because it creates the illusion of productivity (Collewet & Sauermann, 2017; Shepard & Clifton, 2000). One question to ask students is who creates the need for overtime in the game? The answer, of course, is the player. In other words, when overtime becomes necessary in a project, it is because the project manager is doing a poor job. That is, the project manager failed to correctly estimate the duration of the project, failed to consider specific risks, underbid a project (if a vendor), had little ability to manage external stakeholders, etc. and, as a result, the project team suffers because of the project manager's poor project management skills.

The game demonstrates there exists a role for overtime. Near the very end of the project, the loss in character caused by overtime has fewer drawbacks. For example, it is acceptable if team members quit if a project is about to be delivered, provided the deliverable is likely to be accepted! Thus, overtime is much like organizational adrenaline in that it is useful to push the project over the final hurdles. However, recurring use is likely to damage the project beyond repair if the work environment becomes so toxic no productive work can actually be accomplished.

4.5 Coordination

Savvy players will discover that, if they can obtain enough good concepts associated with aiding others, they are able to use the “assist other” commands to be more productive than having their team members produce implementation cards independently. Once aiding others becomes costless, it is possible to, for example, have one team member code, another review the code, and a third perform debugging on the same code. This allows code to be debugged in one turn. In contrast, a naïve player will have three separate team members perform each of these tasks on their own personal code, thereby taking three turns. In effect, having sufficient coordination concepts allows the project team to finish the project at least three turns earlier than they would otherwise. Furthermore, this is done because the coordination concepts allow for a different way of working. They appear to be the only good concepts (i.e., enduring positive encounters) in the game that allow this.

Our experience is that students, even graduate students, do not fully appreciate the importance of coordination as an issue in project management, especially for IT Project Management (Kraut & Streeter, 1995). Students who have never worked in project teams are accustomed to solo development work and may have never encountered the problems associated with coordinated work. Definitely, many of the technology improvements in IT project management, including scrum meetings (Sutherland, 2004), paired programming (Hannay et al., 2009), user sitting with developer (Beck et al., 2001), Kanban board (Gross & McInnis, 2003), version control systems (Spinellis, 2005), and model-view-controller design pattern (Krasner & Pope, 1988) are fundamentally technologies for improving coordination. DevOps (Leite et al., 2019) similarly focuses on breaking down functional silos.

A discussion of coordination strategies in the game allows the instructor to discuss the concept of coordinating implementation in the class and the importance of coordination and coordinating technologies.

5 Evaluation

Evaluation of Project Wars has been an ongoing process and a variety of evaluations have been conducted. These include the following.

Alpha Test. The first author drafted members of his family to alpha-test the game. Alpha testing occurred until no identified bug caused a crash. Two of the testers were high school age, one was a college student. Two testers were female, one was male.

Beta Test. To beta test the game prior to use in class, the first author offered testing of the game as extra credit in his web development class. Web development was considered an appropriate class to test the game in as the techniques to create the game were the same techniques taught in class. Students were asked to play the game and answer three open-ended questions:

- What can be done to make the game more fun?
- Identify a bug and give enough information to reproduce it.
- Describe your experience with the game. Provide comments (negative comments welcome)

Out of 32 eligible students, 8 students submitted the extra credit for a response rate of 25%. Six students were male, two were female. The first author teaches in an institution that assigns pure letter grades (no + or -). There is thus a 10% grade band difference between each letter grade. By the time students did the activity, students could calculate their final grade to a reasonable degree of accuracy. Therefore, for many students, performing the extra credit assignment would not materially change their letter grade; hence, there is a low response rate.

The game tracks logins and whether the game was completed successfully or reset and when. None of the eight students successfully completed the game.

Classroom Test. The first author applied the game in an IT Project Management class. The game was a mandatory homework assessment. Students were required to play the game twice, where, in each run, they had to either win the game or play until they had taken 20 turns and at least 30 minutes had passed. The logs show several students played more than two runs of the game. In total, students played the game 67 times (i.e., 35 times more than the mandated amount (32)). In the 67 runs, only 1 run resulted in a win condition.

Students then had to write a reflection on the game worth up to 16 marks. See Appendix C.3 (Homework) on the nature of the reflection. However, briefly, students had to identify four actionable learning points and demonstrate that the learning points were observed in the game. Students were awarded up to 4 marks for each learning point. One mark was awarded for an explicit statement of the learning point, one mark for elaborating on that statement, and two marks for examples in the two mandatory runs. An actionable learning point was defined as a learning point that converts explicitly into an action the student should take. It is typically structured as an "if then" sentence or an "always" or "never" sentence. For example, "If given a choice between someone skilled and expensive and someone not skilled but not expensive, I will hire the skilled and expensive person" and "I will always ensure all my requirements are clear before I begin coding."

The mean, median, and mode score students received were 13.9 (SD= 2.9), 15, and 16 (5 students). The lowest score was 8 (2 students). This means students articulated on average three things they learned from the simulation. Most marks were deducted for failing to clearly articulate the learning point. For example, a student who could not articulate the learning point but could give a positive and negative example of it would receive 2 out of 4 marks.

Students were also asked to rate how much they learned from the game on a scale from 1-5. Mean, median, and mode were respectively 3.85 (SD=0.7), 4, and 4 (10 students), respectively. Certainly, subject biases can be confounding given the survey was conducted in the developer's own class. The students were also asked the three open-ended questions asked of the beta testers.

A summary of tester demographics is in Table 2.

Table 2. Summary of Tester Demographics

Test	Total Testers	Gender		Educational Attainment		
		Male	Female	High School	Undergraduate	Graduate
Alpha	3	1	2	2	1	0
Beta	8	6	2	0	5	3
In-Class	16	13	3	0	8	8

The remainder of the evaluation explores subject feedback on improving the game and the subject's experience. We do not discuss bugs because, although subjects found bugs, they were very minor.

5.1 What Can Be Done to Make the Game More Fun?

A common thread students reported was clunkiness in the user interface. The game employs static images rather than animations, and interactions involve clicking rather than simulating body motion.

> *There are quite a few things that could make the game more fun for sure. One would be adding a 2D or even 3D mini office room where the user can move their character to different parts of the room to access the hiring, managing, help, about, and anything else [n]ecessary. Instead of just clicking on the characters in the web browser the character could walk into an office filled with the people you've hired and you could go from person to person setting their roles (and making them save every round until they were changed.* (alpha test group)

A common criticism was that the game was not aesthetically pleasing.

> *We can also make the game more responsive instead of keeping all the rows on a single page without scrolling, we can change the layout for mobiles and tablets. We can also add more animations and give more graphical information instead of pictures for errors and in the game.* (in-class group)

Students also suggested adding other kinds of scenarios or features. This appears to be an indication the game is enjoyable. People ask for features when software does its job.

> *By providing the interactive exercises or mini-games that mimic project circumstances found in the real world. In order to engage participants and improve their project management abilities, this might involve team-building activities, problem-solving assignments, or decision-making simulations.* (in-class group)

The in-class students also requested a turn-by-turn log of events so students could more effectively write their reflections. This is being considered as a future feature.

5.2 Experience

The general consensus is the game is complex to play. A common theme was a desire for better online tutorials and help screens.

