Teaching Tip
Using Rapid Game Prototyping for Exploring Requirements Discovery and Modeling

Nikunj Dalal
Department of Management Science and Information Systems
Spears School of Business
Oklahoma State University
Stillwater, OK 74078
nik@okstate.edu

ABSTRACT

We describe the use of rapid game prototyping as a pedagogic technique to experientially explore and learn requirements discovery, modeling, and specification in systems analysis and design courses. Students have a natural interest in gaming that transcends age, gender, and background. Rapid digital game creation is used to build computer games quickly and easily using game creation software that requires little or no programming knowledge. It enables the developer to build a quick prototype game and to view the effects of changes almost immediately. The process can be seen as learning-by-making and learning-by-design, which are grounded in the learning theories of social constructivism and constructionism. Viewing the game as a system, students worked on an assignment to create a game prototype for a user, determine requirements, and specify the requirements using use-case diagrams and UML class diagrams. Students found this approach innovative, instructive, and entertaining. We describe this approach and outline its benefits.

Keywords: Systems analysis and design, Systems thinking, Modeling, Requirements analysis & specification, Teaching tip, Game theory

1. INTRODUCTION

Teaching modeling and requirements analysis in an experiential manner in a systems analysis and design course has its challenges. A typical systems analysis and design course in the IS curriculum involves among other content the study of methodologies of systems development and the modeling of data, processes, and objects. A common approach to providing students with practice in developing prototype systems in a real-world context is by means of group projects in which student teams work with a real client with live users to determine requirements and build a system. But students at that stage have typically not interacted with real users and are not familiar with requirements discovery and analysis in a practical setting. Moreover, it takes time to find a real client and the analysis is carried out in a group context. In other variations of pedagogy, the systems development project may not be in the context of a live organization but instead may be based on a pre-written case study, in which case the students may not interact with live users at all. In any case, it would be helpful if the student had practice in individual requirements analysis and modeling in a fun, safe, and experiential environment with another human before they actually talk to a real client.

We have been using for the past several years, an innovative approach that uses open-source or public-domain software for rapid digital game creation (RDGC) to help students create rapid game prototypes. During the process, the individual student interacts with a human user to create a rapid game prototype. The student also creates models to represent the "user requirements" and the "game system."

2. RAPID DIGITAL GAME CREATION

Rapid game creation enables a creator to build a quick prototype game and to see the effects of changes almost immediately (Dalal, et. al, 2009). There are various RDGC tools available such as Game Maker (http://www.yoyogames.com/), Multimedia Fusion (http://www.clickteam.com/website/usa/), Alice (http://www.alice.org/), and Scratch (http://scratch.mit.edu/). Many of these tools are available in the public domain.

A large body of research from multiple fields demonstrates the power of digital games in learning and a growing number of researchers are incorporating games into education. Digital games are linked to fun, excitement, motivation, creativity, learning, and flow. Not only has game-playing been found to be useful, recent research has shown that game-making as well can be fun and instructive

with the use of high-level game development software (Overmars, 2004). Many studies that have looked at game-making in teaching have reported positive results such as Cooper, et. al. (2003) and others cited in Dalal (2009). The process of RDGC can be seen as learning-by-making and learning-by-design, which are grounded in the learning theories of social constructivism (Solomon, 1994) and constructionism (Harel and Papert, 1991).

3. TEACHING APPROACH

3.1 Assignment

Several years ago, we developed an RDGC assignment for graduate students enrolled in a required systems analysis and design course in the MIS curriculum. (These students typically had computing backgrounds.) The assignment was refined in a subsequent semester and has been continually used with positive results. The assignment reads as follows:

The purpose of this assignment is to learn about use case modeling and requirements analysis by acting as a systems analyst on a micro-scale with a live user (and in a fun way!) to develop a "game system" prototype. For this part of the assignment, your "user" can be a friend, a neighbor, a parent or child or anyone you know who is willing to participate. You will be using a high-level game development tool for this purpose but as our focus is not on the tool; you need only learn enough about the tool to quickly build a prototype game. Learning to use the tool can take as little as an hour.

Tasks:

1. Create a quick-and-dirty prototype of a very simple basic game of your own choice using simple game creation software such free or demo versions of the Games Factory 2 or Game Maker or Multimedia Fusion 2 or any other software you are familiar with. For the purpose of this assignment, it does not matter what type of tool you use or what type of game you create. (Retain your game as it may be needed for future assignments).

2. Demonstrate your game to your "user", let him/her play with it, and help her help identify/discover requirements. The requirements here would be features they would like to see in a sophisticated game based on the same premise. Identify as many requirements as you can.

3. Develop and submit an initial use case diagram of the requirements. Include high-level narratives for at least three use cases.

4. Describe your experience with this process including your learning about requirements discovery by interacting with the user and any challenges you faced in this process.

