2011

University of North Carolina Wilmington

Master of Science in

Computer Science and Information Systems

Proceedings

https://csbapp.uncw.edu/mscsis

A Centralized Integration Interface Repository

Micah Justad

A Capstone Project Submitted to the
University of North Carolina Wilmington in Partial Fulfillment
of the Requirements for the Degree of
Master of Science
Department of Computer Science / Department of Information Systems and Operations Management

University of North Carolina Wilmington
2011

Approved by

Advisory Committee

Dr. Clayton S. Ferner

Dr. Bryan Reinicke

Dr. Devon Simmonds, Chair

Dean, Graduate School

Abstract

Public safety technology is rapidly evolving today with enhancements and innovative techniques to current practices that make public safety virtually automated. Some of these include GPS positioning, computer aided tasks, inmate management, and automatic police and fire dispatching. Along with these automated systems comes some operational overheads such as locally configured and maintained databases, employee training, and product maintenance. The agencies that interact with and through public safety companies communicate with each other several means for example, through sharing jurisdictional data or helping expand county and state databases. This communication is accomplished through the use of 3rd party products. As a result agencies use an array of different products, and these products need to communicate with each other. This presents the companies with the difficult task of integrating different systems. The central product of this integration effort is an interface. This project presents a solution to the interface integration problem in the form of a centralized integration interface repository for a public safety company. The new repository and associated software were integrated with other databases in order to allow integration analysts to explore historical interface data, reuse previous documentation in the interface process, and help in the development of standardized universally compliant interfaces. Results and lessons learned are presented.

Table of Contents

Chapter 1: Introduction

Public safety technology is rapidly evolving today with enhancements and innovative techniques to current practices that make public safety virtually automated. Some of these include GPS positioning, computer aided tasks, inmate management, and automatic police and fire dispatching. Along with these automated systems comes some operational overhead; such as locally configured and maintained databases, employee training, and product maintenance. These business functions are beneficial, because these systems help public safety agencies and their employee's accomplish their jobs faster and more accurately than ever. The number of products being used out in the field isconstantly increasing. For example,a single client such as a law enforcement officer could usefour to five different products while on patrol. The main systems used today include:

- **Record Management System** (RMS) – The RMS is considered the heart and soul of agencies information technology, all other systems within the agency communicate with it. This system acts as a database for all public safety related incidents. The incidents are saved in the RMS as records with specific attributes depending on the particular incident. These incidents can be anything from a traffic violation to a 911 call, regardless all incidents must be recorded for internal organization, employee management, and statistical reference. The incidents can be accessed from any terminal within the agency or remotely via

patrol car terminals or laptops. This helps share the data within the whole agency instead of making the headquarters an informational bottleneck.

- **Computer Aided Dispatch** (CAD) –911 call takers use the CAD system to help them record calls and dispatch officers. The CAD system is usually integrated with the RMS system in order to enter new incidents; these incidents are saved within the RMS for later reference.
- **Mobile Positioning**(GIS) – The addition of GPS has allowed agencies to pinpoint officer locations while on patrol. When integrated it allows the CAD system to dispatch specific officers to particular incidents based on location; lowering incident response time and making their job more efficient.
- **Field Based Reporting** (FBR) –The FBR system allows officers to update incident information remotely from their cars. This will update the RMS within the agency making information real time.
- **NCIC Query Control Panels** (NCIC) – In order for officers to check background information on victims or arrestees, they need to query the National Crime Information Center (NCIC). The NCIC system is actually a combination of all records within the US; creating a super RMS.

These are just a few of the systems being used today by agencies such aspolice departments, fire houses, EMS stations, etc. These agencies and the systems they manage all share the same thing; information.

