# ANDREW FORWARD

203-330 LORETTA AVE S • OTTAWA, ONTARIO • K1S 4E8 •
PHONE (613) 795 5273 • E-MAIL aforward@gmail.com


## EDUCATION

**Ph. D. Candidate Computer Science**
Expected completion date Sept 2010
University of Ottawa

**M. Sc. Computer Science**
Summa Cum Laude
University of Ottawa

**B. A. Sc. Software Engineering**
Magna Cum Laude
University of Ottawa

**Ontario Secondary School Diploma**
Bilingual certificate, top 1% in Ottawa-Carleton
South Carleton High School

**Royal Conservatory of Music**
Grade 8 Piano and Level 2 Theory

## TECHNICAL PROFICIENCY

**Databases**

MS Access, MS SQL, SQLServer, MySQL, Oracle (8i, 10g), PostgreSQL

**Languages**

Ant, ASP, ASP.NET, Ajax, Bash, C#, Capistrano, Crontab, DNS, CSS, HTML, HTTP/S, Java J2SE, J2EE, JavaScript, JavaServerFaces (JSF), JSON, JSP, NAnt, NDoc, Perl, Phing, PHP, PL/SQL, Ruby, Ruby on Rails, Selenium, SOAP, TCP/IP, UDP, UML, VB6, VBScript, Webtest, XHTML, XML

**Servers**

Amazon S3 (Cloud Computing), Apache, CruiseControl, CruiseControl.net, GitHub, Heroku, IIS, LAMP, Linux (SUSE, Ubuntu, Fedora), Mac OS X, Mongrel, Nginx, Passenger, phpUnderControl, Rackspace, Slicehost, Tomcat, UNIX (Tru64), Windows 9x, Windows NT, Windows XP, Windows 2000, Windows 2003, Windows Vista, Windows 7.

**Code Repositories**

CVS, ClearCase, Git, RCS, Subversion (SVN), Visual Source Safe (VSS)

ANDREW FORWARD
203-330 LORETTA AVE S • OTTAWA, ONTARIO • K1S 4E8 •
PHONE (613) 795 5273 • E-MAIL aforward@gmail.com

**Software Development**

Chrome, Eclipse, FireFox, HttpUnit , ImageMagic, Internet Explorer 6, Internet Explorer 7, Internet Explorer 8, JsUnit, JUnit , Komodo, MiniMagick, MonoDevelop, Netscape, Notepad++, NUnit, NUnitAsp , Opera, PHPUnit , RadRails, Selenium, SharpDevelop, SimpleTest, SqlDeveloper, VisualStudio.Net, TextMate, Toad, WebTest, XCode

**Project Management**

Business Process Execution Language (BPEL), Bugzilla, Common Look and Feel (CLF2.0), Continuous Integration (CI), FogBugz, Fusebox Framework, Information Technology Infrastructure Library (ITIL), JavaDoc, Keynote (Mac), Pages (Mac), Lighthouse, Mantis, Microsoft Office (MS Excel, MS Project, MS Word, Visio), Model Driven Architecture (MDA), Model Driven Development (MDD, MDSD), NDoc, Object-Oriented Software Development, PayPal, PHPDocumentor, Rational Unified Process (RUP), Search Engine Optimization (SEO), Service Oriented Architecture (SOA), System Testing, Test-Driven Design (TDD), Traceability, Unit Testing, Unified Modeling Language (UML), Usability Testing, User Acceptance Testing (UAT), W3C, Web 2.0, Zend Framework

## VALUE ADDED SKILL SET

- ✓ Strong academic background in computer science and software engineering
- ✓ Large-project experience working with and coordinating among teams as large as 50 developers
- ✓ Ability to lead and motivate other in a team environment in English or French
- ✓ Academic and professional exposure to various Internet issues including protocols, infrastructure (LAN, VSC), browsers, plugins security, design and graphics
- ✓ Well-balanced, focused individual who is flexible to change.
- ✓ Sensitive attitude towards data and application security
- ✓ Software engineering focusing on architecture, modeling, methodology, documentation, SDLC and security for internet based applications

**ANDREW FORWARD**
203-330 LORETTA AVE S • OTTAWA, ONTARIO • K1S 4E8 •
PHONE (613) 795 5273 • E-MAIL aforward@gmail.com

# WORK EXPERIENCE

## Projects Listed in Reverse Chronological Order (Starting with the Most Recent)

## Project #13

CMS-HS (Collections Management System – High Density Component)
Client: Library and Archives Canada
Duration: September 2009 – Sept, 2010 (1 year, 2 months)
# Days Billed: 180
Position: Application Architecture
Contact: Adam Murray at 613-222-6407 / Adam.Murray@lac-bac.gc.ca

Care of Collections is an enterprise-level Intranet application dealing with multi-million record databases to provide a single point of access to the Library and Archives' collection. The system interacts with and manages information within library and archival environments as well as helping to manage the circulation, and physical management of cultural information.

The CMS-HD application extends the existing COC application framework to support the circulation and physical management of cultural information in a high density facility. The key requirements for an HD facility is loss avoidance.

Andrew Forward was involved with the requirements gathering, use case development, prototyping and application infrastructure of this enterprise-level Intranet application delivering functionality to manage the storage, preservation, and circulation of the institution's archival and published holdings in a high-density facility.

- Application architecture CMS-HD application
- Provided a technical feasibility analysis for the buy-versus-build decision
- Involved with requirements gathering and conducting a feasibility analysis for integrating said features into the existing institutions infrastructure
- Developed use cases and prototype material (MS Word and HTML)
- Managed the application environment within Subversion including merge and branch creation within the Subversion SVN repository
- Coordinated amongst related teams for merge and branch management of the consolidated code base
- Data migration and source code road-map to move the existing system to the target environment
- Transformational development, deployment and testing

- Delivered adequate documentation including road maps, requirements documents, use cases, automated test scripts and defect tracking notes
- Application development using PHP, SimpleTest, and WebTest

Andrew was exposed to enterprise cloud computing solutions for search and search indexing including SOLR and K2.

Mr. Forward used the following technologies during this project:
PHP 5.2.1 (OOP), CSS, Bash, RCS, SVN, Fusebox 3 Methodology, Oracle 9i/10g RDBMS, XHTML, JavaScript 1.6, AJAX, CSS 2.0, XML, TOAD, SQL Developer Putty 0.5, RCS 1.1, Mantis 0.18, UNIX (Tru64 v5.1), Linux (SUSE), Apache web server 1.3, Dreamweaver 8, Mozilla Firefox 1.5, 2.0, Internet Explorer 6.0, SOAP, Web Services, and iChain.

