Work Term Report 1
COMP 3999A

FOR THE SUMMER 2013 PLACEMENT AT NAKINA SYSTEMS – NETWORK INTEGRITY

Ankush Varshneya (100853074)
AnkushVarshneya@cmail.carleton.ca
Manager: Greg Johnston (Development/Design)

Ankush Varshneya
415 Cresthaven Drive
Nepean ON,
K2G 6Z4

Friday, August 16, 2013

Dr. Irwin Reichstein
School Of Computer Science
Carleton University
Ottawa, Ontario
KIS 5B6

Work Term Report 1 COMP 3999A

Dear Sir,

This is my first work term report and was prepared in connection with course COMP 3999A for my Co-op placement at Nakina Systems – Network Integrity. I worked as a junior developer in the development department in the design team under the supervision of my manager Greg Johnston and my mentor Benjamin Boudreau.

Nakina Systems provides a host of Network Integrity Management solutions to the telecommunications industry worldwide. With the large variety of networks platforms and devices available today it gets hard to communicate with the network. Nakina Systems provides solutions to easily communicate with cross platform network elements, all from one secure and centralized group of Product.

This report highlights some key concepts I learned during my placement as well as my contributions to Nakina systems products. During my first term I mainly dealt with fixing bugs, implementing a minor feature as well as working on mock-ups and documenting new features/updates that are going to be released as part of a future release of the Log Manager (a key component of Nakina system software). I have been fortunate enough to be rehired for another term to implement my mockups.

This report has been prepared and written by me and has not received any previous academic credit at this or any other institution. I would like to thank Yves Thibeault (Software Design manager) for helping in the reviewing of this report.

Sincerely,
Ankush Varshneya
Student number 100853074

ABSTRACT (EXECUTIVE SUMMARY)

I was hired by Nakina Systems as a junior developer to help implement a new log management system from the ground up for a future release of Nakina's platform software. This report documents the key concepts I learned about the development cycle at Nakina system. During my first term I was assigned the responsibility of gathering requirements from the current users of the log manger application for what they would like to see in a new release of the log manager application. From the requirements and discussions, I developed conceptual User Interface designs (UI mockups) and design documents that meet everybody’s need. I have been rehired for another term to implement my mockups and documentation into an actual product. Aside from documenting features this documents also talks about some of the software defects I was assigned to fix, some of the minor features implement, and the problems encountered while doing so. Lastly this report highlights my work experience, day to day activities, how did I applied my academic knowledge to work and further my knowledge by getting work experience, different problems faced at the work and the type of approaches used to tackle those problems, tried to relate my work experience to academic studies, my accomplishments, and abbreviations used in my work term report.

ACKNOWLEDGMENTS

I would like to thank my mentor Benjamin Boudreau for helping me make the transition from a school environment to a work environment, for helping me understand the underlying technologies used by Nakina system Products and lastly for guiding me throughout the term for what I need to succeed.

I would also like to thank my supervisor Greg Johnston for helping me with developing my FS (Functional Specification documentation).

Lastly I would like to thank Benjamin Boudreau, Greg Johnston, Dean Clark, Trelaine Farrow, and the people in the support department for influencing my mockups and presenting features they would like to see introduced/improved with the new iteration of the Log Manager

Table of Contents

.

INTRODUCTION

Nakina system is a company based in Kanata; it is a leading manufacturer of telecom network management systems and provides network integrity management solutions to telecommunication industries worldwide. Nakina also provides products that enable service providers to more quickly and cost effectively deliver new services in a multi-vendor network infrastructure. Recently Nakina systems won a prestigious COMET award. Its product range includes NI-Guardian, a Centralized Identity Management and Single Sign On solution for networks that centralizes security administration, NI-Collector an Inventory Discovery and Reconciliation solution, NI-Controller, a gold standard configuration audit solution. The above products work on NOS (Network Operating System) otherwise known as NI-Framework, which rely on OSS framework and Oracle at the backend.

Nakina System R&D organization consists of three main departments which are Development, Verification, and Support. The development department consists of four design teams, framework team, two application teams and a device adapter development team. The role of the support department is to provide support first level to Nakina systems clients. The verification department consists of a quality assurance team of product testers that thoroughly test products before a major release, to ensure the product is bug free and up to the client's satisfaction. Any discrepancies gathered by verification or directly by support (via a client) are reported to the development department through an internal system called JIRA, an electronic product tracker. A JIRA is not only limited to bugs but it may also contain request for new features, JIRA's are then usually assigned to a team in the development department based on the product, it deals with. The job of a developer is to address JIRA's and hence fix bugs or implement new features. After a JIRA has been addressed, it is then reassigned to a member of the verification department so that they can take any appropriate action.