> *It was very interesting, I would say a little complex too. Despite looking at how to play it, there were so many different things to keep in mind while going through and selecting candidates that it really did make me feel like some kind of hiring manager. A few suggestive criticisms are to make some kind of interactive tutorial. I practice better with hands-on training rather than trying to learn from a video.* (alpha test group)

The game models a number of complex decision problems and attempts to simulate real world outcomes of getting those wrong.

> *Some of the events are really frustrating and take away from the game. The events that disabled all employees were especially frustrating and felt like they derailed an entire run if they came too early.* (in-class)

In the game, three events paralyze all developers. These are “infighting,” “firefighting” and “customer anxiety.” “Customer anxiety” only triggers when you run out of budget so it is unlikely to occur in the early game. Infighting occurs when you have developers with poor character and firefighting occurs when the project leader has low leadership.

Mistakes made were punishing in the game.

> *I think the morale is a little unbalanced in that it can kill your run very quickly with people leaving while it doesn't feel like there's a ton of benefit to having good team morale.* (in-class)

It should be noted that the player actually commented on a feature of the game, rather than a bug. The game simulates the idea of turnover contagion (Bartunek et al., 2008; Felps et al., 2009). Once people begin to leave the project, others will consider an exit strategy. This is what makes Project Wars a serious game. Entertainment value is sacrificed for the learning point.

5.3 Summary

The feedback suggests the game conveys the concepts it intends to convey. Furthermore, the fact that students play the game more than the assigned amount suggests players obtain some enjoyment from playing the game.

To be effective, the game requires support from the instructor who must demonstrate how the game is played at the beginning and discuss insights from the game after students have finished playing (Certo, 1976; Wolfe, 1976). This is not only because doing so is good practice, but also because the game is inherently complex and attempts to teach complex concepts. A discussion of how to do this is found in Appendix C.

6 Conclusion

This paper presents an overview of Project Wars, a free online serious game designed to teach concepts associated with project execution. The game has been trialed and has been demonstrated to be playable and conveys the concepts it is designed to convey.

The game provides several notable contributions to the IS academic community. First and foremost, there are essentially no free serious games focused on project execution available for HyFlex classes. Project Wars thus fills an important vacuum in IS educational materials.

Second, Project Wars actively embeds deferral of gratification, organizational mindfulness, and discipline in the face of adversity into gameplay. We would emphasize these are concepts that students can often appreciate on a purely cognitive level, but students will often not behave according to these concepts when faced with actual adversity. Project Wars makes students feel the concepts, thus hopefully actually shaping future behavior.

Third, it is our intention that Project Wars serves as a useful foundation from which to understand that it is difficult to codify concepts that should be part of an IS curriculum and how they can be taught. An incredibly large part of the work IS employees carry out is difficult to codify and yet can be abstracted to articulable concepts. Regardless, Project Wars has been shown to be fit for its intended purpose. Future work is needed to refine the game and test it at multiple levels of courses and student capabilities to assess how effective it is in teaching concepts and engaging students in the important topic of project management.

References

Abt, C. C. (1987). *Serious games*. University Press of America.

Akkermans, H., & van Helden, K. (2002). Vicious and virtuous cycles in ERP implementation: A case study of interrelations between critical success factors. *European Journal of Information Systems*, *11*(1), 35-46.

Anderson, T., Prue, G., McDowell, G., Stark, P., Wilson, C. B., Wisener, L. G., Kerr, H., Caughers, G., Rogers, K., Cook, L., Craig, S., Alanazi, A., & Mitchell, G. (2024). Co-design and evaluation of a digital serious game to promote public awareness about pancreatic cancer. *BMC Public Health*, *24*(570), 1-10.

Baker, A., Navarro, E. O., & Hoek, A. v. d. (2005). An experimental card game for teaching software engineering processes. *Journal of Systems and Software*, *75*(1-2), 3-16.

Barber, E., & Warn, J. (2005). Leadership in project management: From firefighter to firelighter. *Management Decision*, *43*(7/8), 1032-1039.

Bartunek, J. M., Huang, Z., & Walsh, I. J. (2008). The development of a process model of collective turnover. *Human Relations*, *61*(1), 5-38.

Baumeister, R. F., Leith, K. P., Muraven, M., & Bratslavsky, E. (2002). Self-regulation a key to success in life. In D. Pushkar, W. M. Bukowski, A. E. Schwartzman, D. M. Stack, & D. R. White (Eds.), *Improving competence across the lifespan* (pp. 117-132). Springer.

Beatty, B. J. (2019). *Hybrid-flexible course design: Implementing student-directed hybrid classes*. EdTech Books.

Beck, K., Grenning, J., Martin, R. C., Beedle, M., Highsmith, J., Mellor, S., Bennekum, A. v., Hunt, A., Schwaber, K., Cockburn, A., Jeffries, R., Sutherland, J., Cunningham, W., Kern, J., Thomas, D., Fowler, M., & Marick, B. (2001). *Manifesto for agile software development*. http://agilemanifesto.org

Bergeron, B. (2006). *Developing serious games*. Charles River Media.

Bollin, A., Hochmüller, E., & Mittermeir, R. T. (2011). *Teaching software project management using simulations.* 2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T), Honolulu, HI.

Brenčič, V. (2009). Employers' hiring practices, employment protection, and costly search: A vacancy-level analysis. *Labour Economics*, *16*(5), 461-479.

Brooks, F. P. (1982). *The mythical man-month: Essays on software engineering*. Addison-Wesley.

Bruch, H., & Ghoshal, S. (2002, February). Beware the busy manager. *Harvard Business Review*.

Butler, B. S., & Gray, P. H. (2006). Reliability, mindfulness, and information systems. *MIS Quarterly*, *30*(2), 211-224.

Calderón, A., & Ruiz, M. (2013). *ProDec: A serious game for software project management training*. The Eighth International Conference on Software Engineering Advances,

Calderón, A., Ruiz, M., & O'Connor, R. V. (2017). *ProDecAdmin: A Game Scenario Design Tool for Software Project Management Training* 24th European Conference on Systems, Software and Services Process Improvement, Ostrava, Czech Republic.

Carmeli, A., Levi, A., & Peccei, R. (2021). Resilience and creative problem-solving capacities in project teams: A relational view. *International Journal of Project Management*, *39*(5), 546-556.

Carpenter, M. J. (2021). Replaying colonialism: Indigenous national sovereignty and its limits in strategic videogames. *American Indian Quarterly*, *45*(1), 33-55.

Caulfield, C., Maj, S. P., Xia, J. C., & Veal, D. (2012). Shall we play a game? *Modern Applied Science*, *6*(1), 2-15.

Caulfield, C., & Xia, J. C. (2011). A systematic survey of games used for software engineering education. *Modern Applied Science*, *5*(6), 28-43.

Caulfield, C. W., Veal, D., & Maj, S. P. (2011). Teaching software engineering project management – A novel approach for software engineering programs. *Modern Applied Science*, *5*(5), 87-104.

Cermak-Sassenrath, D., Solmaz, S., Alfaro, J. L. D., & Udeozor, C. (2023). Bunno's fabulous soap-making challenge–Educational concept, game design, and initial evaluation. *Chemical Engineering & Technology*, *46*(6), 1055-1312.

Certo, S. C. (1976). The experiential exercise situation: A comment on instructional role and pedagogy evaluation. *Academy of Management Review*, *1*(3), 113-116.