## Project #12

SiteValet
Client: Dave Hyndman
Duration: January 2009 – March 2010 (15 month)
# Days Billed: 170
Position: Senior Web Developer
Contact: Dave Hyndman at (877) 323-3444/ dave@sitevalet.com

SiteValet is a simple online tool to create, manage and host your website. It is built specifically for inns, B&Bs and small hotels. With it you can create a beautiful website in just a few hours. And it will save you thousands of dollars in design and development costs.

In summary, Andrew Forward was a senior software architect dealing with all aspects of the software development lifecycle including requirements, design, implementation, testing, deployment, maintenance, and monitoring. The application is built using Ruby on Rail, MySQL for storage, and Amazon S3 for document management. The application is deployed using Capistrano to a Nginx web server running on top of Passenger and Mongrel. The application is tested using RubyUnit and Selenium and the repository is container within an GIT instance that is also at GitHub.

In greater detail, Andrew's role on this project included the following activities.

The application domain involves a large number of raw high quality photos images. Large photos take longer to upload, require more resources to process and greater bandwidth to present to our end-users. Our review of available techniques revealed that existing photo packages did not support cropping and image scaling to the extent required by our work, resulting in evaluating existing tools and combining various approaches to achieve our end goal of presenting image processing in near real-time performance.

Related to the above, our application accepts multi-media updates (text and images). There is no standard best-practice facility to optimize the compartmentalization of the uploading process. This lead to developing several alternatives including: page-centric data manipulation using standard HTTP post-backs; component-centric data manipulation using an asynchronous communication with the server using remote-procedure calls; and, finally merging both techniques to allow direct integration of managing photo and text together (or separately, if desired).

The application accepts a large volume of rich-text to provide greater customization to our end users. Our review of available techniques revealed security flaws in the approach that could result in malicious attacks on our services. In addition, some approaches were overly strict, limiting the extent to which true *rich-text* could be achieved. And, finally, some approaches did not integrate well with our inline editing approach as described above. Following this analysis, Andrew developed custom overlays for rich text editing, including development to integrate these controls into our inline editing process and finally a more rigorous analysis of the raw rich text to ensure its content was stripped of any potentially malicious *scripting-code*.

Andrew gained experience in evaluating and developing enterprise cloud computing solutions including Amazon S3, Amazon EC2, Amazon RDS Rackspace cloudfiles, and Heroku Ruby Cloud Platform.

Andrew moved the application's image infrastructure to an online cloud storage facility called Amazon S3. The move helped to reduce the footprint of our database as well as reduce the bandwidth requirements on serving up those images. We considered two components for managing photo meta-data, AttachmentFu and Paperclip (image upload components). Neither component satisfied our needs for offline processing, requiring us to develop our own process.

Andrew experimented with mechanisms to display temporary image placeholders while the offline image processing was occurring. Each approach was developed and tested with a sample user base. The approaches included using a simple text message indicating that an image process was occurring. The feedback prompted us to develop placeholder images of the appropriate dimensions to simulate the look of the soon to be processed thumbnail.

Andrew added the ability to manage both plain text input, as well as rich text. We enhanced an existing component TinyMCE to better secure our system from *scripting-like* attacks. We experimented with various approaches to efficiently store the rich-text to enable both convenient editing of the content, as well as securely displaying it.

Andrew continued to develop the infrastructure to collect data from our three primary sources, plain text, rich-text and raw images. We incorporated the progress from previous month and continued to monitor the processing requirements of our approach.

Andrew experimented with localized in-place editing to allow finer grained control of tracking end-user data inputs to mitigate the risk of data loss. This process incorporated audit logging of all changes, and was exposed as an internal service that could be consumed by all editable fields within our content management system.

Andrew further enhanced the image processing approach to include internal tagging. This tagging approach later allowed for generic photo management controls to be seamlessly deployed along with our in-line approach for editing plain text.

Andrew optimized the server infrastructure with more processing power and memory. Because of early performance audits, our physical drive requirements remained relatively constant, despite adding several gigabytes of raw images to our content management system.

Andrew developed a back-end facility to manage the upload of several gigabytes of images using a secure channel that bypassed the web server front-end. From our monitoring metrics, this had a positive impact on the responsiveness of the application. Andrew developed a similar ability for the bulk upload of plain and rich text; all based on our existing infrastructure, but consolidated into one component for massively large edits.

Andrew improved the perceived real-time image processing by developing a mechanism to temporarily display a scaled down version of the uploaded raw images. This change resulted in additional bandwidth requirements for a small percentage of newly uploaded content. But, we were then able to reduce the amount of back-end processing from every minute to every ten minutes. This change improved the overall responsiveness of the application and helped to reduce our memory footprint, as initiating this process does result in additional memory usage and process demands.

Andrew managed client expectations, progress and defect tracking using Lighthouse App. The process allowed our team to work remotely and stay coordinated regarding which issues had been resolved, which were pending resolution and which were still outstanding. Andrew worked closely with several graphic designers to align the development of the CSS and stylesheets with the graphic design templates.

Mr. Forward used the following technologies during this project:
Ruby (1.8), Ruby on Rails (2.3.4), GIT, XHTML, JavaScript 1.6, AJAX, CSS 2.0, XML, MySQL, LighthouseApp, Nginx, Passenger, Mongrel, Max OS X, Linux (Ubtuntu 8.04), FireFox, Internet Explorer (6,7,8), Ruby Unit, Amazon S3, Rackspace, Slicehost, DNS, crontab.

## Project #11

CoC (Care of Collections)
Client: Library and Archives Canada
Duration: January 2006 – March 2007, Nov 2007 – August, 2009 (32 month)

# Days Billed: 440
Position: Senior Web Developer
Contact: Sylvie Béland at 819-994-6955 / sylvie.beland@lac-bac.gc.ca

Care of Collections is an enterprise-level Intranet application dealing with multi-million record databases to provide a single point of access to the Library and Archives' collection. The system interacts with and manages information within library and archival environments as well as helping to manage the circulation, and physical management of cultural information.

The COC application is a complex multi-million row database with multiple main modules of varying sizes and attributes. This system is intended to hold 5 million records of enterprise-critical content and currently holds over 2 million rows in a very complex structure relational database with complex relationships and large variable size text fields. Multiple languages are supported through a customized character set. There are also.

The COC application has a high number of integrated and dependent functions with multiple main modules of varying size and attributes. These modules consolidate the physical management, intellectual management and access to LAC's holdings by delivering functions of circulation and maintenance; conservation treatment; digitization, imaging, microfilming; and audio-visual preservation.

You need to add the description for the Complex Multi-Million row database with multiple main modules of varying size and attributes whose main functions were etc….in either a cultural information management setting or a library/archives environment

With over 60 users, this corporate level application provides support to primary functions and business areas of the enterprise.