5. Submit the use case diagram, a screen shot of the game you developed, and your experience report.

3.2 Sample Deliverables

This section shows some examples of actual models created by the students. The samples include a screenshot of a game, an initial use-case diagram, and a use-case narrative. The diagrams shown here are meant as illustrations and do not necessarily represent ideal, complete or even fully accurate models. Figure 1 show part of a screen shot of a prototype game created by a student using GameMaker. Figure 2 shows the initial use-case diagram representing user requirements, which the student created after interacting with the user. Table 1 is a narrative of one of the use cases.

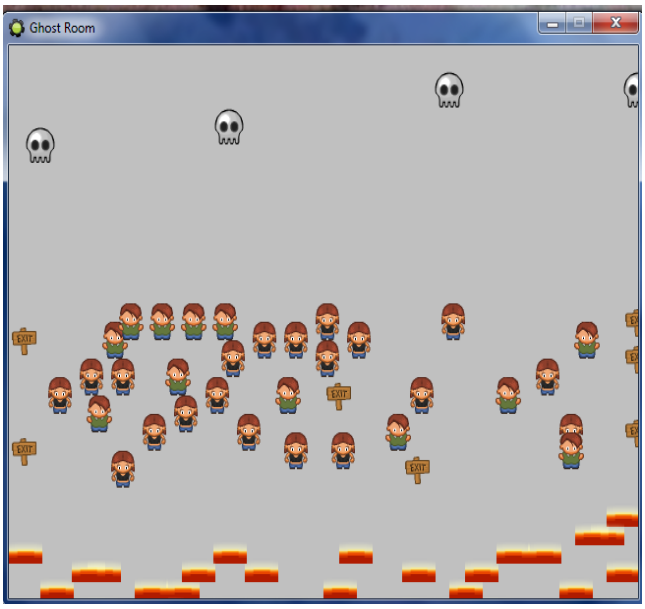

Figure 1: Screen shot of a prototype game

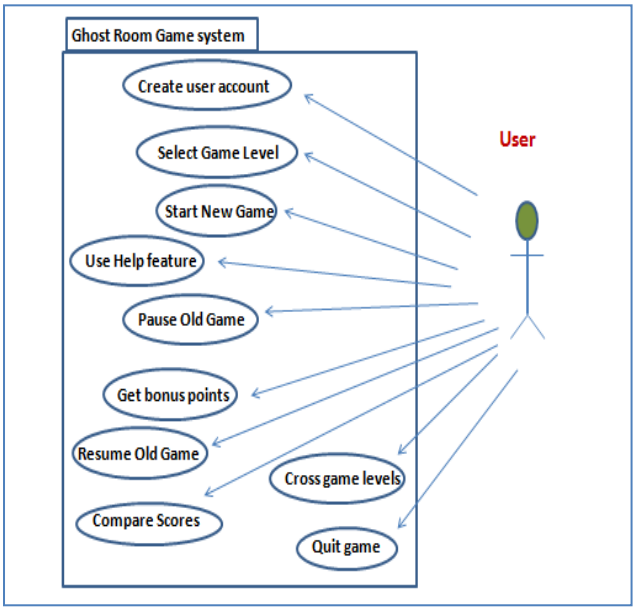

Figure 2: Initial Use-Case Model

Use-Case Name:	Select game level	Use-Case Type Business Requirements: yes
Use-Case ID:	GR-BUC0224	
Priority:	High	
Source:	Requirement-GUT-A2.00	
Primary Business Actor:	User (or Game player)	
Other Participating Actors:	• Game designer: The one who designed and built the game	
Other Interested Stakeholders:	• Game owner: Interested in making the game more user friendly and more interesting	
Description:	The use case describes the event where a user can choose between different levels of difficulty of the game. A new user can choose to play at the "Beginner" stage where the game will be at a basic level in which the ghosts move slowly. When a user is ready, he/she might choose to go for slightly harder version of the game at the "Amateur" level. At this level, the speed of the in-coming ghosts can be slightly increased. For experienced players, they may choose to try a more challenging level, "Expert" which is the toughest level offered. Here, not only is the incoming speed of ghosts increased but also the number of ghost coming at a particular time is doubled. The user can choose to start at any level and after he/she crosses the threshold score, he can be taken to the next level automatically.	

Table 1: A use-case narrative

3.2 Benefits

A large majority of students have reported this process to be highly interesting, relevant, and instructive. Building a game, letting a user play the game, observing the user, interacting and interviewing the user, and modeling the requirements, all together was seen as a "great" learning experience. Several students identified an initial challenge of unfamiliarity with game development but they quickly overcame it as the software was easy to learn and use.

Based on experience reports and preliminary data collected, we can infer that students acquired experiential learning in the following areas: Object-oriented concepts such as objects, events, and behaviors; practice in developing use-case diagrams and other models based on live scenarios; better appreciation of model-based requirements engineering; better understanding of users' needs and the role of the analyst; and clearer understanding of the uses of rapid prototyping. Moreover, the students reported a lot of fun in the process.

Student comments reflect their improved understanding from this experience:

"After devising the basic game and then integrating the results of the interview into the use-case diagrams, I feel as if I have a better map for the next version. I was actually somewhat surprised at how effective this exercise was in giving shape and structure to the project."