Collewet, M., & Sauermann, J. (2017). Working hours and productivity. *Labour Economics*, *47*, 96-106.

Collofello, J. S. (2000). University/industry collaboration in developing a simulation-based software project management training course. *IEEE Transactions on Education*, *43*(4), 389-393.

Converse, P. D., Piccone, K. A., & Tocci, M. C. (2014). Childhood self-control, adolescent behavior, and career success. *Personality and Individual Differences*, *59*, 65-70.

Cooke-Davies, T. J., Crawford, L. H., & Lechler, T. G. (2009). Project management systems: Moving project management from an operational to a strategic discipline. *Project Management Journal*, *40*(1), 110-123.

Coombs, C. R. (2015). When Planned IS/IT Project benefits are not realized: A study of inhibitors and facilitators to benefits realization. *International Journal of Project Management*, *33*(2), 363-379.

Cosentino, V., Izquierdo, J. L. C., & Cabot, J. (2015). *Assessing the bus factor of git repositories*. 22nd International Conference on Software Analysis, Evolution, and Reengineering, Montreal, QC.

Dallaqua, M. F., Nunes, B., & Carvalho, M. M. (2023). Serious games research streams for social change: Critical review and framing. *British Journal of Educational Technology*, *55*, 460-483.

Dantas, A., Barros, M., & Werner, C. (2004). *A simulation-based game for project management experiential learning*. International Conference on Technology in Education, Hong Kong, China.

Davidovich, L., Parush, A., & Shtub, A. (2006). Simulation-based learning in engineering education: Performance and transfer in learning project management. *Journal of Engineering Education*, *95*(4), 289-299.

de Vries, P. W., & Knol, E. (2011). *Serious gaming as a means to change adolescents' attitudes towards saving energy: Preliminary results from the enercities case*. EDEN Annual Conference Dublin, Ireland.

DeMarco, T. (2002). *Slack: Getting past burnout, busywork, and the myth of total efficiency*. Broadway Books.

Felps, W., Mitchell, T. R., Hekman, D. R., Lee, T. W., Holtom, B. C., & Harman, W. S. (2009). Turnover contagion: How coworkers' job embeddedness and job search behaviors influence quitting. *Academy of Management Journal*, *52*(3), 545-561.

Feng, C.-Y., & Chen, M.-P. (2014). The effects of goal specificity and scaffolding on programming performance and self-regulation in game design. *British Journal of Educational Technology*, *45*(2), 285-302.

Fey, S., & Kock, A. (2022). Meeting challenges with resilience – How innovation projects deal with adversity. *International Journal of Project Management*, *40*(8), 941-950.

Finkelstein, A. (1993, February). *Report of the inquiry into the London ambulance service*. International Workshop on Software Specification and Design.

Gozluklua, B., & Sterman, J. (2022a). A management flight simulator to catalyze learning about complex projects. In G. Winch, M. Brunet, & D. Cao (Eds.), *Research handbook on complex project organizing*. Edward Elgar.

Gozluklua, B., & Sterman, J. (2022b). System dynamics to understand and improve the performance of complex projects. In G. Winch, M. Brunet, & D. Cao (Eds.), *Research handbook on complex project organizing*. Edward Elgar.

Gross, J. M., & McInnis, K. R. (2003). *Kanban made simple: Demystifying and applying Toyota's legendary manufacturing process*. AMACOM.

Hannay, J. E., Dybå, T., Arisholm, E., & Sjøberg, D. I. K. (2009). The effectiveness of pair programming: A meta-analysis. *Information and Software Technology*, *51*(7), 1110-1122.

Hellström, M. M., Jaccard, D., & Bonnier, K. E. (2023). A systematic review on the use of serious games in project management education. *International Journal of Serious Games*, *10*(2), 3-24.

Hodgson, D. (2002). Disciplining the professional: The case of project management. *Journal of Management Studies*, *39*(6), 803-821.

Howitt, P., & McAfee, R. P. (1987). Costly search and recruiting. *International Economic Review*, *28*(1), 89-107.

Ishikawa, A., Fujimoto, S., & Mizuno, T. (2022). Statistical properties of labor productivity distributions. *Frontiers in Physics*, *10*.

Jain, A., & Boehm, B. (2006). *SimVBSE: Developing a game for value-based software engineering.* 19th Conference on Software Engineering Education & Training, Turtle Bay, HI.

Joseph, D., Ng, K.-Y., Koh, C., & Ang, S. (2007). Turnover of information technology professionals: A narrative review, meta-analytic structural equation modeling, and model development. *MIS Quarterly*, *31*(3), 547-577.

Keys, B., & Wolfe, J. (1990). The role of management games and simulations in education and research. *Journal of Management*, *16*(2), 307-336.

Kim, Y. J., Knowles, M. A., Scianna, J., Lin, G., & Ruipérez-Valiente, J. A. (2023). Learning analytics application to examine validity and generalizability of game-based assessment for spatial reasoning. *British Journal of Educational Technology*, *54*(1), 355–372.

Kincaid, J. P., & Westerlund, K. K. (2009). *Simulation in education and training*. 2009 Winter Simulation Conference, Austin, TX.

King, M. (2021). The possibilities and problems of Sid Meier's civilization in history classrooms. *History Teacher*, *54*(3), 539-567.

Kingsley, T., & Grabner-Hagen, M. M. (2023). It's a winning condition! Examining the impact of meaningful gamification with preservice teachers. *College Teaching*, *71*(4), 260-272.

Kirsch, L. J. (1996). The management of complex tasks in organizations: Controlling the systems development process. *Organization Science*, *7*(1), 1-21.

Kirsch, L. J. (1997). Portfolios of control modes and IS project management. *Information Systems Research*, *8*(3), 215-239.

Kirsch, L. J. (2004). Deploying common systems globally: The dynamics of control. *Information Systems Research*, *15*(4), 374-395.

Knauss, E., Schneider, K., & Stapel, K. (2008). *A game for taking requirements engineering more seriously*. Third International Workshop on Multimedia and Enjoyable Requirements Engineering, Catalunya, Spain.

Kotter, J. P. (1996). *Leading change*. Harvard Business School Press.

Kotter, J. P., & Cohen, D. S. (2002). *The heart of change*. Harvard Business School Press.

Krasner, G. E., & Pope, S. T. (1988). A cookbook for using the model view controller user interface paradigm in smalltalk. *Journal of Object Oriented Programming*, *1*(3), 26-49.

Kraut, R. K., & Streeter, L. A. (1995). Coordination in software development. *Communications of the ACM*, *38*(3), 69-81.

Kvalnes, Ø. (2016). Living with the unknown unknown: Uncertainty in projects. *Project Management Journal*, *47*(3), 101-108.

Leite, L., Rocha, C., Kon, F., Milojicic, D., & Meirelles, P. (2019). A survey of DevOps concepts and challenges. *ACM Computing Surveys*, *52*(6).

Li, Y., Chen, D., & Deng, X. (2024). The impact of digital educational games on student's motivation for learning: The mediating effect of learning engagement and the moderating effect of the digital environment. *PLoS One*, *19*(1), 1-21.

Lin, C., & Pervan, G. (2003). The practice of IS/IT benefits management in large Australian organizations. *Information & Management*, *41*(1), 13-24.