The system integrates with existing and still/developed LAC components including a complex archival holdings management system and repository (MIKAN) to integrate with very complex physical management system (CoC) and very complex contact information management system (CIM).

MIKAN is an integrated Library archival management system and is also cultural information management system with over 250 users from the NCR and Canada-wide regional offices. This system is the primary corporation wide mission critical archival system used by the National Archives.

The COC application leveraged metadata and full-text search technologies, Enterprise search technologies, and Enterprise Content Management Web Services Portal technologies, as well as Client Relationship Management Electronic commerce REST (Representational State Transfer) OpenURL. The applications support Networking protocols including Z39.50 and Dublin Core. The system stores descriptive metadata about items in LAC's vast archival collection and is searched by over 250 users.

There is a web-based interface for the entire system. The system was developed using HTML, JavaScript, CSS stylesheets, Internet Explorer 6, DOM, DHTML, Apache Web Server, Fusebox, PHP, C and Oracle 9i/10g (RDBMS), including Oracle Text for keyword searching running in a HP Tru64 UNIX Clustered server environment using Storage Area Network (SAN) technology and LINUX (SUSE) environment.

The application architecture has 3 tiers: a presentation level tier consisting of the end user client's web browser running HTML, DHTML, CSS, JavaScript and DOM, a middle tier consisting of the business logic layer running PHP and C and a server tier running the Oracle database in a UNIX server environment.
Multiple languages are supported through a customized character set. CoC is a module of MIKAN.

Mr. Forward acted as a senior software developer implementing features, automated testing, manual testing, defect tracking and fixing, and 3rd party component integration via web services, XML and SOAP. Mr. Forward worked with an Apache 1.3 web server.

Mr. Forward held the same role as described below in Project #3. Andrew Forward design, developed, tested several modules within Care Of Collections including: Circulation, Physical Management, and Client Services. Andrew's main contribution included the following features: Location Maintenance, Container Maintenance, Barcode Print Services, Retrieval Requests and Renewals, and Copy Records. Andrew also created a bridge with the authentication team (CIM) and built a mechanism to tie into the authentication proxy server (iChain) and communication with CIM to control and verify access to the certain aspects of the CoC application (i.e. do you have the authorization to view this data, or perform this particular task).

Andrew introduced two testing framework into the CoC group: PHPUnit and Selenium. PHPUnit is a command line testing tool used to verify the smallest components of a system. Selenium is a User Acceptance Testing framework design to allow *record-and-playback* style testing that acts as a regression testing tool as the user level. Andrew also created a functional testing framework, WebUnit, that enabled verification of the application within its deployed environment (similar to other unit testing tools like PHPUnit but accessible through a browser as opposed to a command line). Andrew interfaced with the database, helped design caching mechanisms (both within the application and at the data level), he developed new features on request, resolved assigned defects and worked within his team to resolve issues outside of his locust of responsibilities.

Andrew was also involved with providing suggestions and implementing features to the underlying framework that supports the Amican applications. This included modifications to the client side communication (AJAX), internationalization, and communication with the authentication engine (CIM).

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address

how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward used the most appropriate technologies to deliver the web based solution. Most appropriate is subjective and decisions are dependent on existing infrastructure, corporate culture, and legacy system and developer expertise. PHP is a great platform for building web applications on a Linux / UNIX infrastructure.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as PhPUnit, Selenium, and WebUnit (custom testing framework for PHP). Mr. Forward has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward tracked, updated, reported to and resolved issues using the Mantis bug tracking system. Using a test-driven approach, most corrections are made by first introducing a failing-test that isolates the bug, and then once passing, that bug is now represented as a regression test to avoid the defect returning to the system.

Andrew Forward was involved in the software documentation process. Mr. Forward created internal software documentation to share design and architecture amongst other software developers. During weekly status meeting, Andrew would present technical progress reports to outline new interfaces, functions and coding practices available to be used by other developers. Andrew also presented a workshop on Selenium, a web testing tool, with accompanying documentation to help others improve on their testing skills.

In summary:
• Involved in full SDLC life cycle of CMS project, and integration with MIKAN for AMICAN project.
• Developed framework and strategy for Application Management department which used for all web application development including CoC
• Invovled with the systems analysis team responsible for gathering and analyzing all business and functional requirements from LAC staff. This involved reviewing existing workflow, procedures, databases structures, applications and sub-systems in order to determine the best solution.
• Evaluated existing LAC legacy technologies to determine whether they could fulfill business and functional requirements; analyzed and evaluated alternative technology solutions.
• Transformed functional requirements into technical designs, and software implementation as part of the development team.

• Ensured new technical designs integrated with existing applications and were consistent with industry and GOC technology direction.
• Modified PHP, JavaScript, HTML and Oracle procedure source code required to implement new functionality.
• As Part of the Application Framework developed routines that manage / search and coordinate user privileges for all AMICAN applications.
• Used RCS, and SVN source control commands in telnet/Unix/Linux environment to manage all PHP, HTML and JavaScript source.
• E/R Data modeled (documented data structure and developed data dictionary) and worked with DBAs to implement all necessary Oracle 9i/10g database changes.
• Roles also included unit testing, system testing, integration and regression testing of all PHP, HTML and JavaScript code and data enhancements.
• Corrected all errors in PHP, HTML and JavaScript source code as bugs were detected.
• Routine status reporting to direct supervisor.
• Monthly time reporting to senior ITS staff member.
• Data migration and source code road-map to move the existing system to the target environment
• Transformational development, deployment and testing
• Delivered adequate documentation including road maps, requirements documents, use cases, automated test scripts and defect tracking notes

Mr. Forward used the following technologies during this project:
PHP 5.2.1 (OOP), CSS, Bash, RCS, SVN, Fusebox 3 Methodology, Oracle 9i/10g RDBMS, XHTML, JavaScript 1.6, AJAX, CSS 2.0, XML, TOAD, SQL Developer Putty 0.5, RCS 1.1, Mantis 0.18, UNIX (Tru64 v5.1), Linux (SUSE), Apache web server 1.3, Dreamweaver 8, Mozilla Firefox 1.5, 2.0, Internet Explorer 6.0, SOAP, Web Services, and iChain.

## Project #10

Information Holdings Management System
Duration: May 2007 – October 2007 (6 month)
# Days Billed: 60
Position: Web Developer
Contact: Sylvie Béland at 819-994-6955 / sylvie.beland@lac-bac.gc.ca

The Information Holdings Management System (IHMS) is a complex corporate enterprise-level PHP application dealing with multi-million record Oracle databases that provides support to primary functions and business areas of the enterprise with a scalable and centralized solution for managing the holdings of eight federal record centers that store, service and ultimately dispose of records provided by government institutions. There are several dependant modules working as one whole integrated system, these modules are as follows: Accession, Disposition, Reference, Reports, Maintenance. The IHMS system is developed using PHP, Fusebox, Javascript, Oracle and LINUX.