WORK EXPERIENCES

Each development team has weekly SCRUM meeting and is arranged by the team managers. This is where developers discuss their weekly progress in terms of what they are working on and if they need help from fellow developers; new products and feature implementations are also discussed here. These meetings are a great opportunity to find out what's going around you and what your fellow colleagues are working on and it gives you a great understanding of the products and to follow up on any question you may have. Some developers such as myself attend other teams scrum meetings as well, as they may be working on a product that requires ongoing discussions with multiple teams. Sometimes SCRUMS are also used to give business updates, such as contracts the company may have gotten.

Tools

Working on Nakina systems products involve heavy use of JAVA programming based on JAVA J2EE architecture and OOP design principles. Its IDE consists of Eclipse and is integrated with Apache Ant, to compile the code and Apache Subversion SVN/CVS to help with software version control.

.

Since Nakina systems product range consists of web applications, one would think that the application relied more on java script but this is not the case here. Java is used due to its object oriented design and use of coding. It is much easier to track bugs and implement new code in a strongly typed language such as java.

Nakina system then uses web frameworks to compile the application into java script. Traditionally Apache struts web framework is used in Nakina systems application, but with the newer applications Nakina systems has moved towards a Google Web Tool Kit (GWT/GXT) solution.

Nakina systems uses an internal Wiki where developers post guides, code syntax, and other various snippets of useful proprietary information for some of the technologies used at Nakina systems. The information provided by the wiki helps one to wrap their head around the large product range.

Oracle data base and SQL Developer are used for the backend store and Oracle Web Logic server is used to deploy the web applications / products used by Nakina systems.

Compiling the project is done through the terminal which helped me develop my Linux bash skills.

Software Defect Fixing

I was originally assigned the task of fixing software defects. Throughout this process I really got a greater understanding of how their products work. I got to see the tools and technologies in action as mentioned in the tools section. Also I got an understanding of how the development department is organized and what developers deal with what part of the product. It also helped me understand how their bug tracking system works. It helped me develop my understanding of their software, the way it is written and the guide lines followed in terms of the UI.

As I was accomplishing this task I found myself using techniques acquired from discrete math and logic courses that I would never have thought to use in a real life scenario. For example, I was using truth tables and logical statements to help pin point bugs that would have otherwise taken much longer to solve and I was able to communicate my bug findings to my mentor much clearly with the use of logical statements and truth tables than I would have been able to do otherwise. It allowed me to connect contents learned from non-programming courses into my coding skills and helping develop my debugging and coding skills.

Fixing the software defects involved a large variety of code written by many writers, looking at such a variety of code helped me explore a variety of coding styles and practices and in turn influenced my own style by a great degree. Sometimes certain code segments lead me to interact with other developers, which allowed me to explore how other developers contribute to Nakina Systems products.

Implementing new software

To complement my bug fixing tasks, I was given the task of implementing a new GWT desktop product based on an existing product namely NI-Guardian. Normally their desktop products require additional dependencies such as their own EAR, table space, and product specific dependencies. This is tedious for other developers because if they are testing for small functionality or if they need to redeploy the desktop software than they have to go through the process of getting a tables space and deploying other dependencies. So with the help of my mentor, Benjamin Boudreau, I made a stripped down version of the desktop software that is essentially the same, but it does not require all the dependencies and it produces one simple WAR file which can be deployed by pretty much any Web logic Console (their web application deploying utility). This allows for developers to run quick tests without having to do a whole lot of deployment.

Log Manager

The main objective for me in this term was to improve the existing log manager. This objective was worked on in parallel to the objectives listed above. The main objective of working on Log manager is to give an exposure to the full software delivery life cycle in four steps i.e. Requirements, Design, Verification, and delivery of the product.

During the requirement phase I had to arrange meetings with the key users of the log manager application to gather what they would like to see in a future release of the log manager application.

My main task in accomplishing the requirement phase was to generate various UI designs via mockups using Balsamiq Mockups software. These mockups were then shown and discussed with my manager, mentor, Support users, and other selected people that used the application to allow for better requirement feedback.

The next task was to see if the requirements were feasible by discussing them with my manager, mentor and fellow developers. Based on the requirement I was asked to draft a functional specification document (FS) which as of the date of writing this report has yet to be reviewed by interested parties.

Some of the broader requirements are arranging the logs to support the syslog format, Internationalize searching logs to allow for clients to support their language, provide better troubleshooting of lots by arranging the logs structure to easily reduce noise.

As for the other phases I have yet to complete them in the following term.

Challenges and Solutions

During my bug fixing phase I encountered bugs that were interrelated to many products, so it was very confusing where to start from. My mentor provided me with some quick walkthrough tutorials of their code and appropriate articles to read on their internal wiki, which complemented his tutorials. After reading the wiki articles I had a better understanding of where I needed to start looking (code wise) to fix the bug. Some of the tips and techniques suggested by my mentor helped me greatly increase my productivity and knowledge.

While debugging I would always try and follow up with my mentor and explain to him the logic behind, how I was approaching fixing the bugs. Discussing the bugs helped me get feedback, so I can stay on the right track.