Lipkin, N. (2022, April 26). *Why many new leaders become too busy to win*. Forbes. https://www.forbes.com/sites/nicolelipkin/2022/04/26/why-many-new-leaders-become-too-busy-to-win/?sh=176aee5dd081

Loch, C. H., DeMeyer, A., & Pich, M. (2006). *Managing the unknown: A new approach to managing high uncertainty and risk in projects*. John Wiley & Sons.

Mähring, M. (2002). *IT project governance*. EFI: The Economic Research Institute.

Mazarakis, A., & Bräuer, P. (2023). Gamification is working, but which one exactly? Results from an experiment with four game design elements. *International Journal of Human-Computer Interaction*, *39*(3), 612-627.

McKnight, D. H., Phillips, B., & Hardgrave, B. C. (2009). Which reduces IT turnover intention the most: Workplace characteristics or job characteristics? *Information & Management*, *46*(3), 167-174.

Merriam-Webster. (n.d.). *Game (noun)*. https://www.merriam-webster.com/dictionary/game

Meskill, C. (1990). Where in the world of English is Carmen Sandiego? *Simulation & Gaming*, *21*(4), 457-460.

Mischel, W., Shoda, Y., & Rodriguez, M. L. (1989). Delay of gratification in children. *Science*, *244*(4907), 933-938.

Navarro, E. O., & Hoek, A. v. d. (2004). *SIMSE: An interactive simulation game for software engineering education*. 7th IASTED International Conference on Computers and Advanced Technology in Education, Kauai, HI.

New Zealand Government. (2013). *Report of the ministerial inquiry into the Novopay project*. http://www.education.govt.nz/assets/Documents/Ministry/Information-releases/Novopay-information-release/MIN130501InquiryReport.pdf

Nidumolu, S. R., & Subramani, M. R. (2003/2004). The matrix of control: Combining process and structure approaches to managing software development. *Journal of Management Information Systems*, *20*(3), 159-196.

Nohria, N. (2021, December). What the case study method really teaches. *Harvard Business Review*.

Noll, J., Butterfield, A., Farrell, K., Mason, T., McGuire, M., & McKinley, R. (2014). *GSD Sim: A global software development game*. IEEE International Conference on Global Software Engineering Workshop, Shanghai, China.

Novopay Staff. (2017). *Novopay unravelled*. School Executive Officers' Association Conference 2017, Auckland, NZ. https://www.confer.co.nz/seo2017/wp-content/uploads/2017/07/IPANZ-EdPay-looped-manual-5May.pdf

Ofosu-Ampong, K. (2020). The shift to gamification in education: A review on dominant issues. *Journal of Educational Technology Systems*, *49*(1), 113-137.

Orlikowski, W. J. (2006). Material knowing: The scaffolding of human knowledgeability. *European Journal of Information Systems*, *15*(5), 460-466.

Parker, S. K., & Skitmore, M. (2005). Project management turnover: Causes and effects on project performance. *International Journal of Project Management*, *23*(3), 205-214.

Pavez, I., Gómez, H., Laulié, L., & González, V. A. (2021). Project team resilience: The effect of group potency and interpersonal trust. *International Journal of Project Management*, *39*(6), 697-708.

Pfahl, D., Laitenberger, O., Ruhe, G., Dorsch, J., & Krivobokova, T. (2004). Evaluating the learning effectiveness of using simulations in software project management education: Results from a twice replicated experiment. *Information and Software Technology*, *46*, 127-147.

Pinto, J. K. (2000). Understanding the role of politics in successful project management. *International Journal of Project Management*, *18*(2), 85-91.

Plant, R., & Willcocks, L. (2007). Critical success factors in international ERP implementations: A case research approach. *Journal of Computer Information Systems*, *47*(3), 60-70.

Plowman, D. A., Solansky, S., Beck, T. E., Baker, L., Kulkarni, M., & Travis, D. V. (2007). The role of leadership in emergent, self-organization. *The Leadership Quarterly*, *18*(4), 341-356.

Porter, C. M., & Rigby, J. R. (2021). The turnover contagion process: An integrative review of theoretical and empirical research. *Journal of Organizational Behavior*, *42*(2), 212-228.

Project Management Institute. (2021). *A guide to the project management body of knowledge* (Seventh ed.). Project Management Institute.

Project Management Institute. (2023). *Process groups: A practice guide*. Project Management Institute.

Ramasesh, R. V., & Browning, T. R. (2014). A conceptual framework for tackling knowable unknown unknowns in project management. *Journal of Operations Management*, *32*(4), 190-204.

Remus, U., & Wiener, M. (2010). A multi-method, holistic strategy for researching critical success factors in IT projects. *Information Systems Journal*, *20*(1), 25-52.

Roe, J., & Elton, J. (1998, March/April). Bringing discipline to project management. *Harvard Business Review*.

Rollag, K., & Parise, S. (2005). The Bikestuff simulation: Experiencing the challenge of organizational change. *Journal of Management Education*, *29*(5), 769-787.

Royce, W. W. (1970). Managing the development of large software systems: Concepts and techniques. In *Westcon Technical Papers* (pp. A1/1-A/1-9).

Rumeser, D., & Emsley, M. (2018). A systematic review of project management serious games: Identifying gaps, trends, and directions for future research. *Journal of Modern Project Management*, *6*(1), 48-59.

Sackman, H., Erikson, W. J., & Grant, E. E. (1968). Exploratory experimental studies comparing online and offline programming performance. *Communications of the ACM*, *11*(1), 3-11.

Sharp, H., & Hall, P. (2000). *An interactive multimedia software house simulation for postgraduate software engineers*. International Conference on Software Engineering, Limerick, Ireland.

Shepard, E., & Clifton, T. (2000). Are longer hours reducing productivity in manufacturing? *International Journal of Manpower*, *21*(7), 540-553.

Spinellis, D. (2005). Version control systems. *IEEE Software*, *22*(5), 108-109.

Srinivasan, J., & Lundqvist, K. (2007). *A constructivist approach to teaching software processes*. 29th International Conference on Software Engineering, Minneapolis, MN.

Standish Group. (2004). *CHAOS Demographics and Project Resolution*.

Standish Group. (2009). *Chaos Summary 2009*.

Standish Group. (2020). *CHAOS 2020: Beyond Infinity*.

Sutcliffe, A., & Sawyer, P. (2013). *Requirements elicitation: Towards the unknown unknowns*. 21st IEEE International Requirements Engineering Conference (RE), Rio de Janiero, Brazil.

Sutherland, J. (2004). *Agile development: Lessons learned from the first scrum*. https://www.torak.com/files/Lessons%20Learned%20From%20The%20First%20Scrum%20by%20Dr.%20Jeff%20Sutherland.pdf

Swanson, E. B., & Ramiller, N. C. (2004). Innovating mindfully with information technology. *MIS Quarterly*, *28*(4), 553-583.

Tabaka, J. (2006). *Collaboration explained: Facilitation skills for software project leaders*. Addison-Wesley.

Taran, G. (2007). *Using games in software engineering education to teach risk management*. 20th Conference on Software Engineering Education & Training, Dublin, Ireland.

Wangenheim, C. G. V., Savi, R., & Borgatto, A. F. (2012). DELIVER! – An educational game for teaching earned value management in computing courses. *Information and Software Technology*, *54*(3), 286-298.