The Interface

This information is constantly created due to the high paced public safety field. In order to benefit from the mass amount of information, businesses must leverage existing practices to gather, store, and effectively use all their data. Thisincludes historical analysis and predicting future behavior. All agencies house a database and the applications used to read, write, and query these databases. Due to the amount of different systems developed by many different companies, there is a high demand for integrated interfaces. Each interface is defined by a description of the information required to perform a task for which the interface is created and a description of the information that the interface provides to users. These interfaces handle the communication between different systems within an agency, the systems can be declared as providing, receiving or both; indicating the flow of information. Interfaces within public safety agencies all have a similar structure. Each interface is defined by The required information is usually incident information from Field Based Reporting (FBR) or Computer Aided Dispatch(CAD) systems. An example of an interfaceis shown in Figure 2. The figure shows the interaction between a CAD system and a RMS database where pertinent information is moved from the CAD system into the RMS database for record keeping. This is the most common interaction within agencies. There are three main parts to the common interface. The first is the RMS system which serves as the data center which therefore requires information. The second part is the information provider, for example, the CAD system. The final functional part is the data transformer which serves as the work horse. The data from other systems will need to be transformed and mapped into the new system's database. In order to do this there needs to be a function which takes the information from the provider, performs business logic for data

translation, and loads the new data into the system. An example of this interface is portrayed in Figure 1.The figure shows the required functional work flow for the communication between two systems (RMS and CAD).

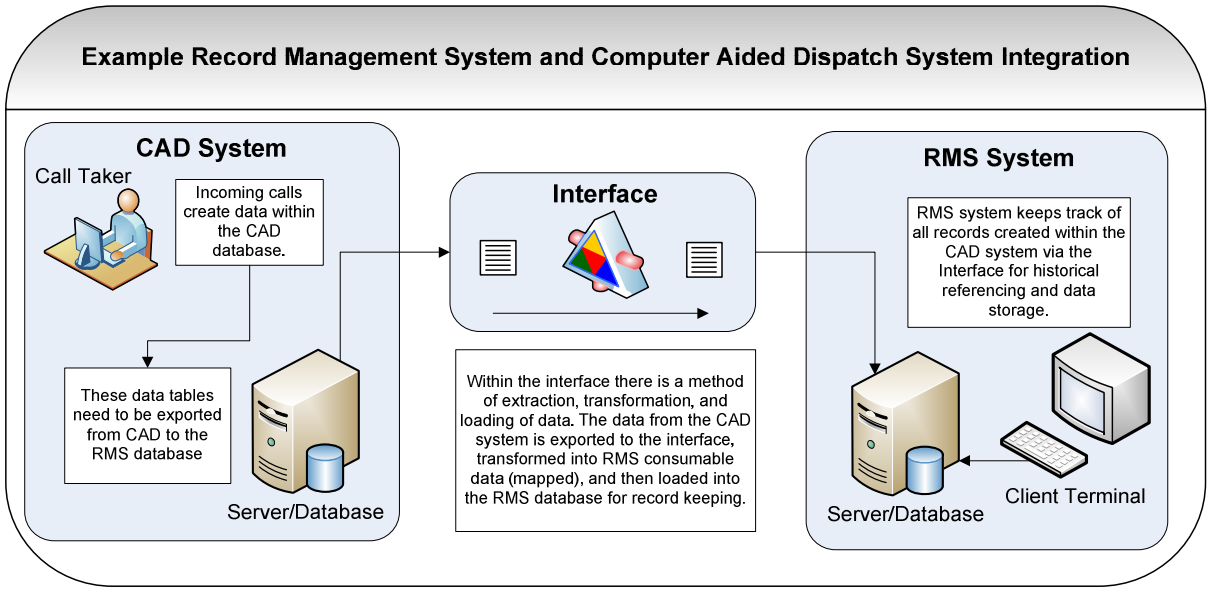


Figure 1: RMS and CAD Integration Diagram

The basic functions of an interface include (see Figure 2):

- **Extraction**- This step involves the gathering of information from the providing system, this can be done through many different methods such as: web services, direct database connections, file transport protocol (FTP), etc. Often this information is in a different format from the system receiving the information and needs to be transformed.

