

If you build it, will they come? Designing an instructional computer game for an undergraduate computer course.

William R. Watson
Indiana University–Purdue University Indianapolis

Introduction

This presentation will detail the process and experience of designing *Lifecycle*, an instructional, web-based computer game for my undergraduate Systems Analysis and Design course. I will briefly present the game, discuss the process of designing the game, present the underlying rules behind the game, present initial feedback from my students, and share lessons learned. This presentation should be of interest to anyone considering the design, development, or application of video or computer games to education.

Why a computer game?

Video and computer games have rapidly become a leading form of entertainment in America and across the world. Huge video game sporting conventions have sprung up in California and Seoul, among other cities, and players regularly compete with each other in local “LAN parties.” The NPD group reported that computer and video game software sales exceeded a record \$10.3 billion in the US in 2002, slowing but remaining strong with \$10 billion in 2003 (The NPD Group Reports Annual 2003 U.S. Video Game Industry Driven by Console Software Sales, 2004). This compared to \$9.5 billion in box office sales for the US movie industry in 2002 (U.S. Entertainment Industry: 2002 MPA Market Statistics).

With the emergence of video and computer games as popular entertainment, it is only natural that educators would turn to this new media to examine its potential for use as instruction. More and more researchers have begun to examine video and computer games and many have found that the media holds great promise for engaging instruction (Reigeluth, 1999a; Prensky, 2001; Molenda, 2002; Gee, 2003; Squire, 2003; Aldrich, 2004). More and more instructors are seeing that their students are demanding more engaging instruction, and video games seem like a strong candidate since the students are clearly playing them. Jenkins reports that while 88 percent of surveyed incoming MIT students had played games before the age of 10, more than 75 percent were still playing games at least once a month.

However, is it truly realistic for an instructor to design an educational computer or video game for his class? I decided to test this for myself and began designing a game with the hopes that it not only would be more engaging for the students but that it might actually improve their understanding of the relationship between SAD tools and concepts that I teach in my class. While I am continuing to refine the game before testing the impact on the students’ understanding, I found that it is definitely possible to design an instructional game and have it developed. A senior in computer graphics at my university took on the development of the game as a senior project, and it was entirely designed and completed in under a year.

The game

Lifecycle is an interactive simulation game which seeks to develop an understanding of Systems Analysis and Design (SAD) concepts using the UML methodology. Players will play the role of a systems analyst in charge of a development team. The development team is seeking to develop a quality product for the Client Corporation using very various UML tools to document the system’s requirements and design specifications. Players will be scored based on the quality of the system, time spent developing the product, and user satisfaction. Other variables will be stored which will impact the scoring of these key variables. The goal of the game is to develop the highest quality system, in the least amount of time, with the highest customer satisfaction. The game will produce a log of the player’s name, player actions throughout the game, and the player’s final score.

I designed the game so that my students would be able to discover underlying UML and SAD concepts as well as the relationship between these concepts and the hands-on tools they use to document system requirements through experimentation, feedback, and yes, “play.” The focus of the game is on instruction and simulating the long process of analyzing, designing, and developing a system. I envisioned the game scaffolding knowledge presented previously in class. Therefore, the students would receive instruction through lecture and utilize specific UML diagrams in class projects. *Lifecycle* is then to be

played following this initial introduction to the material so that the students can experiment as analysts and receive instant feedback on their choices. UML focuses on a lifecycle of iterations, meaning diagrams are created and revised multiple times. The overall lifecycle of the class project and indeed most projects is often too long for the students to be able to easily make the conceptual connections that I hope they will make. Lifecycle hypothetically will give them the opportunity to repeatedly go through a complete system development process in a short amount of time and have a unique experience each time, while receiving important feedback that will allow them to make stronger conceptual connections while also having factual knowledge, such as UML and SAD terminology, reinforced. While the game design was a daunting task, I believe that this experience has shown that it is possible for an individual instructor to design and have developed, an instructional game that students will enjoy and find meaningful.

The design process

I had several goals in mind when approaching the task of designing Lifecycle. I wanted the game to be playable via the Web so that my distance students would also be able to use the game. I also wanted the game design to be playable in paper format so that I could test the underlying structure of the game for accuracy prior to having the game actually built. I wanted the game content to reinforce specific conceptual and factual instructional objectives.

I also had several restrictions that would place limits on the game's design. As I would have to rely on a single student to develop the game, it would need to be simple enough that this was possible. I also would only have the student's help for one semester, so the design's complexity would also need to be simple enough to be completely built in a single semester. Because of this, Lifecycle is primarily a text-based game. However, it does utilize images, and I am currently working with a second student to incorporate more audio and refine the visual design of the game.