Weick, K. E., & Sutcliffe, K. M. (2006). Mindfulness and the quality of organizational attention. *Organization Science*, *17*(4), 514-524.

Westera, W. (2019). Why and how serious games can become far more effective: Accommodating productive learning experiences, learner motivation and the monitoring of learning gains. *Journal of Educational Technology & Society*, *22*(1), 59-69.

Wiener, M., Mähring, M., Remus, U., & Saunders, C. (2016). Control configuration and control enactment in information systems projects: Review and expanded theoretical framework. *MIS Quarterly*, *40*(3), 741-774.

Wolfe, J. (1976). The effects and effectiveness of simulations in business policy teaching applications. *Academy of Management Review*, *1*(2), 47-56.

Wozencroft, A. J., Pate, J. R., & Griffiths, H. K. (2014). Experiential learning and its impact on students' attitudes toward youth with disabilities. *Journal of Experiential Education*, *38*(2), 1-15.

Zwikael, O., & Smyrk, J. R. (2019). *Project management*. Springer.

Appendix A: Prior Serious Games in IT Project Management/Software Engineering

Table A.1. Serious Games for IT Project Management/Software Engineering Instruction

Game Name	Purpose	Type	Citation	Remarks
Physical				
BikeStuff	User/developer communication	Discussion-based using paper documents	(Rollag & Parise, 2005)	One key learning point is how role identity shapes communication pathways (e.g., people in the same role sit together without prompting). This is difficult to replicate in an online environment. Also, stressful elements in the game (e.g., the instructor periodically shouts scores in the room) are difficult to replicate in an online asynchronous environment.
SIMSOFT		Board game	(Caulfield et al., 2012; Caulfield et al., 2011)	Difficult to replicate turn-based multiplayer in an online asynchronous environment.
Deliver!		Board game	(Wangenheim et al., 2012)	Difficult to replicate turn-based multiplayer in an online asynchronous environment.
Unnamed	Understanding requirements engineering	Discussion-based using paper documents	(Srinivasan & Lundqvist, 2007)	Difficult to replicate turn-based multiplayer in an online asynchronous environment.
Unnamed	Risk management	Board game	(Taran, 2007)	Difficult to replicate turn-based multiplayer in an online asynchronous environment.
Problems and Programm	Disciplined software engineering-requirements	Card game	(Baker et al., 2005)	Abandoned. Difficult to replicate turn-based multiplayer in an online

ers	must be done before design and implementation.			asynchronous environment.
Software App				
ProDec	Cost estimation, risk analysis	Java-based game	(Calderón & Ruiz, 2013; Calderón et al., 2017)	No obvious way to obtain it.
SimSE	Project execution life cycle. Multitasking.	Java-based stand alone game	(Navarro & Hoek, 2004)	Abandoned. Version on the official website does not run on the modern Java Runtime Library making it unplayable.
SIMVBSE	Value-based software engineering.	Player visits multiple rooms and interacts with different people to perform tasks.	(Jain & Boehm, 2006)	Abandoned. Website is not functional.
Therefore iManage	Systems dynamics simulator	Developed using the iThink simulation engine	(Collofello, 2000)	More a systems dynamics simulation than a serious game.
The Incredible Manager	Systems dynamics simulator with a focus on hiring and allocation of talent to tasks	It Is difficult to get a sense of what the game is like from the paper description	(Dantas et al., 2004)	Difficult to obtain an appreciation of game from the description. The game does not appear generally available.
GSD-SIM	Global software development coordination. The focus is on how geographic distance creates difficulties.	Node-JS	(Noll et al., 2014)	Reports an error when trying to compile. JavaScript reports an error on the game start.
PMT	Appears to be about understanding Gantt charts and network diagrams		(Davidovich et al., 2006)	Appears to be unfinished.
Open University M880 Game	Graphical Q&A Environment		(Sharp & Hall, 2000)	Integrated as part of a course. No obvious way to obtain it.
Unnamed Cost Estimation Simulation	Not well described. It is just called a System Dynamics	Not well described	(Pfahl et al., 2004)	Not well described.

	(SD) Simulation			
A Media Education Initiative for Software Engineering (AMEISE)	Project execution life cycle. Recruiting talent. Allocation of talent to tasks.	Java based client-server game. Uses the SESAM (Software-Engineering-Simulation through Animated Models) game engine.	(Bollin et al., 2011)	Interface is a pseudo command line, thus there is considerable overhead to explaining how to use the simulation.
Web-Based				
MIT Project Manage-ment Game	Maximization of inputs	Web-based	(Gozluklua & Sterman, 2022a, 2022b) https://forio.com/app/mit/project-management/#/	Very operations management/optimi-zation orientation
ThatPMGame	Visualization of Gantt chart	Web-based	Thatpmgame.com	Just a visualization of a Gantt chart
AbleSim	Allocation of resources to a network diagram	Web-based	Ablesim.com	Simple allocation of human resources based on a network diagram.
Unnamed Software Quantum Game	Flow of requirements from user to code	Web-based	(Knauss et al., 2008) www.se.uni-hannover.de/en/qgame	Website link not working
SimulTrain	Project execution life cycle. Recruiting talent. Allocation of talent to tasks.	Web-based	https://www.meritcd.com/simulation/higher-ed	Expensive group license (USD 905 for a license for 4 people as of this writing). Including explanation, debriefing, etc. game is typically played over two full work days (16 hours).
SimProject	Project execution life cycle. Recruiting talent. Allocation of talent to tasks.	Web-based	https://simproject.com	Expensive (USD 50 per player for 3 runs) Class overhead to explain how the game works because various errors are reported if the student forgets to do something. One game can take six hours or more to complete.

Appendix B: Project Wars Encounters

This is a list of all current encounters in the game.

Card Name	Card Description	Effect
Backups and redundancy	You have implemented a system so damage from code loss is minimized.	When in play as a concept, the card blocks one loss of code and then is discarded.
Modular design	Your implementation of abstractions and other good design principles prevents design problems from occurring.	All developers get +1 design skill
Pleasant Work Atmosphere	You have set in place policies so people want to come to work. Card is blocked and discarded if leader has leadership < 6.	+1 to character
Strong executive leadership	Your executive sponsor is blocking problems by the users.	This card will block one loss of requirements. It is then discarded.
Bus factor management	You have implemented policies so there is no single point of failure. Card is blocked and discarded if leader has leadership < 5.	When in play as a concept, gives +1 Aid. Total aid cannot be greater than 0.
Collaborative software	You have put in place software to facilitate team communication.	When in play as a concept, gives +1 Aid. Total aid cannot be greater than 0.
Close customer engagement	You are engaged with the customer throughout the product life cycle.	Adds a free requirements card.
Coordination mechanisms	You have implemented policies that better enable coordination. Card is blocked and discarded if leader has leadership < 5.	When in play as a concept, gives +1 Aid. Total aid cannot be greater than 0.
Design Tool	Some design tasks are automated.	All developers get +1 design skill
Documentation standards	You have clear documentation standards in place which makes understanding others' work easier.	When in play as a concept, gives +1 Aid. Total aid cannot be greater than 0.
Domain expert	Your background and experience mean you have a good understanding of the customer's business	You gain 2 requirements cards.
Performance Bonus	Your organization has signaled that successful completion will mean everyone gets a reward.	+1 to all skills
Social environment	You have created an environment which brings out the best in your employees. Card is blocked and discarded if leader has leadership < 5.	+1 to all skills