During the mockups phase it was hard to get inspiration for designing the actual mockups. My manager suggested that I should meet with my co-workers in the support department, and they were able to provide me with some used cases of what they used the log manager for in support cases. They also allowed me to view real time logs in their system and provided me with things that they would like in the new iteration of the log manager by showing me other applications they used alongside the log manager to resolve support issues. They suggested more people I can get in touch with to gather more information on requirements and used cases of the log manager. All these information helped me develop useful mockups and hence gather useful requirements.

My Mentor helped me create a wiki article and suggested that I upload all the log manager related items on it. With the help of the wiki I was able to communicate my ideas to more interested people and got greater ongoing feedback on mockups; which paved the way for my functional specification document.

After completing the mockups I had to implement a functional specification document which highlighted the requirement, used cases, and explain the mockups. During this process my manager greatly helped me as to how I should structure this document and proofread the document to show how I can make improvement to suit the standard according to the Nakina systems guidelines.

REFLECTION

Reflecting back at the term, it was a fantastic learning experience. Although the term was not as technical as I had envisioned it to be, it did teach me about a formal software cycle that is present at Nakina Systems. Attending weekly scrum meetings helped me with getting the bigger picture of how organizations such as Nakina systems work at large. I learned a lot about gathering product requirement from various parties involved and how to make useful mockups and documentation. Prior to this placement I actually had no idea how involved the documentation process can get, because in previous jobs I had been spoon fed requirements and only had to provide a solution in terms of a website.

The only idea of documentation I had was commenting my code. In contrast here at Nakina I had the opportunity to go out and interact with the teams and at times people at different departments. This helped to improve my team and communication skills. I got to learn what documentation actually is; learning how to design mockups helped me explore other media in which my ideas can be portrayed. Previously I was accustomed to coding something and evolving the code to meet the expectation of others. But having something in terms of pictures and documentation can really go a long way in helping to start coding. Although I have not officially started coding my mockups this term, I feel at a better position for next term when I will start coding because I will have the documentations and mockups to refer to and hopefully I will not have to alter the code to such an extent as I have had to do for the past jobs.

Working at Nakina systems helped me understand how web applications don't have to be restricted to just languages such as JavaScript or PHP. Languages such as Java can be used with the help of a Google Web Tool Kit (GWT) web framework to create equally powerful web applications as their counterpart in other weakly typed languages. Adding and fixing up code in Java is much easier and organized to do as suppose to a language like JavaScript.

Fixing up bugs helped me understand the importance of bug/project tracking software such as JIRA and how organizations such as Nakina System communicate between their various teams to provide solutions to their clients. I also learned how version control software, such as Apache's Subversion, is used to maintain different versions of software's. I also learned key version control principles such as branching software to fit needs of different clients.

Currently I am exploring the client Java Enterprise Java Bean APIs (more commonly referred to as SAM APIs) involved with Nakina systems software, so that I can help propose changes needed with the introduction of the new log manager. With the help an Oracle SQL Developer I am studying the structure of the current logs so I can see what needs to be changed to achieve the functions outlined in my requirements for the new log manager.

Relation to Academic Studies

This placement helps me enhance the learning I received at university. It helped me see some of the software engineering concepts such as design patterns in action in real life programs. It helped me develop my skills in Java.

Debugging helped me work on the pre-existing material, which is really hard to come by in a course offered at university. Since the pre-existing code has been written by many developers, going through it helps one to really understand the new styles and gives you a variety of coding practices. Hence it helps you build your own style in the process. Going through bugs is really helps you learn trouble shooting tips, which are harder to learn in a course. In contrast the courses I have been through in university required you to write your own code with the exception of 1 or 2 courses, where there is very little code written by the professor or teaching assistant, which doesn't offer as large a variety.

Thinking about the documentation I had to do, I felt that I had not really been taught it in any course to any extent. I think I would have been more productive had I been given some exposure to documenting to what mockups are and how requirements are meant to be gathered.

Career Development

In the past I have been used to just coding and documenting my code. But working at Nakina helped me see that there is more to being a developer than just coding.

Learning the documentation part of the development process has really been an eye opener for me. I was hoping that I would have started implementing code with in the first few weeks of the job, but documenting took a big chunk of my placement. At first I agree I found the topic rather dull, I knew that development involved documenting but I would never thought it to be to such an extent. But as I was going through the process of gathering requirement, making mockups and documenting the features need, I realized how important this process really is and how it will make implementing the code a lot easier, as I will have a lot more material to refer to. In the end, the making mockups and discussing the requirements with everyone, felt exciting and I have a whole new appreciation for the process.

I look forward to moving on to the code implementation phase with my next term at Nakina systems, and hopefully then will help me learn a lot more in detail about Nakina systems.

ABBREVIATIONS

IDE - Integrated Development Environment
UI – User Interface
NOS – Network Operating System
JAR – Java ARchive
EAR – Enterprise ARchive
WAR – Web application ARchive
PHP - Hypertext Preprocessor
API – Application Programming Interfase
NI – Network Integrity
SVN – Apache Subversion
CVS - Concurrent Versions System
GWT – Google Web Tool Kit