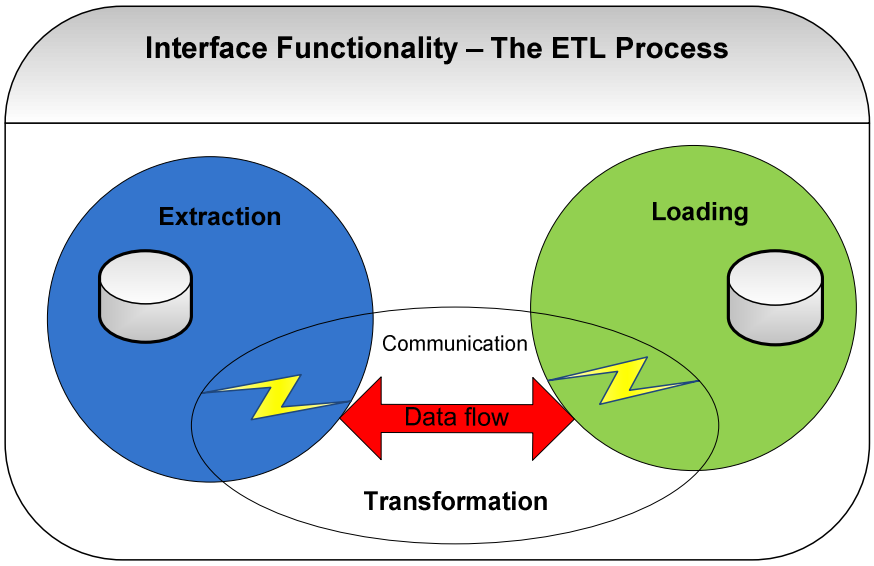


Figure 1: Interface Functionality

- **Transformation** - The transformation process takes the extracted information from the providing system and converts it into an acceptable form, most likely XML. Most companies strive to make XML a standard with integration but due to the older legacy systems and customer request there is a need for other methods.
- **Loading** - The last responsibility of the interface is the loading of the data into the receiving system. This is usually painless for the company developing the interface because of the previous step; all data is already in their format. The following figure illustrates the flow and communication of data within this three function process.

The extraction, transformation and loading (ETL) process is very common in information dependent companies. There has always been a need to integrate and share data between databases. Just like the need to store and reference historical databases within acompany, there is also a need to store and reference integration interfaces within an integration department. This can be accomplished through the use of a centralized interface repository.

The repository keeps track of all interface process related information and allows a user to data mine and leverage historical interface information.

Problem Statement

The agencies that interact with and through public safety companies either communicate or will need to communicate with each other, whether through sharing jurisdictional data or helping expand county and state databases. This communication is accomplished through the use of 3rd party products. In most instances a 3rd party will need to create a custom solution and an associated interface for a particular application. Sometimes a 3rd party will need to leverage older resources to minimize the effort of creating new interfaces, but rarely do they use the same interface twice on different systems. Interfaces that can be used on different systems are called universalinterfaces. Universal interfaces are very valuable to public safety companies and their clients because they do not require any development or agency specific manipulation within the product. The development of universal Interfaces is strongly becoming more popular within businesses due to its high return on low investment. Even though the reuse of interfaces is key from the business standpoint, there is little effort to research the development of these within companies. However companies do reuse interface documentation and business processes related to the creation of particular interfaces. The reuse of business process documents reduces effort put forth by the company on the development and sales process of these custom solutions. Whether it's through a custom solution or a universal interface, interfaces are widespread and proper management of interfaces is central as companies seek to manage the difficult task of integrating different systems.

Proposed Solution

There is a need for a central repository with an associated storage/access process for all custom solutions and Interfaces including documentation. To accomplish this goal, this project consisted of an application of which was developed and will eventually be deployedfor interface storage and manipulation. Thesystemcontains its own local database for interface indexing fields (name, type, product, etc.) and also communicates with an existing human resources database (Customer Relationship Management System, CRM) in order to reference historical and current interface data. Through the use of this repository the staff will have custom solution and Interface data mining; ensuring data, documentation, and product reuse. Figure 3illustrates a high-level architectural overview of the system to be built.