Standup meeting	You have a short meeting at the beginning of the day so everyone has a clear idea of each others' status. Card is blocked and discarded if leader has leadership < 5.	+1 to aid others
Coding Framework	You have mandated use of a coding framework.	All developers get +1 implementation skill
Messy code	Lack of standards in coding means the code is hard to read. Card is blocked and discarded if there are 5 or more design cards.	-2 to aid others
Poor inter-role coordination	There is poor communication between the specification, design and implementation teams. Card is blocked and discarded if the leader has leadership > 4.	-2 to aid others
Short sighted design	Your shortcuts in design have led to glaring errors in code. Card is blocked and discarded if there are 8 or more design cards.	+2 code length. +2 inspected code.
Inflexible architecture	Coordinated work is made harder because of the way the system is designed. Card is blocked and discarded if there are 9 or more design cards.	-2 on aid others
Poor interfaces	Lack of good interface design means your implementers' work doesn't mesh well together. Card is blocked and discarded if there are 9 or more design cards.	-2 to aid others
Poor cohesion	The workplace makes it hard for workers to help each other. Card is blocked and discarded if the leader has leadership > 4.	-2 to aid others
Fundamental problem	Your fundamental misunderstanding of a requirement has made the project more challenging.	+2 to total code length. +2 to number of cards inspected.

	Card is blocked and discarded if there are 2 or more unclear specification cards.	
Crummy work environment	The work environment is so bad, people hate to show up for work. This card is blocked and discarded if the leader has a leadership > 4.	-1 on all stats for all developers
Better job	A developer quit for another job. Card is blocked if all team has character > 4	The selected developer quits, simultaneously reducing morale by 1
Poor version control	A developer did not manage version control well and has to redo work. Card is blocked if all team has character > 4	One developer loses two code cards
Offended coworker	Two team members got into a fight and the better one quit. Card is blocked if all team has character > 2	Most expensive developer quits and -1 morale
Out with a bang	One of your developers just rage quit. Card is blocked if all team has character > 2	One developer left in a huff. Everyone else loses 1 morale.
Infighting	Your developers are arguing with each other Card is blocked if all team has character > 3	All developers paralyzed this turn
Poor documentation	One of your developers was trying to refactor code and could not because the developer could not figure out what the code was doing. Design had to be reworked. Card is blocked if all team has character > 3	Lose two design cards
Poor requirements update	Developers update code based on new requirements, but never updated the specification documents Card is blocked if all team has character > 3	Lose two requirements cards

Poor security discipline	Poor security practices led to a deletion of important files Card is blocked if all team has character > 4	Developer loses three code
Developer not paying attention	The developer was not paying attention and so code written is useless Card is blocked if all team has character > 4	Developer loses 2 code cards
Fake illness	A developer called in sick Card is blocked if all team has character > 5	Developer is paralyzed this turn
Slacking culture	Developers realise they can goof off and get away with it Card is blocked if leader's leadership > 4	-2 all developer skills
Personal problems	A developer has personal issues to attend to today Card is blocked if all team has character > 5	Developer paralyzed this round
Design deviation	A developer chose not to follow the agreed upon design Card is blocked if all team has character > 4	Developer loses a code card
Slight design rigidity	Some inflexibility in your design leads to wasted coding effort Card is blocked if 10 design cards	Developer loses one code card
Coordination mess	Poor design means substantial developer work must be thrown away Card is blocked if at least 3 design cards	Every developer loses 4 code cards
Uncoordinated work	Poor design means some developer work must be thrown away Card is blocked if at least 6 design cards	All developers lose two codes
Overlapping work	Poor design means a small amount of programmer work must be thrown away	All developers lose one code from their stacks

	Card is blocked if at least 9 design cards	
Grave misunderstanding	A programmer misunderstood the work and did the completely wrong thing Card is blocked if at least 3 design cards	One developer loses all their code cards
Missed Requirements	The customer told you things, but the did not show up in design Card is blocked if at least 8 design cards	+2 to total code length required. +2 to code to be inspected.
User infighting	Infighting among users leads to contradictions in design Card is blocked if leader has leadership > 4	Lose 5 design cards
Necessary New Requirements	The customer has new requirements	Lose 2 requirements cards. +75 budget.
Frivolous new requirements	The customer insists on a set of trivial changes Card is blocked if leader has leadership > 3	Lose 2 requirements cards but add 25 to the budget
Unnecessary requirements	The customer has forced through new requirements that could have been negotiated away Card is blocked if leader has leadership > 4	Lose two requirements cards. +50 budget.
Feature creep	The project leader can not get the team to focus on what is important so the project has become bloated with unnecessary features Card is blocked if leader has leadership > 3	Lose 4 design cards
Firefighting	Bad management means all employees are busy doing unrelated work Card is blocked if leader has leadership > 3	All developers are paralyzed this turn
Focus Shift	There has been a change of vision for the project Card is blocked if leader has leadership > 4	Lose 4 specification cards

Lack of clear vision	A developer has gotten frustrated with not understanding what the project is supposed to be about and has quit Card is blocked if leader has leadership > 3	The selected developer quits, simultaneously reducing morale by 1
Misguided instructions	The project leader asked the developers to do the wrong thing Card is blocked if leader has leadership > 4	-1 code from all developers
Policies unenforced	The project leader has failed to enforce policies that were set. This has now come back to bite your project. Card is blocked if leader has leadership > 4	All developers lose 1 code
Poor work layout	The workplace makes it hard for workers to help each other. Card is blocked if leader has leadership > 4	-2 to aid others
Executive override	An executive has made a decision about design against your better judgement. Card is blocked if leader has leadership > 5	Lose one design card
Serious requirements	The customer has forced through new requirements that are important but could have been negotiated to the next project. Card is blocked if leader has leadership > 5	Lose two requirements cards. +50 budget.
Fundamental Error	An issue with requirements has caused an emergency in the project. Card is blocked if at least 7 specification cards	Lose 4 design cards
Wrong platform	The framework, operating system etc. used for the project has been found unsuitable for the project. Card is blocked if at least 4 specification cards	All developers lose two code cards

Wasted effort	One developer spent a week doing the wrong thing because of a misunderstanding about the design. Card is blocked if less than 2 unclear design cards	One developer paralyzed for one round
Code confusion	The confusing design means all your implementers did the wrong thing Card is blocked if less than 5 unclear design cards	One code card is lost from every developer.
Incompatible work	One developer spent the whole week doing the wrong thing Card is blocked if less than 2 unclear design cards	-3 code from one developer stack
Unneeded work	The work is more complicated than it has to be because your requirements are unclear Card is blocked if less than 5 unclear specification cards	All developers lose 1 code card
Incorrect assumptions	Unclear requirements has led to inappropriate code Card is blocked if 0 unclear specification cards	-2 code from all developers.
Confused implementation[2]	Unclear requirements has led to unproductive work Card is blocked if 0 unclear specification cards	A developer loses one code card.
Confused implementation[2]	Unclear design has led to unproductive work Card is blocked if 0 unclear design cards	A developer loses one code card.
Misinformed design	Unclear requirements led to bad design Card is blocked if 0 unclear specification cards	Lose 2 design cards
Fundamentally wrong idea	You completely misunderstood what the customer wanted	Lose all code cards

[2] Both of these cards have the same name and effects but are triggered by different conditions.

