

Marlon Rabara

550 Natalino Circle
Sacramento, CA 95835
(916) 267-3315
marlon@rabara.net
<http://www.rabara.net>

Career Profile

Introduction

I have a passion for developing software that began in junior high. I continuously study the art and science of software engineering and have embraced it in my life as a career and hobby.

The format of this resume seeks to streamline the content of career information delivered to a potential employer. The [Career Profile](#) section delivers a basic overview and the [Career Detail](#) adds the intricacies of my career.

Programming Assets

- C#
- C/C++
- Java / JavaScript
- Visual Basic / VB Script
- PowerBuilder
- PERL
- PHP
- Transact-SQL
- Assembly / MSIL
- Xml, XSLT, DTD, and Xsd (Schema)

Management Assets

- Project Lead
- Project/Risk Management
- Excellent Communication Skills
- Multiple Distributed Team Management

Career Evolution (since 1992)

- Software Engineer
- Senior Software Engineer
- Technical Lead
- Application Architect
- Enterprise Architect

Career Goal

“To make a difference...” I found that the joys in my career were attained when being able to make differences in the organization that I am employed at. Using my skills in software engineering, it has come in the form of increasing sales and profit margins to dramatically cutting costs. I am the happiest in my career if the software that we build makes a positive change. Ultimately, I want to be in a position that allows me to contribute decisions that can make that difference.

Software Engineering Assets

- .NET Framework (1.0, 1.1, 2.0)
- Design Patterns
- UML
- Entity Relationship Diagramming (ER) / Data Modeling
- Microsoft Practices
- Enterprise Architecture
- Distributed/Networking Development
- Service Oriented Architectures
- Mortgage/Construction Industry

Software Lifecycle/Process Assets

- Extreme Programming
- Agile Methodology
- Microsoft Solutions Framework
- Waterfall

Education

- B.S. in Computer Science

Career Detail

The following sections are details of my career and education. Hopefully, the problems I've solved in the past can some way align with a prospective employer's own.

Employment History

MortgageIT, Inc.

2003-Current

A large-sized residential mortgage lender based in New York, MortgageIT has several channels of business under its umbrella of management. For more information, see www.mortgageitholdings.com.

Enterprise Architect

2005-Current

The primary responsibility of an enterprise architect is to skillfully align information technology and business needs. As the EA, management duties covered team members from multiple projects as well as distributed across the country. Other duties included: developing patterns and practices of enterprise design, developing guidelines for software development, performing design reviews, participate in large project planning, etc.

Enterprise Service Oriented Architectural Design

Fall 2005 - Current

Problem: The current silo approach of development architecture makes integration of services across business and department boundaries difficult to do without lengthy refactoring work. It creates inflexible enterprise architectures and does not lend it self to agility as the organization expands business functionality with new and legacy systems.

Solution: Identify key business processes and flows and derive service definitions as well as standard service level agreements. Create pure service oriented architectures and establish patterns for achieving SOA in the enterprise. Train developers across the nation on the guiding principles of enterprise architecture and perform high level design reviews for all projects submitted for development work. Develop and maintain an enterprise architecture Share Point site to govern standards in EA evolution.

Tools: Microsoft .NET 1.1 & 2.0, WSE 3.0 (authentication), Enterprise Architect 6.0, Test Driven Development, MISMO, NUnit, Web Services, Visual Studio Team System Architect Edition, Team Foundation Server, SharePoint, and XML.

Application Architect / Team Lead

2004-2005

The primary responsibility of an application architect is to design the specifications for any large scale solution. As a team lead, additional responsibilities included (but not limited to) managing and coordinating tasks for a team of software engineers, estimating development work, coordinating staff schedules, developing riskier components of a system and reporting progress for a particular project to the respective project management.

Automated Product Eligibility and Pricing 2.0

June 2005 – November 2005

Problem: The first release of the software wasn't too user friendly. It also didn't exploit the real-time pricing systems. Also, integration with e-Pass required a workflow-driven related set of interfaces that could be "plugged" into any master-page layout.

Solution: The user interface form entry needed to be simplified. The initial complex user interface was broken down into simpler reusable components that could be pieced together as a complex user interface (with more data entry fields). Pricing grid controls were consumed to provide real-time pricing views. The pricing controls were wrappers around a service-oriented base pricing API. The entire user interface was a customizable collection of

related workflow-type user controls embedded in a master-page framework. It allowed for multiple master-page designs to reuse common web user interfaces for e-Pass integration.

Tools: Microsoft .NET 1.1, ASP.Net 1.1, Test Driven Development, NUnit, NAnt, Transact SQL (T-SQL), SQL Server 2000, IIS 5.0, IMX, DataTrac, Web Services, C#, and MISMO.