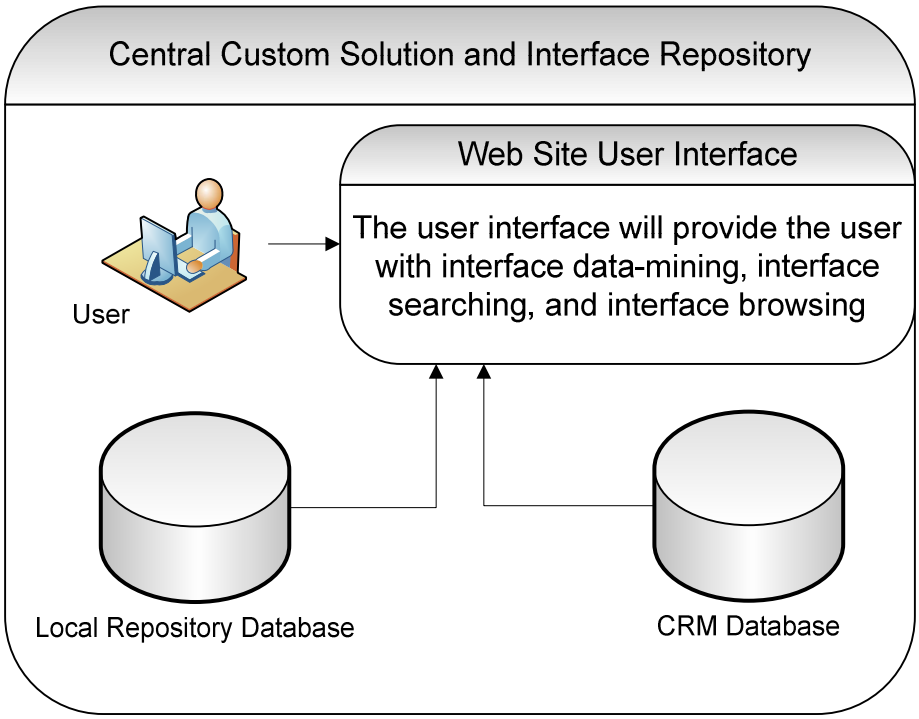


Figure 3: System Architecture

Solution Architecture

The central repository's architecture consists of three parts: the user interface which resides on an internal server, a local database (Interface DB) which in this case holds "custom interface" information, and an external CRM database. The user interface's goal is to present data and allow the user to search through that data for specific information. The local database named ‘interfacedb’ contains information specific to only custom interfaces, this separation from other departments helps compartmentalize the knowledge. Knowledge of which will be extracted and added to, through the use of database connectivity to the CRM and local database. Figure 4 is an abstract layout of the solution architecture.

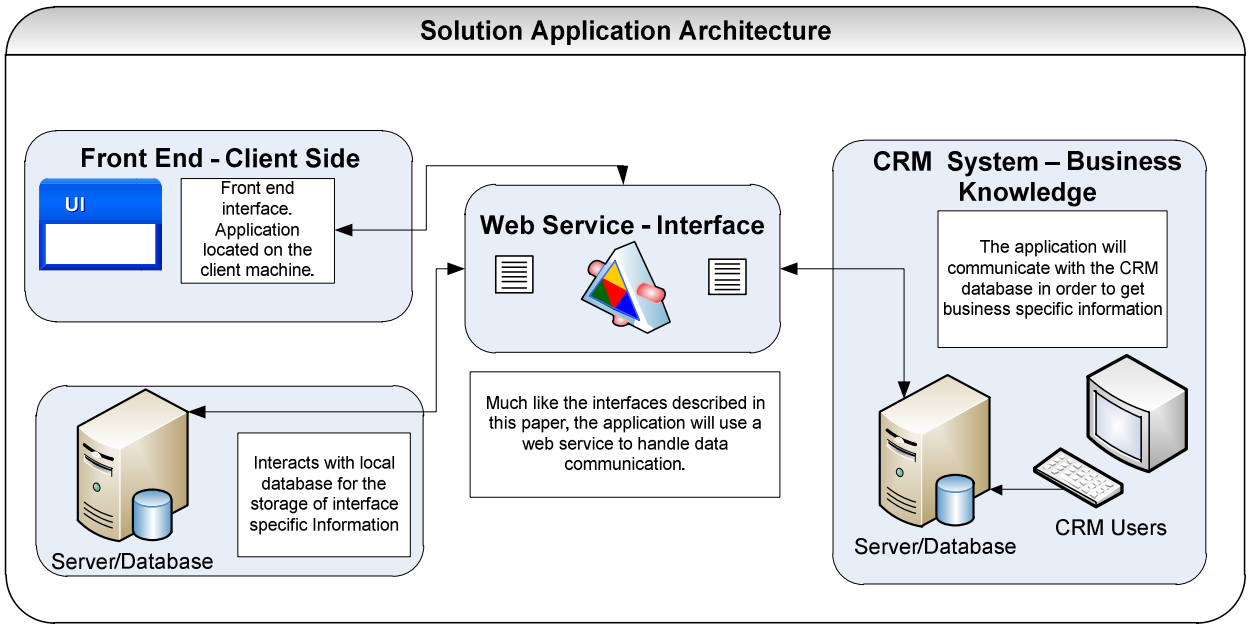


Figure 4: Solution Application Architecture

Contribution

The developed productwill serve as the front end of all interface development within the business using it. It gives product managers and integration analysts a wealth of