After recognizing both my primary objectives and the limitations that I was working under, I set about trying to figure out how I would create an underlying structure of rules that would cause the game to behave as needed and reinforce the instructional objectives that I had decided on. I had never designed a game before, so I began by reading a number of books on game design. By far the most helpful resources I had were Salen and Zimmerman's *Rules of Play* (2003) and an interview with Dr. Michael Molenda of Indiana University (personal communication, February 25, 2005). *Rules of Play* is very helpful to the novice game designer because it breaks down the structural design (rules) of a game (usually a board game) at the end of each chapter. This helps you see how game designers tie contextual and conceptual objectives to game rules that will reflect these objectives. Dr. Molenda had designed an instructional board game several decades ago that tried to simulate the process of managing the introduction of change in a school system. He went over the game and how the dice the players' rolled determined game response based on a chart of percentages which was tied to game feedback and action outcomes. This game has since been translated into a Web-based game by Dr. Ted Frick and a group of his graduate students (2005); although, access is currently limited to those on the IU system.

After reading *Rules of Play*, talking with Dr. Molenda, and playing the Web-based Diffusion game, I had a stronger understanding of how to structure my game so that it could incorporate random occurrences, create variable player experiences, and still promote the learning objectives that I was interested in. I first developed the context of the game, the design situation and the system stakeholders. I then determined how I would score the game and translated this into scoring variables which would track player performance. Next, I created a list of player actions, some immediately available, others only available after certain conditions had been met and decided how these actions would impact the scoring variables as well as reflect the learning objectives I had in mind. Finally, I created a list of random impacts and tied many of these to player actions and learning objectives, so that even though the impacts were random, they reinforced learning objectives and could be somewhat controlled by player action. I detailed feedback for all player actions and random impacts, play-tested the design and wrote up a design document which was given to the student developer. I met each with the developer to review the process of the game and help her to understand the underlying rules that shaped and supported the game and its use as scaffolding for learning.

The game structure

Variables

As examining the underlying structure of other games was most helpful to me in designing Lifecycle, I will try to briefly present Lifecycle's structure and how it reflects my instructional goals for the

game. I will do this by describing the game's variables, player actions, and variable impacts and how they are related to each other and my instructional objectives.

The scoring and indeed much of the game itself is based on five variables: time, business knowledge, quality, client satisfaction, and project status. The player's final score is determined at the conclusion of the game (when time has run out or the project has been completed) by adding the player's scores from the time, quality, and client satisfaction variables. A player has a maximum of 50 weeks to complete the project (achieve 100% in project status variable). The time variable measures how many weeks (turns) have been used. This variable is impacted by player actions (which use up one or more week, depending on the action), as well as random impacts which can add to or reduce the number of remaining weeks the player has. At any time, the player can look at the screen and see how many weeks he has left. The pressure of this time limit helps reflect to the students the actual pressures that come on a systems analyst in a real-world project. Furthermore, the pressure is there to try and help demonstrate some of the conceptual issues I cover in class, including how the developer's natural inclination is that taking time up front to document the analysis and design of a system before building it is a waste of time, while, in reality, jumping into building a system without establishing what the system needs to do and how these objectives can be best met can result in longer development times. Therefore, players are trying to complete the project in the shortest amount of time, but also with the highest quality and client satisfaction.

Business knowledge is a percentage rating that identifies the analyst's understanding of the system and the business problem it is trying to solve. It is heavily influenced by how much the player has involved stakeholders in the development process. This supports the user-centered design process which I teach in class. This score is never revealed to the player, but is used to measure the impact of various activities throughout the process. It has a direct impact on the success or failure of the different activities. The state of the business knowledge is alluded to through conversations with stakeholders but never quantifiably defined to the player.

The quality variable is related to the quality of the system that is being developed by the player-analyst. This variable is impacted by player actions and is also heavily impacted by the business knowledge variable. This reflects the concept that if an analyst does not truly understand the business logic behind the system, then quality will likely suffer. The quality score can only be determined through the user testing action, which will reveal what level of quality the currently developed aspects of the system are showing.

Client satisfaction is a very important variable in terms of player score. This reflects the concept we discuss in class that clients must be involved or they will have no investment in the resulting system, even if it is a good one. Client satisfaction is therefore impacted by the level of stakeholder involvement in addition to the quality of the system that is developed. The level of client satisfaction is only revealed to the player by talking with stakeholders or conducting user testing on the system.