Mr. Forward acted as a senior software developer implementing features, automated testing, manual testing, defect tracking and fixing, and 3rd party component integration via web services, XML and SOAP. Mr. Forward worked with an Apache 1.3 web server.

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward used the most appropriate technologies to deliver the web based solution. Most appropriate is subjective and decisions are dependent on existing infrastructure, corporate culture, and legacy system and developer expertise. PHP is a great platform for building web applications on a Linux / UNIX infrastructure.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as PhPUnit, Selenium, and WebUnit (custom testing framework for PHP). Mr. Forward has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward tracked, updated, reported to and resolved issues using the Mantis bug tracking system. Using a test-driven approach, most corrections are made by first introducing a failing-test that isolates the bug, and then once passing, that bug is now represented as a regression test to avoid the defect returning to the system.

Andrew Forward was involved in the software documentation process. Mr. Forward created internal software documentation to share design and architecture amongst other software developers. During weekly status meeting, Andrew would present technical progress reports to outline new interfaces, functions and coding practices available to be used by other developers.

Mr. Forward used the following technologies during this project:
PHP 5.2.1 (OOP), Fusebox 3 Methodology, Oracle 10g, XHTML, JavaScript 1.6, AJAX, CSS 2.0, XML, TOAD for Oracle 6.3, Putty 0.5, RCS 1.1, Mantis 0.18, TRU64 UNIX 5.1, Apache 1.3, Dreamweaver 8, Mozilla Firefox 1.5, 2.0, Internet Explorer 6.0, SOAP, Web Services, iChain.

**ANDREW FORWARD**
203-330 LORETTA AVE S • OTTAWA, ONTARIO • K1S 4E8 •
PHONE (613) 795 5273 • E-MAIL aforward@gmail.com

# Project #9

Mikan (Archival)
Client: Library and Archives Canada
Duration: April 2007 (1 month)
# Days Billed: 12
Position: Web Developer
Contact: Tom Morelli at 819-994-6958 / tom.morelli@lac-bac.gc.ca
Alternate Contact: Shane Lemon at 819-994-6970 / shane.lemon@lac-bac.gc.ca

MIKAN is an integrated archival management system and is also cultural information management system with over 250 users from the NCR and Canada-wide regional offices.

There is a web-based interface for the entire system, running on both the Archives' intranet websites.

The system was developed using HTML, JavaScript, CSS style sheets, Eclipse, Internet Explorer 6, DOM, DHTML, Apache Web Server, Fusebox, PHP, C and Oracle 9i/10g (RDBMS), including Oracle Text for keyword searching running in a HP Tru64 UNIX Clustered server environment using Storage Area Network (SAN) technology.

The application architecture has 3 tiers: a presentation level tier consisting of the end user client's web browser running HTML, DHTML, CSS, JavaScript and DOM, a middle tier consisting of the business logic layer running PHP and C and a server tier running the Oracle database in a UNIX server environment.

The system has several primary modules including Registration, Accession, Description and Data Conversion working in conjunction with each other. Records are registered, then accessioned and finally described in an integrated sequential work flow. This system was planned to hold 5 million records (as described in the contract) of enterprise-critical content and currently holds 1 million records in a very complex structure relational database (many entities (over 100) with complex relationships and large variable size text fields with average record length of approximately 1,000 bytes).

The MIKAN version 3 System has at least 150 named users of which there is an estimated 30 or more simultaneous users.

Multiple languages are supported through a customized character set. There are also a high number of integrated and dependent functions. For instance, a record must be registered before it can be accessioned.

This system is the primary corporation wide mission critical archival system used by the National Archives. The search engine supports various metadata standards and Networking protocols including Z39.50 and Dublin Core. Alternate search technologies such as Oracle Text and an Oracle XML cache improve searching capabilities and speed.

All records use MARC tagging as a basis for record displays and editing. Records from the database are converted to and from MARC format.

Mr. Forward worked on the integration of Mikan archival registrations within the Care of Collections infrastructure.

His role was the same as described above in Project #1. Andrew developed functionality to enable Mikan users to seamlessly create new new registration containers from within the Care of Collections application without leaving the familiar interface of Mikan. Other features that were implemented include: managing physical restrictions on container contents, activating / deactivating regisrtration containers, and assigning containers to new media (i.e. a container described as art can now also be classified as photo, if that container does contain mixed media). The features were implemented using the same technology as Care of Collections (PHP 5, Fusebox, Oracle 10g, Apache). Andrew used PHPUnit, Selenium and QaUnit for automated testing.

Andrew interfaced mostly with the Mikan technical lead, Shane Lemon and the project coordinator Tom Morelli but also dealt with the client for feedback on specific issues.

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward used the most appropriate technologies to deliver the web based solution. Most appropriate is subjective and decisions are dependent on existing infrastructure, corporate culture, and legacy system and developer expertise. PHP is a great platform for building web applications on a Linux / UNIX infrastructure.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as PhPUnit, Selenium, and WebUnit (custom testing framework for PHP). Mr. Forward has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward tracked, updated, reported to and resolved issues using the Mantis bug tracking system. Using a test-driven approach, most corrections are made by first introducing a failing-test that isolates the bug, and then once passing, that bug is now represented as a regression test to avoid the defect returning to the system.

Andrew Forward was involved in the software documentation process. Mr. Forward created internal software documentation to share design and architecture amongst other software developers. During weekly status meeting, Andrew would present technical progress reports to outline new interfaces, functions and coding practices available to be used by other developers.

Mr. Forward used the following technologies during this project:
PHP 5.2.1 (OOP), Fusebox 3 Methodology, Oracle 10g, XHTML, JavaScript 1.6, AJAX, CSS 2.0, XML, TOAD for Oracle 6.3, Putty 0.5, RCS 1.1, Mantis 0.18, TRU64 UNIX 5.1, Apache 1.3, Dreamweaver 8, Mozilla Firefox 1.5, 2.0, Internet Explorer 6.0, SOAP, Web Services, iChain.

## Project #8

EMIS (Expenditure Management Information System)
Client: Treasury Board Secretariat
Duration: May 2003 – December 2005 (32 month)
# Days Billed: 600 days
Position: Senior Consultant
Contact: Arman Mirchandani at (519) 569-9977 or amirchandani@deloitte.ca

The Expenditure Management Information System (EMIS) is a collection of systems, applications, databases and government-wide business processes that support the Treasury Board of Canada, Secretariat (TBS) in fulfilling its expenditure management role. The outputs of EMIS include Main Estimates, Supplementary Estimates and Governor General Special Warrants.