information at their fingertips through a front end user interface. All of the information relates to diverse custom interfaces and hopefully will eventually contribute to interface reuse. Through long-term use the interface repository will grow in size making it very valuable to the business. Having a central location for "custom interface" specific information is useful for historical interface lookups, and code/document storage. Also having a central location for custom interface knowledge will reduce the risk of previously product managers leaving. All of the information is within the repository, meaning there will be no way to lose valuable information or documents if an employee decides to leave or is terminated. Eventually, through the reuse of the information,the business will be able to leverage the shared knowledge to standardize interfaces. At least at a high level, then gradually work their way toward a universally accepted interface standard.The rest of the paper is organized as follows:

- Chapter 2 describes background and related work including the evolution of public safety automation; CAD, RMS, etc.
- Chapter 3 describes the research method taken to resolve the issue at hand.
- Chapter 4 details the research results.
- Chapter 5 presents a discussion of the research finding and implications.
- Future work and conclusions are presented in chapter 6.

Chapter 2: Background and Related Work

Automated Public Safety Evolution - CAD Systems

Computer automation and the systems that support it have evolved substantially within the last decade. It has become a commodity within public safety and public/private transportation services; therefore the need to develop and maintain these systems has increased. For example the CAD system, a CAD system is a generic family system that can provide automatic dispatching of the requested tasks within their critical timing requirements. [3] The operational workflows of most systems follow a similar structure as the figure below.

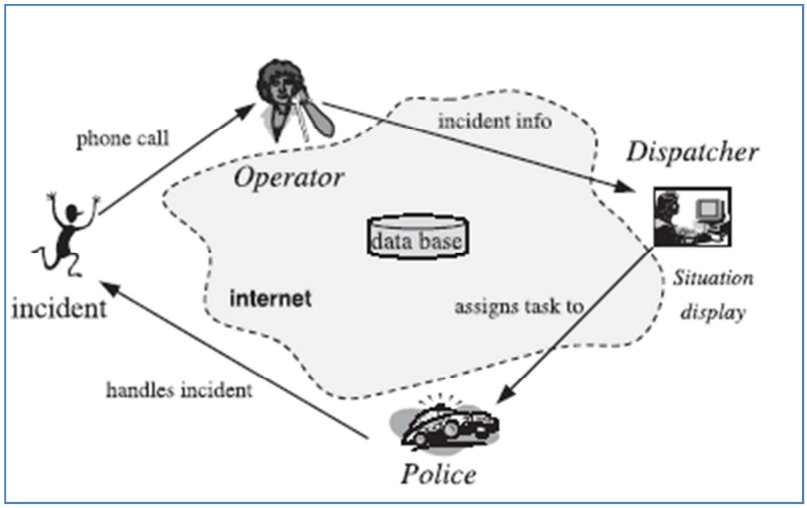


Figure 5 Operational scenario in CAD system for Police [3]

There are many different CAD developers in the field today, many of which got their start in radio and call systems. With the growing market these vendors need to distinguish themselves and offer competitive core competencies; as a result new innovative techniques are created. Some examples include:

Omnitronic5's IPRdispatch combines SIP with other technologies that have been tailored specifically for land mobile radio, including RTP/RTCP and the company's proprietary digital audio conferencing technology. Together, these enable VoIP connections to remote radios to be established and managed. [5]

Zetronoffers many CAD-compatible products for integration at the workstation. For 911 call-taking, Zetron's CAD interface sends ALI information to CAD and also sends TDD messages. In addition, the company's radio dispatch consoles support external CAD control of channel selection and voice transmission. [5]

VPI's Capture Pro call and radio recording system features the company's Fact Finder CAD screen analytics tool, designed for optimized compliance and risk management, incident evidence reconstruction, incident response, quality assurance and training. The system simultaneously classifies calls by corresponding CAD data and events — case number, incident type, ID, and more — extracted directly from CAD operator screens. [5]