Automated Product Eligibility and Pricing 1.0

October 2004 – March 2005

Problem: Mortgage lock requests were received as faxes or transmitted via e-mail. An electronic form was needed to provide a migration path from old fax and e-mail processes to a more modern on-line submission process. Also, the first cut of a vendor's implementation of a rules engine needed to be embedded as part of the workflow.

Solution: A user interface that modeled the faxed lock form was implemented as a web form. A vendor service abstraction layer for rules processing wrapped the vendor's http xml implementation.

Tools: Microsoft .NET 1.1, ASP.Net 1.1, Transact SQL (T-SQL), SQL Server 2000, ERWin, Internet Information Server 5.0, Web Services, C#, IMX, DataTrac, and MISMO.

MISMO 2.1.3 Object Framework

January 2004 – April 2004

Problem: The Mortgage Industry Standards Maintenance Organization (MISMO) has been pushing Xml definitions for standard interchange of loan documents over the internet. The organization wanted a common model for loan processing for future AUS systems, pricing systems, credit services, etc.

Solution: A MISMO-driven model was extracted from a few DTD's for AUS (automated underwriting systems). Using a software factory, a relational database was created from the schema that reflected MISMO elements and object structures. Fields were generated from element attributes, and a full execution against the software factory generator created a MISMO API with full support for MISMO AUS Xml to object-oriented model and back to Xml. Additional features built-in included parsing of Fannie Mae's Residential Loan Application format (DU 3.2).

Tools: Microsoft .NET 1.1, OrcaLogic Software Factory 1.0, DTD, and Xsd (schema), SQL Server 2000, ERWin, Web Services, C#, IMX, and MISMO AUS 2.3.1.

Team Lead / Senior Software Engineer

2003-2004

As a team lead, responsibilities included (but not limited to) managing and coordinating tasks for a team of software engineers, estimating development work, coordinating staff schedules, developing riskier components of a system and reporting progress for a particular project to the respective project management.

Wholesale Lending Website Refactoring

June 2005 – November 2005

Problem: The original website conversion was a basic ASP to ASP.Net migration. The migration didn't exploit better UI reusability patterns in ASP.Net. It also inherited the old look and feel of the site which didn't have the professionalism of a financial institution that the firm was looking for. The security mechanism was old and data for authentication was stored in an access database.

Solution: Senior management wanted a new team to redesign the website architecture. The website would be structured so that reuse of interface components could be facilitated. This was achieved by following early patterns in Microsoft's prereleases of Master-Pages and developing a 1.1 library for master-pages. After comparing the site to a few other major banks, a layout was selected that improved the professional skin of the site. To address the archaic security, the authentication and security architecture moved to integrate the enterprise user authentication services constructed in a previous project.

Tools: Microsoft .NET 1.1, Transact SQL (T-SQL), ASP.Net 1.1, Web Services, C#, and Master Page web patterns.

Pricing Matrix

March 2004 – December 2004

Problem: Pricing was offered in two parts, base pricing and adjusted pricing. It was based on information collected by various buyers of loans (investors) and then offered back up to clients in order to generate sales. The pricing collection and publishing process was a very manual process. Pricing needed to be compiled from buyer sources, the data was stored in multiple source excel documents, rate sheets needed to be generated from the source documents, multiple websites needed to publish the data found in the excel documents, and algorithms needed to be executed against the collection of excel documents to verify pricing integrity.

Solution: Various pricing sources were mapped to a standardized import facility. A win32 application was built to pull the standardized excel documents and store them into a historical base pricing system. Along with a multitude of pricing management features, the client application was able to provide views of price batches before publishing. It was also able to verify pricing integrity. Using Visual Studio Tools, an excel add-in was created to automatically generate pricing from web services designed to work with the pricing system. The same web services, provided for rate sheet generation, were served up for web consumption, and service layer libraries as well as custom server controls were designed for quick drag and drop publishing of pricing grids on the web.

Tools: Microsoft .NET 1.1, ASP.Net 1.1, Transact SQL (T-SQL), SQL Server 2000, Web Services, Visual Studio Tools for Office 2003, and C#.

Enterprise User Directory and Authentication

December 2003 – February 2003

Problem: A vast number of applications and reports have introduced their own authentication and security implementation, creating a lack of standard user maintenance and management. A handful of implementations were hard-coded user names and passwords. Active directory was not an option for internet web application authentication and existing broker management facilities in their DataTrac system was not robust enough for account management.

Solution: A data model very similar to an active directory model was implemented. A web application was developed that allowed creation and management of users, groups, organizational units, and application-based security. Enterprise user web services were exposed to allow various applications across the enterprise to authenticate and verify group membership of users.

Tools: Microsoft .NET 1.1, ASP.Net 1.1, Transact SQL (T-SQL), SQL Server 2000, IIS 5.0, Web Services, ASP, and C#.

OrcaLogic, Inc.