The final variable, project status, simply shows what percentage of the project has been completed. This variable is only impacted by the player action implement system. To complete the game, the player must have finished implementing (building) the system (100% score) within the 50 week time limit. The concept that an analyst can build a system more quickly if he has a true understanding of what the system's requirements are is reflected by the variable impact and time taken by the implement system action. The higher the business knowledge score, the higher the gain in project status when implementing the system, and the less amount of time used up by the action. So, if the player tries to implement the system at the very beginning of the game without taking the time to understand what must be built before trying to build it, the action takes a long time in terms of weeks and results in little gain in terms of completing the project. Furthermore, the low business knowledge rating in this instance would also result in poor quality. This reflects how these variables tie together to support the underlying concepts which I want the game to reinforce.

Player actions

Player actions are the actions the player can take each turn of the game. Initially, there are certain actions which the player can take and additional actions are revealed to the player when certain conditions are met, such as talking with specific stakeholders or completing other actions. The following outline shows the player actions available in the game and which ones are initially available as opposed to which are "unlocked" by other actions in the game.

- A. Initially Available:
 - a. Talk to Stakeholders
 - b. Generate/Refine Requirements Specification Document

- c. Brush up technical skills
- d. Take Ms. Manner's Composition and Elocution Course
- e. Schedule Team Meeting
- f. Implement System
- B. Unlocked by Stakeholders:
 - a. Develop Change Management Plan
- C. Unlocked by Actions
 - a. Conduct Feasibility Analysis: Unlocked by Generating Requirements Specification Document.
 - b. Develop/Refine Use Cases
 - c. Develop/Refine Class Diagrams
 - d. Develop/Refine Activity Diagrams
 - e. Develop/Refine Sequence or Communication Diagrams
 - f. Iterate Documents
- D. Unlocked by Actions AND Stakeholders
 - a. Conduct User Test

The actions take a different number of weeks to complete, with most taking one week, some more, and one- implement system taking a variable amount of time based on the player's current business knowledge. Each action results in different impacts on the variables discussed above. For example, if a player were to talk to a stakeholder as an action, it would use one of the fifty weeks available to complete the project. Depending on which stakeholder the player tries to talk to, the action could unlock additional actions, unlock additional stakeholders, positively impact business knowledge and client satisfaction, and so forth. If the player tries to talk to certain stakeholders without first talking to the "boss" and getting management support, he might waste a week trying to talk with the stakeholder without getting any response. This is an example of another concept we cover in class, the need for management buy-in. As another example, talking with the secretary could also reveal the general client satisfaction level around the office.

Other actions are specifically tied to diagrams taught and used by the students in class. The use cases, class, activity, sequence and communication diagrams are all diagrams the students complete in class. The students are able to have the sequence of these diagrams, their names, and what they are used for reinforced by playing the game. Each of these actions help to enhance the analyst's business knowledge score, and once all of the documents have been completed once, the player is able to iterate the documents, which allows for them all to be refined at a lower cost in terms of time. This reinforces the concept of an iterative system lifecycle, as taught in class.

Finally, there are some actions tied to additional concepts covered by the random impacts built into the game. I talk with the students in class about what skills are important for a systems analyst to have. These include strong communication, technical, and teamwork skills. These concepts are reflected by negative random impacts that can occur if the player does not have the foresight to strengthen each of these areas by using the appropriate action. This can be further described by looking at the random impacts in the game.

Random impacts

Random impacts are used in the game to both add variability and increase the value of replaying the game, as well as to reinforce additional concepts, such as the importance of various skills to a systems analyst. The random impacts in the game are as follows:

1. Technical miscommunication
2. Foot-in-mouth
3. Poorly written communication
4. Super programmer
5. Programming mishap
6. Change request
7. Team argument
8. Stakeholder unavailable
9. Team member sick

Each turn there is a percentage chance that a random impact will occur. Most of the impacts are negative and can cause the player to lose turns, such as if there is an argument on the team. The chance of these impacts can be reduced by certain actions. For example, the more team meetings are scheduled, the

smaller the chance that if a random impact occurs, and that impact is a team argument, that the negative impact will actually occur. Instead, the player will be given feedback that identifies how the team was able to handle a potential argument well and kept working seamlessly. This reinforces the importance of teamwork skills. Other impacts are similar, such as the change request. If the player has developed a change management document, then weeks can be lost adding additional functionality requested by the client. The importance of managing change requests is another concept we discuss in class. One of the impacts is positive, the super programmer. If the player is implementing that turn and this random impact occurs, one of the analyst's team members turns in a great programming performance, and completes the implementation in much less time than normal. These impacts are meant to add to the game's fun and unpredictable nature and make each replay different than the previous while also reinforcing additional concepts covered in class.