The system involved five separate but related applications. The first application was the backbone categorization of information management: the Program Application Architecture (PAA). The PAA decomposed the federal government into departments, crown corporations, strategic outcomes, programs and sub-programs (and sub-sub programs, …). The initiative also supported the idea of horizontal entities that could cut across and include several programs within separate departments. These horizontal structures allowed analysis at the same level of detail as a normal department. The second application was a detailed survey application that gathered detailed financial details, infrastructure, and personnel details from all federal departments (around 120). This information fed into the first application the Mains Estimates. Each year, the federal government releases a *Blue Book* detailing the budget for the entire government. The Mains Estimates application allowed individual departments to manage their budgets for their particular department down to the program / sub-program level (as defined by the program architecture). When ready, the Mains Estimates fed into a publishing service that was interfaced to an adobe publishing infrastructure to allow the printing of a the Blue Book. The final application developed for EMIS was the actual governmental expenditures (called ARLU). The estimates from MAINS fed into the ARLU application

whereby departments would have to report actual spending and *close-off-the-books* before the next year's budget estimates were released.

The EMIS system was a large .NET implementation that was development in multiple languages including HTML, C#, JavaScript, and interfaced with legacy publishing and database technologies. Each component of the system was used by all departments at different times of the year. The application's ability to "close the books" improved from weeks in the legacy system to minutes with the new system.

Mr. Forward's role included the design, development, testing, deployment, maintenance of a large scale web application written in ASP.Net, C#, JavaScript, SqlServer 2000 and Windows 2k3. The project involved writing automated tests using NUnit, NUnitAsp, JsUnit, and Mock objects. Mr. Forward initiated, implemented and monitored the projects continuous integration effort and automated deployment using CruiseControl.Net, Nant, Ant and Batch scripts. Mr. Forward executed performance and memory profiling using Redgate Ants and Selenium. Mr. Forward acted as a mentor and team coach to several junior developers. Mr. Forward was exposed to the entire software life cycle including modeling, methodologies, architecture principles, software engineering, testing & validation, Database management. Mr. Forward used UML modeling tools such as TogetherJ and Rational XDE to document the design to fellow developers. Mr. Forward developed, managed and verified the application's adherence to the Federal Governments Common Look and Feel (CLF) standards.

Mr. Forward worked with an Microsoft IIS 5 web server.

Mr. Forward was involved in user acceptance testing using Selenium, JavaScript, PHP and Java for 14 months during this contract. Andrew developed PHP modules to launch, monitor, track and report results on the Selenium testing using PHP. Andrew also integrated this testing into the build process by writing custom Ant tasks (written in Java) to bridge the data captured from the PHP mechanism and stored in our Clear Case repository.

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward used the most appropriate technologies to deliver the web based solution. Most appropriate is subjective and decisions are dependent on existing infrastructure, corporate culture, and legacy system and developer expertise. ASP.Net is a great platform to build full testable, object-oriented web applications, and integrates nicely with Microsoft Office products such as Excel / Word, but it must run under Windows technology.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as NUnit, NUnitAsp, NCover, Selenium, WebEnvironment (custom testing framework for Asp.Net). Mr. Forward has also developed use cases for external testers, as well as managed manual testing teams. Mr. Forward has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward has used defect tracking tools such as Fogbugz, and Rational ClearQuest. Using a test-driven approach, most corrections are made by first introducing a failing-test that isolates the bug, and then once passing, that bug is now represented as a regression test to avoid the defect returning to the system.

Andrew Forward was involved in the software documentation process. Mr. Forward has created both internal software documentation to share design and architecture amongst other software developers, as well as external software documentation to demonstrate the proper usage of the particular software system. The tools include Microsoft Word and Excel for textual and chart information, as well as Rational XDE and Together J for UML notations. The artifacts created include use cases, user stories, test cases, build scripts, deployment manuals, end-user manuals, high level architecture description, and low level design documents in UML including class diagrams, state diagrams, sequence diagrams, activity diagrams, package diagrams and deployment diagrams.

Mr. Forward used the following technologies: Html, Xml, Asp.Net 1.1, C#, Java, Php, JavaScript, VbScript, CSS, SSL, Http, Https, TCP/IP, DHCP, IIS 5, IIS 6, UML 1.*, SVN, Rational ClearCase, Rational ClearQuest, Rational XDE

## Project #7

LootXchange, eCommerce for Online Games
Client: 4079914 Canada Inc.
Duration: October 2002 – December 2005 (39 month – 7 non overlapping)
# Days Billed: 450 days
Position: Software Architect
Contact: Van Mardian (613) 729-4264 / vmardian@gmail.com

LootXchange.com (formerly PowerLevel.com) introduced a vastly superior way of selling your virtual goods such as currency, weapons and characters for multiplayer online video games. The system tracks inventory for sellers including: gold, items, and characters; and allows sellers to describe and offer services to clients including: quests, levels, and ability points. The system implemented a more comprehensive feedback mechanism to reduce the occurrences of fraud. As the system is new, users were also able to link to other feedback systems including Ebay and Player Auctions. The buyers

are able to filter through the available listings by game, location, server and/or faction. The buying process is integrated with Paypal (IPN – Instant Payment Notification).

Mr. Forward worked with an Microsoft IIS 6 web server. Andrew's roles included **developer,** designer, database administrator, business analyst, software integrator, software infrastructure and system administrator.

As a developer and designer, Andrew concentrated on the data access and business tier – helping to create an object-oriented view of the system. Andrew wrote code in C#, HTML, CSS, SQL, he designed the database scheme and created the necessary business objects to be consumed by the user interface. Andrew integrated several automated testing techniques including unit testing using NUnit, user interface testing using WebPage (a custom framework built specifically to deal with the ASP.Net infrastructure) and Selenium an user acceptance test (record-and-playback driven).

As a business analyst, Andrew worked with sellers, buyers and gamers to ensure that the system dealt with real issues of the on-line commerce as opposed to simply building a copy-cat auction site.

As a software integrator Andrew developer several scripts to automate the building, testing, deployment and database migration of the application. These scripts enabled our team to make frequent and extensive changing to the system without adversely affecting down-time when deploying new features.

As a software infrastructure role, Andrew carefully selected the appropriate technologies to build, maintain, secure and back-up the application. This included using the Microsoft .Net framework (ASP.Net and C#); JavaScript and AJAX for client side scripting, Subversion and Nant as source code control; PHP, Nant, Ant and CruiseControl.Net for daily builds; NUnit, WebPage and Selenium for quality assurance; and IIS 6, MailEnable and DNS servers for hosting, email and web address maintenance.