	Card is blocked if less than 2 unclear specification cards	
Cost Cutting	Management has decided to cut out extras Card is blocked if remaining budget > 0	Lose all concept cards that cost budget
Financial hardship	Economic bad luck has hit you hard	Lose 25 budget
Hardware failure	Hardware failure has led to project delay Card is blocked if leader has leadership > 4	One programmer loses 1 code card
Illness	One developer is sick	Developer paralyzed this turn
Anxious atmosphere	Anxiety about the budget means a developer has left for another job Card is blocked if remaining budget > 0	The selected developer quits, simultaneously reducing morale by 1
Customer anxiety	Your people spent the whole week calming anxious customers because of the budget overrun Card is blocked if remaining budget > 0	All developers paralyzed this turn
Project leader quits	The project leader knew something was wrong with the project and quit Card is blocked if remaining budget > 0	Project leader quits. Everyone else loses one morale.
Executive Favor	An executive is helping the project and diverted budget to it. Card is blocked if leader's leadership < 6.	+75 budget
Walkthrough	You carefully review requirements and design with the user.	All known unclear requirements and design are replaced as if by someone with skill 5
Well Maintained Library	You have a well documented, easily reusable library of code from previous projects.	One developer gets two code cards

Appendix C: Sample Lesson Plan

The first author employs Project Wars as part of a 2.5-hour once-a-week HyFlex course in IT Project Management. Project Wars is assigned as a homework assignment and used to springboard a class discussion. The actual lesson plan is broken into three parts. The three parts are: (1) pre-assignment briefing, (2) homework, and (3) class discussion.

C.1 Prerequisites

It is assumed the following concepts have been covered prior to the lesson.

- **Project Management Processes (Project Management Institute, 2023):** Students should know that every project begins with some kind of plan and vision. Students should know about concrete manifestations of the plan such as project scope documents or business plans.
- **Waterfall Model (Royce, 1970):** Students should know that in most projects there is a requirements gathering, design, and implementation stage. Furthermore, students should know that the more complete and clear the requirements are the better the design will be. The more clear and complete the design the better implementation will be.
- **Coordination and Integration are Necessary in Large Projects:** Students should know that projects-in-the-large are different from the kind of projects they work on in school (Brooks, 1982). When multiple people work on separate aspects of a project, their deliverables need to be coordinated and integrated so the deliverable is a seamless whole. Just doing the separate parts of a project is insufficient for project completion.
- **Review is a Key Element to Quality Assurance.** Deliverables are not just produced, but they are reviewed and corrected.

C.2 Pre-Assignment Briefing (15 minutes)

Prior to assigning Project Wars, the instructor introduces students to the program and plays a short, incomplete game in front of class. The instructor first shows the URL and the opening screen. The instructor creates a login, emphasizing to students they can use any login name and password except one in active use by someone else. The instructor explains that if they use a login name in active use, the system will deny them access because they won’t know the linked password.

The instructor then shows the hiring screen. The instructor explains that all onboarding is costly, and the game allows the player to choose how much the player will spend on onboarding. Spending more gives the player more flexibility in choosing who works on the project.

The instructor then shows a set of prospective hiring candidates and explains each statistic.

- **Character:** How easy it is to work with the potential candidate.
- **Leadership:** How effective the candidate is in a leadership role.
- **Specification:** How effective the candidate is in creating specification documents.
- **Design:** How effective the candidate is in creating design documents.
- **Implementation:** How effective the candidate is in building the implementation.

The instructor then explains that initial statistics range from 1 to 6, but during the game, these statistics can go up or down. The statistic reflects how good the worker is at that task, with a worker of skill 6 being six times better than one of skill 1.

The instructor hires some candidates who become workers and proceeds to the work screen. The instructor explains that the top 1/3rd of the work screen will have permanent benefits and drawbacks. Also, the player can engage in two core groups of actions, hiring and work assignments.

First, it is possible to hire more candidates but such hiring will be delayed as new candidates will take two weeks to arrive and become workers. Second, each worker can be assigned work. Such work includes:

- **Do Nothing:** The worker is paid to do nothing. The instructor explains that the player can assign workers to do nothing, but sometimes this will be the only thing the worker can do, especially if (for example) the worker has fallen ill.
- **Lead:** The worker is paid to lead. Workers paid to lead will basically do nothing - the idea is to elicit some laughter from the class.
- **Create Specification / Design/ Implementation:** The worker will create specifications/design/implementation. The instructor will explain that at most 10 specification/design documents can be created. Assigning a worker to create more produces nothing. Initially, the project requires the creation of 10 implementation cards. Things can happen in the project so more than 10 implementation cards may be required.
- **Review:** The worker can check and correct the quality of specification/design.
- **Inspect:** The worker can check implementation. The worker can find minor bugs or major bugs. The presence of too many minor or major bugs will cause the player to be unable to win the game.
- **Debug:** If bugs are found during inspection, this allows a worker to correct bugs. Complex bugs are harder to fix than simple bugs.
- **Package:** The worker reconciles differences across implementation to make things ready to ship to the customer.
- **Assist:** A worker who works on implementation created by someone else is less effective.

The instructor then emphasizes there are elements of the unknown unknown (Kvalnes, 2016; Loch et al., 2006; Sutcliffe & Sawyer, 2013). Just because you did review/inspect doesn't mean you will find all the problems with requirements/design/implementation. If you find a problem, you know it exists. However, when you don't find a problem, it doesn't mean the problem doesn't exist.

The instructor then explains that once the player has 10 integrated implementation cards, a new button called "Submit Deliverable" will appear. The instructor hits the "Do Work" button to proceed to the Events screen. The instructor shows the class that this causes the workers to do work and also creates various encounters.

The instructor asks for any questions.

The instructor then assigns the game for homework. The student receives the following instructions:

Individual assignment.

Before beginning this, you must have played Project Wars at least twice. Furthermore:

(1) You must have won Project Wars at least once or

(2) Played Project Wars for at least 30 minutes and played until week 20 on at least one game. Please note the game tracks this by login name.

If none of these conditions are satisfied, you will get a 0 for the assessment.

Put your LOGIN NAME at the beginning of the reflection. This will be used to check the above.

Then:

Identify four things you learned from the simulation. For each learning point:

(1) In bold, state the thing you learned. This learning point should be actionable. In other words, you are able to take concrete steps to address a situation. The typical actionable learning point looks like: If <condition> do this. Or I will always/never do <action> as a project manager.

(2) Elaborate on and explain the bold statement. Keep this short and clear please. You should be able to do this in 3 sentences or less.

(3) Present the negative example from the simulation.

(4) Present the positive example from the simulation. To do 3 and 4, you will have to have gone through the simulation at least twice.

Learning Reflection

Rubric

For each learning point

1 mark for the actionable learning point

1 mark for the elaboration

1 mark for the negative example

1 mark for the positive example

The student also receives a sample reflection as follows:

I learned four things as a result of the computer simulation:

I will always record a police encounter in public spaces. Getting a record of what actually happened is important because the police can lie in court. In the first simulation run, I turned off my phone recording because I was asked to do so by the police. Later, I found out the policeperson lied in court about what actually happened, there was no proof and the judge sided with the police officer. In the second simulation run, I stated, "I am not complying with your request officer. The First Amendment guarantees my right to record in public spaces. Are you ordering me to cease recording?" If the officer orders you to cease recording that is an unlawful order and you then have recourse to sue the police.

