Memo

|  |  |
| --- | --- |
| To: | Sandra Stark |
| From: | Martin Lurette |
| cc: | Kevin Brascoupe |
| Date: | August 24th 2017 |
| Re: | Computer Science Co-op Work Report |

Between May 1st 2017 and Aug 18th 2017, I was employed by Versaterm Inc. as a student developer working with the U.S. Records Management System (RMS) team under my supervisor and mentor, Kevin Brascoupe. Versaterm is a leading supplier of public safety systems used by police, fire and ems services throughout North America. Versaterm believes that technology is not the solution but only an enabler to solve problems in an environment where both reliability and predictability are paramount.

While working at Versaterm I had the opportunity to work on many different projects and tasks. The most notable are: string localization, statistical extracts for Pennsylvania state police, adding new fields to the existing REST API load service, creating automated test cases for the REST API, simplifying a deployment tool for updates and patches on client sites and a tool to help create relationships between entities.

The string localization was logic that already existed on the Canadian RMS to be able to switch between English and French, so I took that existing logic and moved it over to the U.S. RMS to offer agencies the ability to localize regional terminology for printed hardcopies of reports (for example, one agency might use the term “field interview” while another might use “street check”).

Creating the statistical extract used existing logic from the National Incident-Based Reporting System (NIBRS) already used by the FBI for calculating and researching crime statistics. This data contains information about all related criminal activity for a region for a given month. Pennsylvania’s State Police needed to implement a slightly different format for their own data collection. I converted the existing NIBRS logic to conform to Pennsylvania’s different requirements.

The REST API is a new tool for the U.S. RMS team that allows stateless transfers of data to the database using HTTP by sending and receiving JSON strings. I added fields to the REST API’s load functionality to allow users to add medical, prisoner property, release hold and privatization information. Continuing with the REST API, I then created automated test cases using the existing automated test engine to test whether the data received from the REST API matches what we expect to receive from the database. The test scenarios also included the different ways to search for records, non-existing records, invisible or private records and authorization issues. These test cases proved useful as they could find a bug in the authentication process.

I then created another automated test case to validate the documentation for the REST API load functionality. The REST API uses YAML (stands for YAML Ain’t Markup Language. It is a human readable data serialization language) to create intuitive web pages for the documentation at the click of a button. While this is a very easy way to create documentation, it leaves room for error if it has any spelling mistakes, incorrect data types or incorrect data structures. The first way to find these errors was to test existing JSON files and compare it against the YAML file to find if any fields are missing or misspelled. The second way to find errors was by converting the YAML file into a JSON file with dummy data that corresponds to the data type. This JSON file can then be converted and then loaded into the existing record data types. If any of the loaded fields came up null, you know it is an incorrect data type or an incorrect data structure. As a bonus, this method provides JSON templates that can be saved and used for future testing, since the only ways to get a JSON file that would load into the REST API were to make one manually or find someone who has an existing JSON file from previous work on the REST API.

My next task focused on a deployment tool used to release patches and updates to the client. The main issue was that the patch would have to be downloaded locally before downloading it onto the client server. For clients with a slow VPN, this could take hours to complete for a large file. The solution I implemented made it so that the user could select to download directly to the client server. I also added the option to avoid the download process completely if the download destination is on an external server, and display the temporary location on screen of the patch or update so it can be downloaded using SCP over command line.

The next project I worked on was a prototype for a tool that would intuitively and visually create relationships between victims and offenders. For a given occurrence there could be multiple offenses, multiple victims and multiple offenders. The victim of one offense could be the offender of another offense and it gets complicated to create and maintain the relationships between those people. To accomplish this, I created a prototype what would display to the user a list of offenses and people related to an occurrence, and allowed the user to create groups where offenses and people can be added. Once people are added as victims and offenders, it creates another list of all the possible relationships and the user can then add the corresponding relationship.

My workstation was a Windows 10 desktop computer. The server uses Linux’s Red Hat operating system. The main programming language I used over the summer was Genero BDL (Business Development Language) but I also used XSL, XML and YAML. The database uses IBM Informix.

During this past summer, I had the opportunity to use some of the skills I’ve gained at heritage. Most notably, the planning and deployment of new projects. I feel that the Systems classes at Heritage gave me the skills required and prepared me to properly design, explain and make new projects/applications from concept to deployment. During the summer, the main skill that I acquired and improved was learning to communicate within a business environment. I had many chances to sit down and communicate ideas with professionals who work in very different fields and enjoyed being forced to look at problems in a whole new way.

Overall, I felt quite prepared for this co-op placement. I could be a little biased since this was my second summer with Versaterm. I came into the company already knowing some people and was familiar with the office culture. I felt comfortable with the responsibilities and workload I was given. I always had people I could reach out to for help if needed and was offered many different projects and tasks to work on to keep me busy. Since Versaterm’s product line is so vast and uses an uncommon programming language, I don’t think there is much more that could have been done to prepare me for this position.

In conclusion, I am very satisfied at how this summer experience turned out. I’d like to extend some warm thanks both to Versaterm and Heritage College for making this possible and I thank you for all the hard work that is put into the co-op program. I would recommend this position to anyone interested in a serious career in Computer Science.