"Being able to experience something, you get a better idea of what is actually missing than just trying to come up with these requirements on your own. I found that she was able to pick out very useful requirements with only a few minutes playing the game."

"This was an interesting experience. Determining what the user wanted along the lines of functionality was easier than I thought it would be. For example, the user wanted to have a timer to limit the current player's time to make a play. This requirement was simple to understand and there was no room for interpretation. Another requirement of the user was to keep track of the running score of multiple games. This requirement was a little harder to determine what I should do from a design standpoint. This was because the user did not care if the data was stored for future games or if it was lost when the application was closed. I feel that by using rapid prototyping techniques, analyst can achieve better communication with the user and determine the needs quicker. It is also helpful to the user to see a prototype of the system in order to know what he or she wants it to do. "

"I realized how important (is) the interaction with the user while developing any system…I realized that I had ignored various important aspects of the game because I was developing the game only by considering my point of view."

"After this exercise, I have learned that some requirements are easily understood by the systems analyst; whereas, others may be vague or the user may not know exactly what he or she wants. I can see that, in a real organization, this method of development can be very advantageous. It allows the users to see the system and allows the designers to make changes before it is too late."

"From this I realized how important user interaction is, as whatever feedback that we get from the user will help us in building our system in a better way. Sometimes these interactions help in thinking out of box and hence in order to satisfy the user requirements we can come across new options that we would not have explored before."

4. DISCUSSION

We have tried this approach with both graduate and undergraduate students and have seen good results. The ideas described in this paper are platform independent as several open-source and/or public domain tools can be used for pedagogical delivery. Students are able to individually

explore prototyping, requirements analysis, and modeling in a fun environment with "users" who are typically family members or friends, before they actually talk to a real client. As the focus of this one-shot exercise is on interactions with users, we believe the sophistication or quality of the game is not important and the student's work should not be evaluated on this basis. (There are other assessment opportunities such as the semester-long systems project for evaluating system quality.)

We have found that it makes sense to ask students to start the assignment in the fourth week of a 16-week semester, after basic systems development concepts, model-based systems analysis and design, prototyping, and use-case modeling concepts have been taught in class. Sufficient time should be given to students to complete their work. Students may be shown the parallels between a digital game and a computer system in that a game involves inputs, outputs, and processes and that developing such a game system involves rapid prototyping as well as determining and specifying requirements. Advanced students of game design may wish to talk to users prior to creating their first prototype.

Additional modeling exercises may be added in which students are asked to draw other diagrams such as a UML class diagram showing objects (e.g., ball, racket) and their relationships within the game context. The reflection part of the assignment is very valuable for students' learning as it gives them an opportunity to reflect on their experience and explicate their resulting understanding of requirements discovery by prototyping. We have required students to post their experience reports online in a discussion forum so that they may share their learning with others, which has been helpful. A small minority of students has reported difficulties in learning the game creation software but this issue was seen more as a function of the student's background with games or the type of software they chose. In the latter case, some guidance in software selection may be helpful to provide to students, and typically, this is best done by other more experienced students.

5. CONCLUSION

We have found that the use of game prototyping as a pedagogic technique to communicate concepts of model-based software development and requirements determination and specification can be innovative, instructive, and entertaining. Student have a natural interest in gaming and the process of rapid game creation with a user makes for an interesting exercise in exploring user requirements and capturing them via use-case diagrams, UML class diagrams, and other modeling approaches. Obviously, this type of assignment can be a useful supplement but is not meant to be a replacement for teaching foundational concepts or for a semester-long systems project. We hope instructors can creatively adapt the ideas and approaches in this paper to suit their unique teaching situations.

6. REFERENCES

Cooper, S., Wanda D., and Randy Pausch. (2003) "Teaching objects-first in introductory computer science." ACM SIGCSE Bulletin. Vol. 35. No. 1. ACM.

Dalal, N., Dalal, P., Kak, S., Antonenko, P. and Stansberry, S. (2009). Rapid Digital Game Creation For Broadening Participation In Computing And Fostering Crucial Thinking Skills. Int. J. Social and Humanistic Computing, 1, 2, 123-136.

Harel, I. E. and S. E. Papert (1991). Constructionism, Ablex Publishing.

Overmars, M. (2004), "Teaching Computer Science through Game Design.", Computer, v.37 n.4, p.81-83.

Solomon, J.: 1994, 'The Rise and Fall of Constructivism.' Studies in Science Education, 23, 1-19.

AUTHOR BIOGRAPHIES

Nikunj Dalal is Professor of Management Science and Information Systems in the Spears School of Business at Oklahoma State University in Stillwater. His research interests include learning, wisdom computing, rapid computer game creation, philosophical issues in information systems, modeling, and Web perception. His work has been published in journals such as *Communications of the ACM*, *Decision Sciences*, *European Journal of Information Systems*, and *International Journal of Human-Computer Studies* among others.

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096