2004-Current

OrcaLogic, Inc. is a startup formed and founded by mortgage technology experts to build software for the mortgage industry.

CTO and Co-Founder

2004-Current

The primary responsibility of a CTO is to determine the organization's technology strategy and vision.

RulesLogic 1.0

December 2005 – Current

Problem: Plans for building a smart MISMO enterprise service bus, require some advanced rule processing. The rule grammar is much more complex than offered by existing vendor systems.

Solution: Design a rule framework. The rule framework will support the complex grammar by having intelligent grammar parsers and mappings to rule syntax. The test-driven approach is to be used to flush out API design and exploit the benefits of a test-driven agile approach.

Tools: Microsoft .NET 2.0, Win32, C#, NUnit, Test Driven Development, T-SQL, and SQL Server 2005.

Software Factory 2.0

July 2005 – December 2005

Problem: Many features of the first version of the software factory were designed based on the .NET 1.1's limited technology (synchronous ADO, base classes for generation, custom development to support nullable types, complex custom collections).

Solution: Exploit new features of .NET 2.0 by refactoring software generation algorithms to use: asynchronous ADO, partial classes for generation, the use of the Nullable generic type, generic collections in place of custom collections, and many more optimizations.

Tools: Microsoft .NET 2.0, Win32, C#, T-SQL, and SQL Server 2005.

Software Factory 1.0

January 2004 – January 2005

Problem: One of the most tedious aspects of developing solutions is to build and maintain the model (object-model). As the corresponding data model changes, it does a ripple effect of changes through the data tier components (stored procedures, etc.) as well as through both the DAL and object model assemblies.

Solution: The organization decided to build a software asset for internal use and possibly for sell to other development shops. The software asset would be known as the software factory which analyzes a properly designed object oriented database design (OOD) and completely automates the generation and maintenance of the system framework, including stored procedures and functions at the data tier. It allows for code maintenance by structuring the class generation in such a way that base classes can be overwritten, preserving custom additions to the API or library. Native value types that were nullable also had built in support by the factory to generate a nullable type (storing a bit flag to determine if a class' attributes had a value or not).

Tools: Microsoft .NET 1.1, Win32, C#, T-SQL, and SQL Server 2000.

Capitol Commerce Mortgage Company

2002-2003

CCMC is a large-sized residential mortgage wholesale lender based in Sacramento, California.

Team Lead (Senior Software Engineer)

December 2002- August 2003

The primary responsibility of a team lead was to coordinate development with a team of engineers, meet with the IT director to go over projects and priorities, update project management, define development standards and practices, and review designs and code completed systems.

Secondary Marketing Rules Engine

March 2003 – August 2003

Problem: On-line loan pricing required two parts, a base pricing repository and a rules-based adjustment engine to process loan level adjustments to loan pricing. While the organization

had a base pricing repository, all loan-level pricing was calculated manually. In moving to an on-line system, there was a need for automated and historically accurate rule definition (changes to pricing rules over time).

Solution: Three primary components were developed: a win32 rule management and test interface for the secondary marketing department, a rule framework and API, and a historical database model to support persistence for the framework objects.

Tools: Microsoft .NET 1.1, Win32, Transact SQL (T-SQL), SQL Server 2000, and C#.

Senior Software Engineer

August 2002- December 2003

The primary responsibility of a senior software engineer is to provide expertise in software development by creating software designs and developing algorithmically correct software code. It also included being a mentor to junior level engineers.

D3 (Pick) Data Provider

August 2002 – December 2002

Problem: A legacy database system prevented the organization from providing on-line services. There was no API or connectivity from modern programming environments such as .NET.

Solution: Since the database resided on a UNIX system, a socket based service was developed that processed SQL query like syntax and returned result sets back to the calling service. A .NET wrapper API was built to abstract the service TCP/IP interface which simplified consuming and building data applications for the legacy system in .NET.

Tools: Microsoft .NET 1.0, C/C++, D3/Pick, UNIX, Windows 2000, Window Services, TCP/IP, and C#.

HonBlue, Inc.

1993 - 2002

Hawaii's largest and most technically advanced reprographic firm (<http://www.honblue.com>).

Development Lead / Senior Software Engineer

1996 - 2002

The primary responsibility of a senior software engineer is to provide expertise in software development by creating software designs and developing algorithmically correct software code. It also included being a mentor to junior level engineers. As a development lead, it included managing and coordinating development activities with junior software engineers.

JobTracking Web Application

July 2001 – February 2002

Problem: A previous version of the software was a Win32 application and as updates were made to the application, hundreds of machines needed the new software updates. This didn't make the project agile as new features were added to the system continuously.

Solution: Management wanted to implement a web-based version of the same Win32 application. With the 1.0 release nearing final, a prototype was designed in Asp.NET to determine feasibility and later modules were implemented using .NET technology.