When the police officer asks me questions, I will only tell the police officer the things required by law. The reason you never tell the police anything more is because anything can and will be used against you in a court of law. You need to only tell the police officer your name, address, and date of birth. If you are walking, you do not need to show ID. In the first run of the simulation, I truthfully told the police officer I was on the street to meet a friend. Because I accidentally had an inconsistency in my story, the police officer arrested me for lying (perjury). He then searched me, found the laundry detergent stain in my pocket, and arrested me for cocaine possession. In the second and third run, I just told the police officer, "Officer, I understand you are trying to do your job, but I refuse to answer any questions." Because the police officer had no probable cause, he could not do anything else.

I will always assert my right to silence and then tell the police officer nothing. The right to remain silent is a **civil right** and **cannot be used to demonstrate guilt**. In some jurisdictions, you must state you are asserting this right so it is important to not just keep silent but state you are doing so in accordance with your rights. In the second run of the simulation, when I told the police officer I was asserting my right to silence, the police officer told me that because I was asserting my right to silence, that could be considered evidence of guilt. I believed him and then was charged with perjury, and cocaine possession. In the third run, I told him that asserting your right to silence is legally protected under the fifth amendment to the Constitution, and he was forced to let me go because he did not have probable cause.

If asked, always refuse consent to a search. A police officer can only search you if you give permission or if they have probable cause. Because nothing that happens can work in your favor, it is important to politely refuse any police request, especially a request to a search. If the policeperson insists on a search or arrests you, only state your rights verbally. Do not resist the policeperson. In the simulation, the police officer says he has to pat you down for weapons and search your pockets. In the first and second run, I allowed him to do this, and he found a laundry detergent stain in my pocket. He then arrested me for possession of cocaine. In the third run, I told him, "Officer, I refuse all requests to a search. You do not have probable cause." Because he did not search, he did not find the laundry detergent stain, and I was allowed to go free.

If you are wondering what this is based on, watch this: https://www.youtube.com/watch?v=d-7o9xYp7eE

C.3 Homework

When students submit homework, the instructor should first check that students have played the game. The instructor should check two files:

- https://cecilchua.online/stressgame/logs/entryexit.log and
- https://cecilchua.online/stressgame/logs/<login name>20.log

Note, the second file will not exist if the student abandoned or won the game.

The first file records the time the student logged in and the time the student either clicked the abandon game button or wins the game.

The game also logs the actions the student took each turn. This second log is deleted when the student abandons or wins the game. Note, summary information on play is recorded in the entryexit.log file, so even though details of the student's play are lost, the summary of the student's play remains documented.

These checks are necessary, because there is a certain class of student who will fake doing an assignment regardless of how engaging the assignment is.

The instructor should then read the student reflections and copy things the student has written into a word processing/presentation document. This document should be divided into at least 7 categories. Each category should be on a different page. Reflection elements can go into more than one category.

- Deferral of Gratification - see Sections 2.1 and 4.1.
- Mindfulness - see Sections 2.1 and 4.1.
- Discipline - see Sections 2.1 and 4.1.
- Management and Control of a Project Team - the role of self and input control is a topic the game allows a discussion on - see Section 4.2.
- Project Leadership - see Section 4.3
- The relationship between productivity, slack, overtime and related concepts - see Section 4.4.
- Coordination Roles and Technology - see Section 4.5.

The instructor should not label the categories in the word processed/presentation document but keep a mental note of what each category is about. The idea is to have the concept emerge from the student discussion.

The instructor should then mark the reflection and provide feedback. Feedback should include "right" answers (i.e., explaining to students why they encountered the situation they did). For example, a student who writes about how their workers kept leaving should receive feedback about turnover contagion. It should be explained that hiring a worker with poor characteristic or treating workers badly (forced overtime) doesn't just impact individual workers but the whole team.

C.4 Class Discussion

At the beginning of class, the class is recorded and the word processing/presentation document is projected. The class discussion proceeds in a loop where each cycle of the loop relates to one category.

At the beginning of the loop, a student is invited to tell a story related to the projected quotes. Another student is invited to tell a similar story. The instructor then invites students to tell us what the central learning point is. Allow back and forth discussion about the issue. At the end, the instructor gives a mini lecture on the category topic. For example, for the mindfulness topic, the instructor identifies the categories of organizational mindfulness (see Section 2.1) and explains how they work. For the leadership topic, a short lecture is given about why leaders should not be doing production work.

The loop is repeated for each category until about 10 minutes before the end of class. At this point, the categories from the game and their relationship to future lessons the instructor will teach are displayed and discussed.

The standard recommendation in HyFlex classes is that in-class discussion should be supplemented by a discussion board for students who attend the course asynchronously (Beatty, 2019). However, the first author, like many other instructors and students, didn't find discussion boards to be useful pedagogically

(https://www.reddit.com/r/CollegeRant/comments/116lcoq/discussion_boards_suck/,
https://www.reddit.com/r/Professors/comments/of755q/discussion_boards_are_mostly_useless/).

Instead, the first author gives direct feedback to the reflection writeup that goes beyond the correctness of the reflection to discussing the key concepts associated with each reflection point (see C.3 above).

About the Authors

Cecil Eng Huang Chua is an associate professor at the Missouri University of Science & Technology. Cecil has published in 10 of the IS Senior Scholar basket of 11, having multiple publications in several, including in *MIS Quarterly.* He is an editor for the following journals: *AIS Transactions on Human-Computer Interaction, Communications of the AIS, European Journal of Information Systems, Pacific Asia Journal of the Association for Information Systems, Information Systems Journal, Project Management Journal.* Cecil has consulted for a range of organizations including Daimler SEAsia, General Motors Singapore, the Singapore Ministry of Defense, and Fonterra.

Veda C. Storey is a Distinguished University Professor, Tull Professor of Computer Information Systems, and Professor of Computer Science at Georgia State University. Her research interests are in intelligent information systems, data management, conceptual modelling, and design science research. She is particularly interested in the assessment of the impact of new technologies on business and society from a data management perspective. Dr. Storey is a member of AIS College of Senior Scholars and a member of the steering committee of the International Conference on Conceptual Modelling and the Workshop on Information Technologies and Systems. She is a recipient of the Peter P. Chen Award, an ER Fellow, an AIS Fellow, and an INFORMS Fellow.

Linda Wallace is a Professor and the Konrad W. Kubin Senior Faculty Fellow in the Department of Accounting and Information Systems at Virginia Tech. She obtained her Ph.D. in Computer Information Systems from Georgia State University in 1999. Her research typically involves empirical studies in areas such as online communities, fitness technologies, software project risk, and crowdfunding. Her research has been accepted for publication in *MIS Quarterly*, *Journal of Management Information Systems*, *Decision Sciences*, *Information Systems Journal*, *Communications of the ACM*, *Information & Management*, *IEEE Security & Privacy*, *Decision Support Systems*, and others. She has served as Associate Editor for Information Systems Journal, Decision Sciences, and Information & Management.