An Evaluation of Using a Game Development Framework in Higher Education

Bian Wu , Alf Inge Wang , Jan-Erik Strøm , and Trond Blomholm Kvamme
Dept. of Computer and Information Science
Norwegian University of Science and Technology
Bian/alfw@idi.ntnu.no, janerist/trondblo@stud.ntnu.no

Abstract

This paper describes an application of a Game Development Framework (GDF) - Microsoft XNA in software architecture (SA) course at Norwegian University of Science and Technology (NTNU) and evaluates how well the GDF is to use and integrate in a software engineering (SE) course. The result of the evaluation is based on the questionnaire with 9 types of general questions related to SE learning. In most aspects, the result shows that XNA is a suitable teaching aid in SE learning and can be used to teach SA. It is easy to use and save students time in development, thus let them have more time focusing on the course theory.

Keyword: *Game development framework, XNA, Software architecture, Software engineering education, Evaluation.*

1. Introduction

Research on games concept used in higher education has been done before, e.g. [2, 4, 3], but we believe there is an untapped potential that needs to be explored. This paper will change the angle from using games to teach to applying GDFs in student projects for learning computer skills, extending its application as a teaching aid in higher education. The GDF can be integrated mainly in three ways with a university course. *First*, it can be used to develop games that can replace traditional exercises. *Second*, it can be used to develop games that can be integrated in lectures to improve the participation and motivation of students. *Third*, the students can use a GDF in projects to develop software to understand the courses content related to computer science.

This paper will focus on GDF's application in higher education and evaluate its application in an existing course. The evaluation focuses the suitability of a specific GDF to be used by students, and whether the GDF is useful for the teaching and understanding course theory.

2. Application of a GDF in Higher Education

This section is a case study of Master course of SA at Norwegian University of Science and Technology (NTNU) to elaborate the application of a GDF used as a teaching aid in higher education.

2.1. Choice of the GDF

The course staff started searching for any GDF that provided high-level APIs to ease game development and that was easy to learn. As many GDFs were immature, we ended up with choosing Microsoft XNA (Xbox/DirectX New Generation Architecture) framework [5]. It was the most suitable framework for fast game development at that time. Another reason for choosing it was that support for developing game for XBOX 360 would be a motivation factor for students to put an extra effort into projects.

2.2. Student projects based on XNA

In NTNU's SA course, the goal of the project is for the students to apply the methods and theory from the course to design a SA and to implement a system based on XNA framework. The project consists of the following phases: 1) *COTS (Commercial Off-The-Shelf) exercise*: Learn the technology to be used through developing a simple application. 2) *Design pattern*: Learn how to use and apply design pattern by making changes in an existing system. 3) *Requirements and architecture*: List functional and quality requirements and design the SA for the application (a game). 4) *Architecture evaluation*: Use the ATAM (Architecture Tradeoff Analysis Method) evaluation method to evaluate the SA of project in regards to the quality requirements. 5) *Implementation*: Do a detailed design and implement the application based on the created architecture and on the changes from the evaluation. 6) *Project evaluation*: Evaluate the project as a whole using a PMA (Post-Mortem Analysis) method [1].

The course staff issues the task to make a functioning game using XNA. The game has to be designed according to a specified SA. Further, the students had to develop an architecture where they had to focus on one particular quality attribute: *Modifiability*, the game architecture and implementation should be easy to change in order to add or modify functionality; or *Testability*, the game architecture and implementation should be easy to test in order to detect possible faults and failures. The course's workload was 25% of one semester and students were grouped in 3-4 persons and spent most time on the implementation phase (6 weeks).

3. Evaluation of XNA used in a software architecture course

This paper investigates the XNA framework's usefulness for teaching students SA based on book "Software Architecture in Practice" [6]. Concretely, we investigate the following research questions:

- R1: To what degree does the COTS influence the learning process?
- R2: How much time is spent on technical matters and on architectural matters?
- R3: How difficult is it to integrate known architectural and design patterns and learn the necessary prerequisite skills to be able to develop programs?
- R4: How much do the students feel they have learned about SA through the game development project?

3.1. Questionnaire and Result

The participants of our survey were postgraduate students of NTNU's SA course. We published a questionnaire using the existing e-learning platform (It's Learning) three days after the delivery deadline of the students' projects, and received a total of 46 responses to the general questionnaire. Table 1 shows statistical results of quality attributes from students' projects. Table 2 lists the 9 general items and students responses.