Lessons learned

My goals for this project were to create a fun, instructional game which could be developed by one undergraduate student in one semester. I believe I have succeeded in this and have also learned a number of lessons along the way. While I was able to have the game completed in one semester, the student did not manage to get out all of the bugs, and despite pledging to tighten them up over the summer, promptly disappeared and stopped answering emails, so a large lesson coming out of this was to not turn in that final grade until the game was truly finalized. While I can look back on this with a bit of a chuckle now, I was quite panicked at the time because it was very important that the game accurately followed the underlying rules which I had defined and the student's final version did not. I have been able to locate a second student now who is removing these bugs and adding additional multimedia elements, such as sound, to the game, so I believe my delay in introducing the game to my class will be worth the wait, as the new version should be more entertaining and require less reading on the player's part.

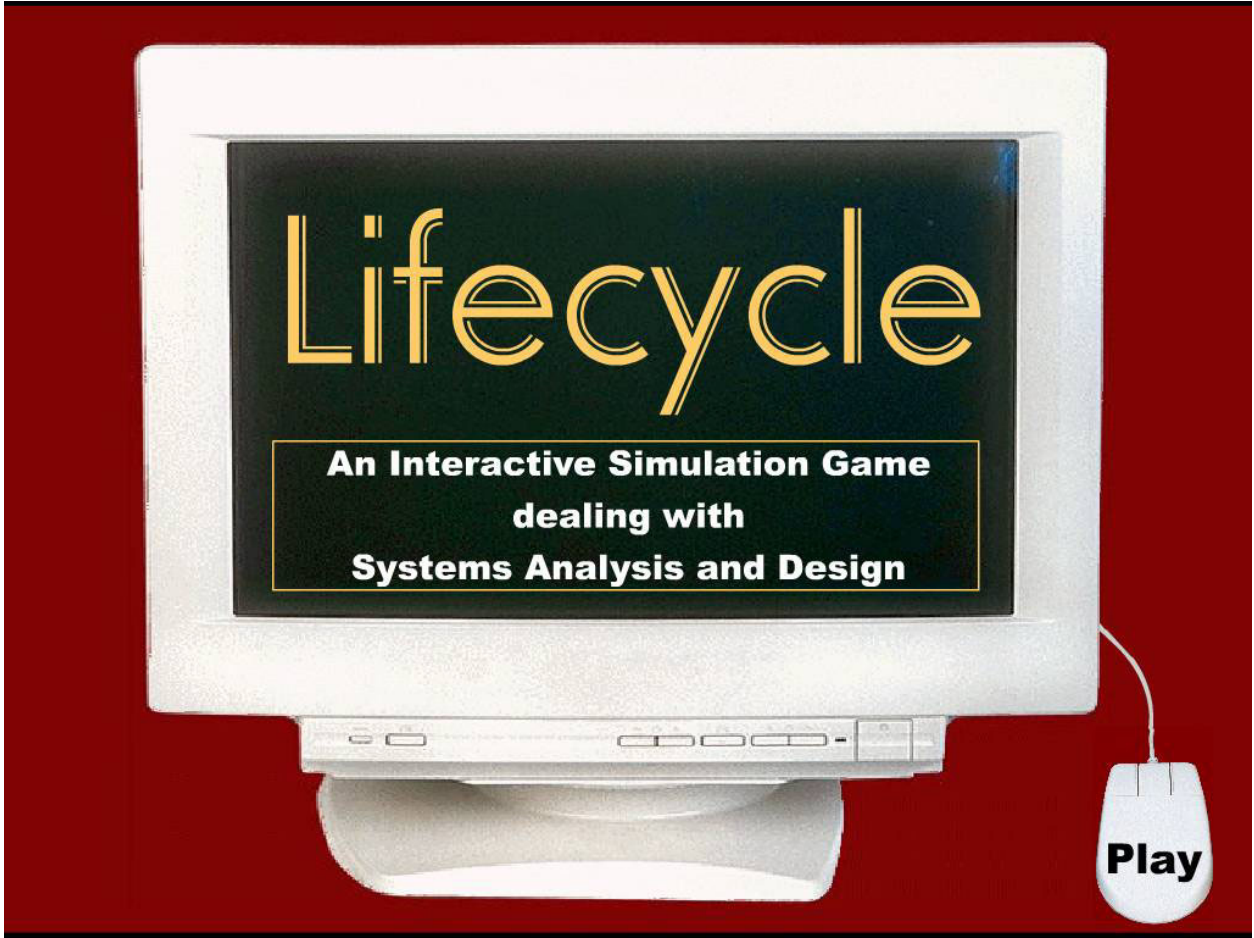
Another important lesson which I learned in this process is how important it is to come to an understanding of what the underlying rule structure of your game will be and how this reflects your instructional goals. It was very important for me to design the game and be able to test it on paper before turning it over to a developer. This allowed me to test the logic that I had developed and ensure that it was sound.