As a system administrator, Andrew tracked the usage of the server, planned staged deployments, tracked web site statistics about usage patterns, managed the email and webserver, and managed the various environments from development, user acceptance testing, staging the production environments.

Mr. Forward developed and prepared diagrammatic plans for web based service delivery in this project. Andrew's work involved an ASP.NET application running on Windows IIS 6 web server and tied into a MySql database, and XML configuration files. The project supported a three tier development system, Quality Assurance, Staging and Production and included back up mechanisms for the source code and the application data (i.e. the database).

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address

how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward used the most appropriate technologies to deliver the web based solution. Most appropriate is subjective and decisions are dependent on existing infrastructure, corporate culture, and legacy system and developer expertise. ASP.Net is a great platform to build full testable, object-oriented web applications, and integrates nicely with Microsoft Office products such as Excel / Word, but it must run under Windows technology.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as NUnit, NUnitAsp, NCover, Selenium, WebEnvironment (custom testing framework for Asp.Net). Mr. Forward has also developed use cases for external testers, as well as managed manual testing teams. Mr. Forward has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward used an in-house ASP.Net web application to track and manage software features, issues and defects. Based in an agile environment, this application helped to track our team's velocity (how efficient we were at delivering features) as well as identified outstanding issues on a per release / iteration basis. Using a test-driven approach, most corrections are made by first introducing a failing-test that isolates the bug, and then once passing, that bug is now represented as a regression test to avoid the defect returning to the system.
Andrew Forward was involved in the software documentation process. Mr. Forward has created both internal software documentation to share design and architecture amongst other software developers, as well as external software documentation to demonstrate the proper usage of the particular software system. The tools include Microsoft Word, PowerPoint and Excel for textual and chart information, as well as Visio for UML notations. The artifacts created include use cases, user stories, test cases, build scripts, deployment manuals, high level architecture description, and low level design documents in UML including class diagrams, package diagrams and deployment diagrams. Andrew presented the architecture to teammates, investors and potential stakeholders.

The tools and technologies being used include Asp.Net, C#, Html, Css, Sql. Andrew integrated with 3rd party websites including Paypal, Ebay and PlayerAuctions. Andrew dealt with administration issues including security, email, installation, deployment and data migration.

Mr. Forward used the following technologies: Html, Xml, Asp.Net 1.1, C#, Php, JavaScript, CSS, SSL, Http, Https, TCP/IP, DHCP, IIS 5, IIS 6, UML 1.*, SVN, Selenium

## Project #6

Use Case Editor (UCed)
Client: University of Ottawa
Duration: January 2002 – December 2002 (12 month, 9 non overlapping)
# Days Billed: 170 days
Position: Software Developer
Contact: Stephane Somé at (613) 562-5800 ext. 6714 / ssome@site.uottawa.ca

The Use Case Editor integrates a set of tools for use cases edition, domain model edition and requirements simulation. Based on use cases and a domain model description, UCEd generates a state model that realizes the use cases. The project was developed using Java on a Linux environment. The project uses IBM's SWT framework for rendering the user interface and the project incorporates JUnit for quality assurance testing.

Andrew was involved in the translating the requirements into a working prototype. The system was implemented in Java and used both XML and XMI (an XMI schema to describe UML diagrams) as a data format. Andrew developed the user interface (written in SWT) and provided some graphic design support for the various icons used in the application. Andrew initiated the testing effort by incorporating JUnit into the development process and automated the build and deploy using an open source tool called Ant. The application was deployed on the Windows, Linux and Mac operating systems.

## Project #5

The Hotels and Resorts Directory (hotels.org)
Client: BOOKDirect and Spyre Infrastructure
Duration: March 2001 – December 2001 (10 month)
# Days Billed: 180 days
Position: Web Developer
Contact: Adam Murray at 613-591-2936 / armurray@ca.ibm.com

The Hotels and Resorts Directory is designed to be the Internet's premier online directory of hotels, including providing custom-designed hotel reservation system for hotel chains.

Mr. Forward built a hotel cultural and resort amenities web application consolidating and updating over 300 amenities for over 20,000 hotels (within a multi-million record database). The directory has since grown to host over 54,000 hotels. The system was created to allow hotels managers to track every little detail about their properties so that interested guests could get know as much as possible before making their decision. The

system also featured a client portal for basic and advanced searching. The directory integrated with a sister product "Hotel-In-A-Box" which allowed hotels to manage their own website and online booking engine without having to build the technology in-house. The application was originally written using a PHP like technology, ASP (application server pages) / JSP (Java server pages). The system ran on both an IIS and Apache web server and the data was maintained in an Oracle database.

Mr. Forward developed and prepared diagrammatic plans for web based service delivery in this project. Andrew's work involved the planning the infrastructure for user interface running on Windows IIS 5 web server and tied into an Oracle database running 9i.

Andrew Forward analyzed the problems of analysts with respect to information transfer over the internet. The primary problems that Mr. Forward dealt with were to address how to minimize the information communicated between the browser and the server without hindering usability. The second problems Mr. Forward faced were minimizing the communication with the database (which in turn improves the response time to the user). Technologies such as AJAX, CSS, and HTML helped with the first issue. Intelligent software implementation and caching helped with the second.

Andrew Forward was involved in the bug tracking and fixing. Mr. Forward used an in-house spreadsheet to track and manage software issues.

Mr. Forward used the following technologies: Html, Asp, Jsp, Java, Xml, JavaScript, VbScript, CSS, Http, TCP/IP, IIS 5, Apache, Oracle 9i, Toad for Oracle.

## Project #4

Contact Management
Client: Soul Wurkes
Duration: September 2000 – December 2000 (4 month)
# Days Billed: 45 days
Position: Web Developer
Contact: Travis Burke at 416-402-9305 / Travis.Burke@infor.com

A small contact management system for a student group at the University of Ottawa. Andrew was responsible for translating the requirements into a working prototype for further analysis. The system tracked contact information for former students and the University of Ottawa and provide basic message board functionality. Andrew wrote all required code to implement the system, designed the database and managed the deployed application. The project was developed in PHP and integrated with a MySql database. We used a PhpUnit framework for testing and quality assurance.

Andrew Forward was involved with the software verification and testing of this system. Mr. Forward has used various automated testing techniques including unit, integration, functional, and user-acceptance testing using technologies such as PhPUnit. Mr. Forward

has used software development techniques such as test-driven development to promote and ensure high standards of software quality.

Mr. Forward used the following technologies: Html, PHP, MySQL, JavaScript, CSS Http, IIS 5 and CVS.

