Joe Klemmer

Jill Ellen Coddington

SG 150

June 28, 2009

Methodologies Used in Creating Game and Application Software

To the nonprofessional, software on a computer that does something through interaction with the screen, disk drives, keyboard, mouse, and/or other input/output devices. How it does, what it does is not a concern for them. As long as it works, as long as it does what it is supposed to do, everything is fine. However, for people in the software development field, software is much more than the result of typing and clicking. It is a symphony of ones and zeros composed by programmers whose instruments are programming languages, design tools, development frameworks and methodologies that pull it all together. However, is all software development the same? Can there be a universal way to build programs and systems? The answer to that question is no. Software development is different depending on what the software does and where it will run. Two kinds of software that every computer user has experienced are games and applications. The development of game and application software is interesting to compare due to their similarities; programming languages, development models, design methodologies, and differences; developmental focus, technical requirements, eventual goals.

Game development is, by its very nature, software development. Practices and methodologies for general software development exist in game programming just as in any other programming field. The primary programming languages used for game development are C++, C, and Java supplemented with assembler and, for online games, AJAX and/or Flash. These languages are relatively portable to other architectures/platforms, which gives developers the opportunity to release games for multiple platforms. Development models for programming have started to adopt Agile Programming Systems. It is no different for game development. Agile models allow for faster development with fewer show stopping bugs. Additionally, game development is very graphics heavy.

Application development includes a number of the same characteristics as game development. Similar programming languages and development models apply to application programming. Graphics are also an important part of applications as the three major operating systems, Microsoft Windows, Apple Macs and Linux, all function in graphical modes. It would seem by the preceding information that there is little difference between game and application development. This, however, is not the case.

The differences between game and application development are significant once you look beyond programming languages and development models. The biggest differences are the goals towards which the software intends to reach. Applications focus on accomplishing a task or solving a problem. They can range in size and scope from a simple text editor to massive programs like those that run Wall Street and Army Knowledge Online. However, no matter what their size or industry they exist to DO something. With games, the goal is to make a program that is entertaining within its genera. Games rarely have any other designs but to entertain us. An exception to this are games like Wii Fit. Having no single goal gives the developers room to develop storylines, plots and characters. Games also have far greater and more complex audio/visual requirements than applications. A word processor does not need a script and voice actors.

Along with the various aspects of developing software, be it games or applications, there are legal matters that need addressing. The issue of copyrights is one that can be very tricky to handle. Copyright law in the US, and through out the entire world, is complex and convoluted. The laws and definitions that govern it have changed so drastically that the original lawmakers would not recognize it at all. Now that we are in the computer age things like copyrights and so-called “intellectual property” are forced to fit the concept of digital media. Nevertheless, software development needs to abide by the laws as they are. For example, over the last decade, a number of software related copyright infringement cases have come up. The BlackBerry Lawsuit and Amazon’s suit against Barns and Noble along with SCO sues everyone are some of the cases which typify the complexity of copyrights in the digital age.

In order to address these and other pitfalls in the modern software development arena, has worked to formulate a Code of Ethics for developers. The most prominent of these is the “Software Engineering Code of Ethics and Professional Practice” recommended by the ACM/IEEE-CS Joint Task Force (Association for Computing Machinery, Inc. and the Institute for Electrical and Electronics Engineers, Inc.). This document is considered the “the standard for teaching and practicing software engineering” and is a guideline for the software development community. Other resources available that addresses this issue are the books “Professional Awareness in Software Engineering” by Tom DeMarco (DeMarco) and “The ethics of cyberspace” by Cees J. Hamelink (Hamelink). An example of unethical behavior is the hypothetical scenario of the “Case of the Killer Robot” (Epstein).

In the final analysis, software engineering for games and applications not only have practical matters to contend with, like development models and languages, but must contend with legal and ethical aspects as well. When all of these factors align properly the product, be it the next great game or killer app, will be both technically and functionally superior. This is worth the effort.

Works Cited

Association for Computing Machinery, Inc. and the Institute for Electrical and Electronics

Engineers, Inc. "Software Engineering Code of Ethics and Professional Practice." 1999.

Association for Computing Machinery. 27 June 2009

<http://www.acm.org/about/se-code>.

Chalmers, Rachel. "Amazon Sues Barnesandnoble.com for Patent Infringement." 25 October

1999. Find Articles at BNET. 27 June 2009

<http://findarticles.com/p/articles/mi\_m0CGN/is\_3775/ai\_56914605/>.

DeMarco, Tom. "Professional Awareness in Software Engineering." The Atlantic Systems Guild.

27 June 2009 <http://www.systemsguild.com/GuildSite/TDM/Professionalism.html>.

Epstein, Richard G. "Silicon Techtronics Employee Admits Faking Software Tests." 2000.

Online Ethics Center for Engineering. 27 June 2009

<http://www.onlineethics.org/CMS/computers/compcases/killerrobot.aspx>.

Hamelink, Cees J. The ethics of cyberspace. Sage Publications Ltd, 2001.

Pamela Jones, et. al. Groklaw. 2003-2009. 27 June 2009 <http://www.groklaw.net>.

Patterson, Lyman Ray and Stanley W. Lindberg. The nature of copyright. University of Georgia

Press, 1991.

Pegoraro, Rob. "BlackBerry Lawsuit Is Patently Absurd." 25 February 2006.

washingtonpost.com. 27 June 2009 <http://lispurl.com/caaadddadddadar>.

Radding, Alan. "Agile Programming Systems." 4 February 2002. Computerworld. 28 June 2009

<http://is.gd/1gEq1>.

Stanford Center for Internet & Society. "Frequently Asked Questions (and Answers) about

Piracy or Copyright Infringement." Chilling Effects Clearinghouse. 27 June 2009

<http://www.chillingeffects.org/piracy/faq.cgi>.

US Army. Army Knowledge Online (AKO). <http://www.us.army.mil>.