Ultimately, while the process of developing Lifecycle was very valuable to me, I have not yet determined if the game is a true success because I have not yet turned it over to my students. Once the game has been revised, and I reach the point in the semester where my students have been introduced to all of the UML documents covered in the game, I hope to be able to test its impact on students who play it. I envision this game being incorporated into the class as a form of scaffolding. I believe it will be very important to debrief the students and talk about lessons learned from their experience. Students must be asked to analyze their play and what they learned from the game or it will likely remain nothing more than a game to them. I think this analysis both in a group discussion and possibly through a journal assignment, would be very important to successfully using an instructional game like Lifecycle. In conclusion, this was a challenging but valuable experience which I hope will prove valuable to others. Much research remains before I can call Lifecycle a success, but the first steps have been taken, and it is my hope that this will help others to begin their own journey into designing instructional games.

References

- Aldrich, C. (2004). *Simulations and the future of learning: an innovative (and perhaps revolutionary) approach to e-learning*. San Francisco, CA: Pfeiffer.
- Frick, T. (2005). *Diffusion Simulation Game*. Retrieved February 25, 2005, from <http://www.indiana.edu/~istdemo/dsg/login.phtml>
- Gee, J.P. (2003). *What games and have to teach us about learning and literacy*. New York, NY: Palgrave Macmillan.
- Jenkins, H. (2002, March 29, 2002). *Game Theory: How should we teach kids Newtonian physics? Simple. Play computer games*. Technology Review.
- Molenda, M. (2002). *A New Typology of Instructional Methods (updated February, 2005)*. Paper presented at the Annual Conference on Distance Teaching and Learning, Madison, WI.
- The NPD Group Reports Annual 2003 U.S. Video Game Industry Driven by Console Software Sales. (2004). Retrieved February 21, 2004, from http://home.businesswire.com/portal/site/google/index.jsp?ndmViewId=news_view&newsId=20040126005198&newsLang=en

- Prensky, M. (2001). *Digital game-based learning*. New York, NY: McGraw-Hill
- Reigeluth, C. M. (1999a). What is Instructional-design Theory? In C. M. Reigeluth (Ed.), *Instructional-design theories and models: A new paradigm of instructional theory*. Mahwah, NJ: Lawrence Erlbaum Associates.
- Salen, K., & Zimmerman, E. (2003). *Rules of play: game design fundamentals*. Cambridge, MA. The MIT Press.
- Squire, K. (2003). Video games in education. *International Journal of Intelligent Simulations and Gaming* (2) 1.





Lifecycle

Choose the Activity you wish to perform

Rules

Game

Your project is 0%

Weeks used: 0

STAKEHOLDERS



Mortimer Bossman
Profile



Sylvia A. Sisstant
Profile



Ed User
Profile



Janet Hardware
Profile

ACTIVITIES

- | | |
|--------------------------|---|
| <input type="checkbox"/> | Talk to Shareholders |
| <input type="checkbox"/> | Generate/Refine Requirements Specification Document |
| <input type="checkbox"/> | Brush up Technical Skills |
| <input type="checkbox"/> | Take Ms. Manner's Composition and Ellocution Course |
| <input type="checkbox"/> | Schedule Team Meeting |
| <input type="checkbox"/> | Implement System |
| <input type="checkbox"/> | Develop Change Management Plan |
| <input type="checkbox"/> | Conduct Feasibility Analysis |
| <input type="checkbox"/> | Develop/Refine Use Cases |
| <input type="checkbox"/> | Develop/Refine Class Diagrams |
| <input type="checkbox"/> | Develop/Refine Activity Diagrams |
| <input type="checkbox"/> | Develop/Refine Sequence or Communication Diagrams |
| <input type="checkbox"/> | Conduct User Test |
| <input type="checkbox"/> | Iterate Documents |