## Project #3

Optical Carrier Team
Client: Nortel Networks
Duration: May 2000 – August 2000 (4 month)
# Days Billed: 80 days
Position: Tester / Web Developer
Contact: Shawn McCormick

The OC-48 and OC-3 Multiplexers fulfill the growing need to service hand-offs in the Backbone and Core Metro space with industry leading in-service velocity and reliability. It is a high density carrier grade DS3 multiplexer for Service Providers and Enterprise customers.

Mr. Forward's role was in the testing group to build, deploy and execute the optical carriers in a staged environment to ensure the equipment met both functional and non-functional requirements. Andrew's work involved physically configured the networking hardware under different topologies, as well as configuring and testing the software running on-top of the hardware.

In additional, Mr. Forward wrote a support web application to present the results of the automated testing. The technologies included Html, CGI (Perl), Perl, Http, Xml and Apache.

## Project #2

Department Lab Teaching and Coordination
Client: University of Ottawa
Duration: January 1999 – April 1999
Duration: September 1999 – April 2000
Duration: September 2000 – April 2001 (20 month, 14 non overlapping)
# Days Billed: 250 days
Position: Lab Technician, IT Lab teacher
Contact: Dr. Tyseer Aboulnasr

Mr. Forward's instruction included an introductory course to Engineering Computation. Problem solving for engineering case studies. Emphasis is on the design of algorithms and their implementation for solving engineering problems using C.

Mr. Forward's instruction included an introductory course to data structures and file management. The concept of abstract data types. Simple methods of complexity analysis. Trees. The search problem: balanced trees, binary-trees, hashing. Sorting. Graphs and simple graph algorithms: traversal, minimum spanning tree. Strings and pattern matching.

Mr. Forward's instruction included an introductory course to computer science and program construction. Introduction to computer based problem solving for scientific applications. Design of algorithms and algorithms descriptions. 4th generation languages. Software packages. Structured program development. Modular and object-oriented programming. Program testing.

Mr. Forward's instruction included an advanced course in software quality. Quality: how to assure it and verify it, and the need for a culture of quality. Avoidance of errors and other quality problems. Inspections and reviews. Testing, verification and validation techniques. Process assurance vs. Product assurance. Quality process standards. Product and process assurance. Problem analysis and reporting.

In the teachings listed above, Mr. Forward would deliver class room instruction, as well as coordinate and lead hands-on lab work. Mr. Forward provided feedback to the supervising professor regarding the student's progress as well as insight to help guide student evaluation.

Mr. Forward used the following technologies: Java, JUnit, Unix/Linux, Windows, C, C++, Php, Pascal, and Xml.

## Project #1

Adventures in Engineering and Science / Computer Camp
Client: University of Ottawa
Duration: May 1999 – August 2000 (16 months, 4 non overlapping)
# Days Billed: 250 days
Position: Science / IT instructor and coordinator
Contact: N/A

Adventures in Engineering and Science (AES) is an award-winning, not-for-profit, bilingual educational outreach program committed to introducing young minds to the wonders and merits of science and engineering. Through the support of the Faculty of Engineering at the University of Ottawa, AES has reached over 200,000 children since its creation in 1991. In addition, AES host a summer computer camp where students learn about about animation, multimedia-enriched web sites, robotics (LEGO MindStorms, ROBIX), and how to create games and computer hardware.

Mr. Forward held two roles within the AES organization. First he was a IT instructor specializing in Software Engineering. He presented interactive science and engineering

workshops to children aged 5 to 14. He developed the computer / IT related projects for the week-long summer sessions and he delivered throughout eight 1-week summer courses.

Mr. Forward's second role was as team lead for a staff of 20. Mr. Forward also managed the registration of summer students and reported directly to the AES president.

# PUBLICATIONS / RESEARCH

***Improving Program Comprehension by Enhancing Program Constructs: An analysis of the Umple language***
Proceedings of International Conference on Program Comprehension ICPC (2009)
Umple is a set of extensions to existing object-oriented languages (currently PHP and Java) that provides a concrete syntax for UML abstractions like associations as well as certain software patterns. Umple, we argue, will help increase software program comprehension by allowing developers to describe a system at a more abstract level, and also by significantly reducing the amount of code that needs to be written and later understood.

***A Taxonomy of Software Types to Facilitate Search and Evidence-Based Software Engineering***
Proceedings of Centre for Advanced Studies Conference CASCON (2008)
Empirical software research could be improved if there was a systematic way to identify the types of software for which empirical evidence applies. This is because results are unlikely to be globally applicable, but are more likely to apply only in certain contexts such as the type of software on which the evidence has been tested. We present a software taxonomy that should help researchers to apply their research systematically to particular types of software. The taxonomy was generated using existing partial taxonomies and input from survey participants. If a taxonomy such as ours gains acceptance, it will facilitate comparison and appropriate application of research. In the paper, we present the benefits of such a taxonomy, the process we used to develop it, and the taxonomy itself.

***Problems and opportunities for model-centric versus code-centric software development: a survey of software professionals***
Proceedings of Modeling in Software Engineering (2008)
We present some results of a survey of 113 software practitioners conducted between April and December 2007. The aim of the survey was to uncover their attitudes and experiences regarding software modeling, and development approaches that avoid modeling. We were motivated by observations that modeling is not widely adopted; many developers continue to take a code-centric approach. Key findings overall include: Modeling tools are primarily used to create documentation and for up-front design with little code generation; and participants believe that model-centric approaches to software engineering are easier but are currently not very popular as most participants currently work in code-centric environments. Key findings from sub-samples include: problems identified with model-centric approaches are similar regardless of a participant's country. Programmers that model extensively (versus those that do not model much) are more likely to agree that models become out of date and inconsistent with code.

***CodeSnippets Plug-in to Eclipse, Introducing Web 2.0 Tagging to Improve Software Developer Recall***
Proceedings of Software Engineering Research, Management and Applications (2007)
Putting aside the marketing hype of Web2.0, a few central themes have emerged: user generated content, software to enable users (not restrict them), building a community and instant feedback. Many of these concepts can equally apply to rich client applications and a focus on software development teams. The initial motivation for Code Snippets was to fulfill the personal needs of software developers. At times, they need to reuse the same coding constructs, library access, method calls, and web services. Recalling (or finding) this information is Drawing on similar success as online tagging libraries like del.icio.us and flickr.com, we have developed an Eclipse plug-in called CodeSnippets that allows software teams to document, store, share and search communal code to help answer the question, "How do I code that?".