Table 1: Distribution of responses related to assigned quality attributes

46 Responses in XNA	55% Testability
	45% Modifiability

Table 2: The 9 general questions labeled Q1-Q9

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly agree

Q1: I found it hard to come up with good requirements	5%	30%	40%	20%	5%
Q2: I think the COTS did not hinder the design of a good architecture(Total)	5%	20%	35%	35%	5%
Q2.1 I think the COTS did not hinder the design of a good architecture(Testability)	10%	25%	30%	35%	0%
Q2.2 I think the COTS did not hinder the design of a good architecture(Modifiability)	0%	10%	45%	35%	10%
Q3: I found it difficult to evaluate the other group's architecture in the ATAM	0%	20%	15%	45%	20%
Q4: I think the COTS made it easier to identify architectural drivers(Total)	10%	15%	55%	20%	0%
Q4.1 I think the COTS made it easier to identify architectural drivers(Testability)	15%	15%	60%	10%	0%
Q4.2 I think the COTS made it easier to identify architectural drivers(Modifiability)	0%	15%	50%	35%	0%
Q5: I found it difficult to focus on our assigned quality attributes(Total)	10%	25%	10%	25%	30%
Q5.1 I found it difficult to focus on our assigned quality attributes(Testability)	10%	0%	0%	30%	60%
Q5.2 I found it difficult to focus on our assigned quality attributes(Modifiability)	10%	50%	20%	20%	0%
Q6: I found it easy to integrate known architectural or design patterns(Total)	0%	10%	40%	40%	10%
Q6.1 I found it easy to integrate known architectural or design patterns(Testability)	0%	10%	35%	45%	10%
Q6.2 I found it easy to integrate known architectural or design patterns (Modifiability)	0%	10%	45%	35%	10%
Q7: I spent more time on technical matters than on architectural matters	5%	20%	30%	30%	15%
Q8: I spent too much time trying to learn the COTS in the start of the course	5%	35%	30%	25%	5%
Q9: I have learned a lot about software architecture during the project(Total)	10%	15%	30%	40%	5%
Q9.1: I have learned a lot about software architecture during the project(Testability)	10%	10%	35%	45%	0%
Q9.2: I have learned a lot about software architecture during the project(Modifiability)	10%	20%	25%	35%	10%

3.2. Analysis of questionnaire results

Here we will evaluate the results against the stated problems.

R1: To what degree does the COTS influenced the learning process?

- *The requirements gathering and specification*: Reflected in Q1 of Table 2, the result did not show that the COTS made a significant impact on requirements phase of the project.
- *The design of the architecture*: From Q2, most of students agreed that the COTS did not hinder design of a good architecture. The major reason is that XNA supports different types of games development with flexible architecture. There was a tendency that modifiability groups thought the COTS was a less hindrance than the testability groups.
- *The ATAM evaluation*: Reflected in Q3, most of students found it difficult to evaluate another group's ATAM document. The main reason is because of XNA's open environment and different types of games, all of them have their own structure and playing style.
- *Architectural Drivers*: From Q4, it seems safe to say that the COTS did not influence the difficulty of identifying architectural drivers. And students who focused on modifiability

tend to agree more that the COTS made it easier to identify architectural drivers, while students who focused on testability tend to disagree more.

- *Quality Attribute Focus*: From Q5, the results indicate that choice of COTS does not have much influence on difficulty of focusing on the assigned quality attribute. And generally students with modifiability focus found it easy and students with testability focus found it difficult. The probably reason is that students have a much better understanding of modifiability, but creating a program that makes testing easier is a whole new area.

R2: How much time is spent on technical matters and on architectural matters?

From Q7, the choice of COTS influences the time the students have at their disposal to focus on architectural matters. From our own experience, the XNA environment is much more user-friendly with a high-level API, the probable reason was that many students were completely new to both C# and XNA.

R3: How difficult was it to integrate known architectural and design patterns and learn the necessary prerequisite skills to be able to develop programs?

- *Integrate known architectural and design patterns*: From Q6, the result shows that generally the majority of students thought it was easy. Also, we found when looking at the quality attribute distribution, there was several suitable patterns presented, such as Model-View-Controller, Pipe and filter, Layered, Task Control and so on.
- *Learn the necessary prerequisite skills to be able to develop programs*: From Q8, results show that majority students disagree that they spent too much time learning the COTS. Also, from our own experience, we received almost no help requests from students' groups.

R4: How much did the students feel they have learned about SA through the project?

From the results of Q9, most of the students feel they have learned a lot about SA throughout the project. For the negative attitude in result, probably due to the first time to use XNA in teaching and some of students might have become spellbound by the fun of creating a game, thus focusing more on game play than architecture.

5. Conclusion

In this paper we have presented an evaluation of the XNA integrated into a SA lecture. The result shows that XNA is easy to use, requires little time to develop, and supports different types of game development. Further, the students claimed that XNA framework contributed to increased learning and motivation.

6. References

[1] A. I. Wang, T. Stålhane. Using Post Mortem Analysis to Evaluate Software Architecture Student Projects, Conference on Software Engineering and Training 2005, 8 pages.

[2] Alex Baker, Emily Oh Navarro, and Andr´e van der Hoek.Problems and Programmers: an Educational Software Engineering Card Game. In ICSE '03: Proceedings of the 25th International Conference on Software Engineering, pages 614–619,Washington, DC, USA, 2003. IEEE Computer Society.

[3] Emily Oh Navarro and Andr´e van der Hoek. SimSE: an Educational Simulation Game for Teaching the Software Engineering Process. In ITiCSE '04: Proceedings of the 9th annual SIGCSE conference on Innovation and technology in computer science education, pages 233–233, New York,NY, USA, 2004. ACM Press.

[4] Lasse Natvig, Steinar Line, and Asbjørn Djupdal. Age of Computers: An Innovative Combination of History and Computer Game Elements for Teaching Computer Fundamentals. In FIE 2004: Proceedings of the 2004 Frontiers in Education Conference, 2004.

[5] Microsoft corporation.XNA developer centers. http://msdn.microsoft.com/en-us/xna/aa937794.aspx, Retrieved June,2008

[6] P. Clements L. Bass and R. Kazman. Software Architecture in Practice Second Edition, 2003. Addison-Wesley.