Tools: Microsoft .NET 1.0, ASP.Net 1.0, SQL Server 2000, and C#.

JobTracking Win32 Application

January 1999 – August 2000

Problem: Numerous jobs were sent to the company and tracking their progress through production was a very manual process. The delivery system was a collection of clipboards and delivery slips. Department productivity and load couldn't be efficiently analyzed for human resource needs on particular days and even during the days.

Solution: A 4GL tool was selected by management as a development platform. After requirements were captured and analyzed, a modular approach was the strategy for

developing the object-oriented client-server system. Also, the system integrated with bar-coding products and technologies to stream-line tracking of a job through its workflow. Productivity dashboard-type reports were provided to allow management to determine production load and make quick decisions on resource allocation at any point in time.

Tools: PowerBuilder 6.5, Install Shield (deployment software), Workflow Patterns, OO design, ERWin (data modeling) and SQL Server 7.0.

SalesLogix Implementation and Customization

March 1997 – June 1998

Problem: The organization's contact tool was ACT. As the sales staff increased, a need for maintaining sales relationships in a more robust data environment required an enterprise solution.

Solution: An enterprise CRM solution was selected. Having a custom programming interface for extending functionality, aspects of the system were enhanced using these capabilities.

Tools: SalesLogix, SQL Server 6.5, Visual Basic, and VB Script.

Software Engineer

1993 - 1996

The primary responsibility of a software engineer is to provide expertise in software development by creating software designs and developing algorithmically correct software code.

Archiview (Building Maintenance Imaging System)

August 1996 – July 1997

Problem: As popularity of the Blueview system grew, construction document storage was also quickly becoming a growing need. Building management (customers of HonBlue, Inc.) needed to be able to store their building documents for long periods of time and be able to retrieve them for plumbers and other maintenance staff.

Solution: A variant of the Blueview system was developed for building management to store engineering size documents. Large construction documents were scanned in using a large scanner and then indexed with the software system. All that was needed was a compact disc and images could be pulled onto the screen in a matter of seconds.

Tools: Visual Basic 4.0, CD Burner, Diamond Head Imaging OCX controls, and TIFF file formats.

Blueview (Legal Imaging System)

July 1996 – September 1996

Problem: The organization began to enter the digital reprographics side of the business and started with the legal arena. Many banker boxes of legal documents made it a pain for lawyers to go to court with and to do research with.

Solution: The banker boxes of legal documents were scanned electronically. Each scan was run through OCR engines so that the text could be identified and quickly mapped to each image. Hundreds of thousands of single documents in many banker boxes were reduced to a collection of compact discs. A win32 interface was built to link the indexed text to the image and provide annotation tools (and many other features). It was integrated with popular litigation software systems like Concordance and Summation via DDE (dynamic data exchange).

Tools: Visual Basic 4.0, OCR, Text Indexing, CD Burning and Archiving, Diamond Head Imaging OCX controls, and TIFF file formats.

Project Hold System

February 1993 – July 1994

Problem: The organization serviced companies from the construction industry by storing plans for various building projects and reproducing blueprints for interested parties. The

storing and tracking of these documents were a very manual process with binders and pages of information. It was very easy for a page to be torn out and notes about the document status lost.

Solution: A system was built to store basic information about the project. Other information such as document and revision tracking was put in place. Users were able to find out who had their project documents, the state it was in, and correctly track changes to the project over time.

Tools: Microsoft FoxPro 2.6 for Windows

Education History

University of Hawaii, Manoa

June 2000

B.S. in Computer Science

Below are highlights in pursuing a computer science degree.

Student Assistant Instructor

Database Design and Modeling

With my real world experience, I approached the instructor with my concerns with the content he structured for the semester and commented on projects he assigned in previous semesters. Being thoroughly impressed, he offered me an A-grade for the semester if I assisted him with lecture reviews and grading student projects. Eventually, I began performing lectures on database development and model optimization which I enjoyed doing. It helped build my mentoring skills that I believe make me a strong lead and manager today.

Team Lead

ICS Website

In one of our senior projects, I was selected as a lead by our group as I had previous experience managing teams and as I was looked at as a natural leader.

Problem: The University's website for information and computer sciences was static and rather dull. Students of the program rarely visited the site because of the lack of features and teachers rarely felt the pressing need to put content up because of lack of student visits.

Solution: Human interaction studies were done and navigation prototypes were built. A dynamic web page rendering technology, PHP, was selected in order to provide the speedy dynamic rendering of pages. A fresh and very attractive layout was introduced as well as features to attract more visitors (student bio, employer job posting, etc.).

Tools: Sun Solaris SPARC, Apache, PHP, JavaScript, and MySQL.

Fun Facts: Since 2000, the website in its original design layout still remains (<http://www.ics.hawaii.edu>).

References

Available upon request...