***How Software Engineers Use Documentation: The State of the Practice.***
IEEE Software 20(6): 35-39 (2003)
Software engineering is a human task, and as such we must study what software engineers do and think. Understanding the normative practice of software engineering is the first step toward developing realistic solutions to better facilitate the engineering process. We conducted three studies using several data-gathering approaches to elucidate the patterns by which software engineers (SEs) use and update documentation. Our objective is to more accurately comprehend and model documentation use, usefulness, and maintenance, thus enabling better decision making and tool design by developers and project managers. Our results confirm the widely held belief that SEs typically do not update documentation as timely or completely as software process personnel and managers advocate. However, the results also reveal that out-of-date software documentation remains useful in many circumstances.

***The Relevance of Software Documentation, Tools and Technologies: A Survey***
Proceedings of Symposium on Document Engineering, 2002
This paper highlights the results of a survey of software professionals. The survey was conducted in the spring of 2002. The results are compiled from 48 individuals in the software field ranging from junior developers to managers and project leaders. One of the goals of this survey was to uncover the perceived relevance (or lack thereof) of software documentation, and the tools and technologies used to maintain, verify and validate such documents. The survey results highlight the preferences for and aversions against software documentation tools. Participants agree that documentation tools should seek to better extract knowledge from core resources. These resources include the system's source code, test code and changes to both. Resulting technologies could then help reduce the effort required for documentation mainte-nance, something that is shown to rarely occur. The data reports compelling evidence that software professionals value technolo-gies that improve automation of the documentation process, as well as facilitating its maintenance.

***Software Documentation – Building and Maintaining Artefacts of Communication***
Master's thesis submission (2002)
Software documentation is an important aspect of both software projects and software engineering in general. In fact, documentation engineering has become a popular sub-domain in the software engineering community. Unfortunately, the current perception of documentation is that it is outdated, irrelevant and incomplete. For the most part, this perception is probably true. Regrettably, the documentation concern cannot be resolved by simply mandating more and better

documentation. This approach fails to resolve the problem as the solution ignores the fundamental goals of software engineering. The role of documentation in a software engineering environment is to communicate information to its audience and instil knowledge of the system it describes. Documentation should efficiently allow for future software development without hindering current progress. Our research focuses on the issue of documentation quality. In particular, which attributes of documentation make it effective to the audience and how can this information be monitored and parameterized to provide a better perspective about the relevance of documentation in a software project.

**Research Assistant**
University of Ottawa (Sep 2006 – March 2010)
Developed a model-oriented programming language called Umple, also available online (http://cruise.site.uottawa.ca/umpleonline).

**Teaching Assistant**
University of Ottawa (Sep 2000 – Jan 2002)
Computer and Engineering courses in English and French including quality assurance, introduction to engineering and file management in Java and C language

- Software Quality Management including the following topics:
  - Black / White Box
  - Unit, Component, Integration
  - Coverage analysis
  - Test-driven development

- C Programming including the following topics:
  - File Management
  - Introduction to Engineering

## COMMUNITY / CLUBS / ASSOCIATIONS

**Junior Achievements (2004 – 2008)**
Involved with their Economics of Staying In School program helping to education youth about education.

**Ottawa Regional Science Fair (2002)**
I judged the Junior Engineering projects with a panel of four other judges.

**IEEE Student Branch (1998 – 2002)**

- Coordinated a technical conference (S-PAC) hosting 150 participants
- Recognition for recruiting the most new student members in Canada in 2000

- Coordinated Ontario Engineering Competition OEC 2002 hosting over 180 competitors, and 50 judges.

**In School Mentoring (Sep 2000- Jun 2001)**
Similar to Big Brothers meeting with a grade 8 student on a weekly basis on school grounds.

| COMPETENCY | PROFICIENCY | LEVEL | EXPERIENCE |
|---|---|---|---|
| Government-On-Line (GOL) Initiative | Knowledge | Working | 2004 – 2010 (6 year) |
| The World Wide Web | Knowledge / Skill | Expert | 1995 – 2010 (15 years) |
| Internet, Intranets, Extranets | Knowledge / Skill | Advanced | 1997 – 2010 (13 years) |
| Mainframes/Minis/Micros | Knowledge | Working | 1999 – 2000 (8 month) |
| Database Management Systems | Knowledge / Skill | Advanced | 1999 – 2010 (11 years) |
| Architecture Principles | Knowledge / Skill | Advanced | 1999 – 2010 (11 years) |
| Software Engineering Principles | Knowledge / Skill | Expert | 1998 – 2010 (12 years) |
| System Development Life Cycle (SDLC) | Knowledge / Skill | Expert | 1998 – 2010 (12 years) |
| Modelling | Knowledge / Skill | Working | 1999 – 2010 (11 years) |
| Methodologies | Knowledge | Advanced | 2000 – 2010 (10 years) |
| Documentation | Knowledge / Skill | Advanced | 2000 – 2010 (10 years) |
| System Integration Principles | Knowledge | Working | 2000 – 2001 (1 month) |
| Testing/Validation | Knowledge | Expert | 1999 – 2010 (11 years) |
| Telecommunications Principles | Knowledge | Working | 1999 – 2000 (8 month) |
| WAN, LAN, Gateways, VPN | Knowledge / Skill | Working | 2000 – 2010 (10 years) |
| Encryption Concepts & Standards | Knowledge / Skill | Advanced | 2004 – 2010 (6 year) |
| Secure Applications | Knowledge / Skill | Advanced | 2004 – 2010 (6 year) |
| Personnel Identification & Authentication Concepts & Technology | Knowledge | Working | 2004 – 2010 (6 year) |
| Internet Security Technology | Knowledge | Advanced | 2004 – 2010 (6 year) |

| | | | |
|---|---|---|---|
| Internet Protocols | Knowledge | Advanced | 1999 – 2010 (12 years) |
| Internet Browsers | Knowledge / Skill | Expert | 2000 – 2010 (10 years) |
| Internet Browser Plugins | Knowledge | Working | 2000 – 2010 (10 years) |
| Web Servers | Knowledge / Skill | Advanced | 2003 – 2010 (7 years) |
| Web Site Design | Knowledge / Skill | Advanced | 2000 – 2010 (10 years) |
| Web Site Authoring Tools | Knowledge / Skill | Working | 2001 – 2002 (1 year) |
| Web Site Graphics | Knowledge / Skill | Working | 2000 – 2010 (10 years) |
| Website Multimedia | Knowledge / Skill | Working | 2000 – 2010 (10 years) |
| Usability | Knowledge / Skill | Advanced | 2000 – 2010 (10 years) |
| Web Site Databases | Knowledge / Skill | Expert | 2002 – 2010 (8 years) |
| Internet Security Technology Implementation | Knowledge / Skill | Advanced | 2004 – 2010 (6 year) |
| CGI, HTML, JavaScript, Perl | Knowledge / Skill | Expert | 2000 – 2010 (10 years) |