These vendors along with VisionAIR are setting the standard in automated public safety. But how do these systems fit and perform within an agency? For example a deployment of automated dispatch technology can profoundly impact the operations of an agency from out in the field to the generation of the work schedule. An excerpt from an article on the deployment of a CAD system (MOM):

"The system will be in use by troublemen, T&D line crews, electricians, line maintenance inspectors, line clearance inspectors, servicemen and first-class meter installers. Total, some 700 vehicles will be outfitted with the MOM system, and some 1,300 employees

will use it to receive and complete their daily job assignments. These jobs will include trouble orders, construction jobs, maintenance jobs, meter orders, underground locates and collection orders. In the office, managers and support staff can view jobs, track the current status of each, move work among crews when necessary and optimize the schedule if needed—all from a single application." [4]

CAD systems prove to be worthwhile for the agency and the people they support. The same is true for record management systems, RMS systems are becoming the core system of most agencies. This is because all systems within the agency and surrounding agencies deal with information. This information is valuable to the agency in order to keep track of previous statistical incidents and even trend and predict patterns.

RMS Systems

The idea of the RMS system has been around for as long as the traditional database. An RMS system is essentially a central database that houses and manages all data related to an agency relinquishing the need for hard paper files. The RMS system is usually integrated with the CAD system within the agency, resulting in centralizing incident, personnel and training, and equipment inspections information. The system allows dispatchers to respond to and mange citizens' requests for fire and rescue services more efficiently. [15]

Incident data isn't the only type of data managed by the RMS; it also stores public documentation which can be shared online to the public. These documents can be internal as well, such as financial records or court citation documents. Using RMS systems for document

storage can be very beneficial as long as there is a strict procedure and control process. [13] Paper documents are being phased out, on average 65 cents of every dollar spent to house records is wasted. Record keeping is only 35% efficient, and the overwhelming amount of paper work to be filed makes mining for information a difficult task. [14]

In order to keep up with the market, RMS systems today need to keep paper documents in the form of PDFs as well as the incident data in order to utilize the true benefits for the agency. Also documents need to be stored and indexed in a way in which a user can retrieve data or whole documents using simple filters. This is also a benefit of the central repository discussed within this project.

Component Based Engineering Background

The interfaces created in order to integrate and share data between systems such as the CAD and RMS systems are crucial to agency operation. The CAD system interfaces with the RMS system in order to store data coming in from 911 incidents and other sources; this data is used when dispatching units (fire or police) to the scene. The handling of the data is controlled by the interface; therefore the interface is the component in which these systems communicate. Component based software engineering is strongly related to and can be compared to the development and life cycle of these interfaces.

A component is a coherent package of software implementation that can be independently developed and delivered, has explicit and well-specified interfaces for the services it provides, has explicit and well-specified interfaces for services it expects from others,

and can be composed with other components. This basically means a component is anything you develop that acts as a 'middleman' between data sources. The component must know what it needs as input and what it needs to deliver as output. In order for the component to communicate with its environment, it needs ports. Ports are what specify a distinct interaction point between the component and its environment. [2]

All interface and component engineering share the common goal for standardized models through the use of meta-models or template interfaces or components. There are many challenges when trying to develop reusable interfaces or components. One is the different methods for data processing are introduced between the new interface and the generalized base code. This adds complexity to the structure and makes the interface's code grow depending on the interfaces functional capability. This increased complexity will make the code fragile and become problematic later on with code evolution. But on the other side; too much generalization can cause unintentional execution through overlapping methods and break the program's semantics. Finding the perfect balance between precision and generality is a focus of CBSE. [1]

There is a need to develop component meta-models which can serve as general interface templates. These templates serve as the foundation for interfaces, through using these the developer can cut down on work done and overall cost. The central repository has this goal in mind and can achieve it through the use of the database integration and a rich internet application user interface.

Interface Creation Workflow Process

The process from client contact to interface conception involves many steps and I believe some steps can be more efficient and even mitigated through the use of a central repository. This can be accomplished through methods of knowledge sharing; resulting in a faster more efficient process for these conceptual interfaces. Below is a flow chart showing this process from client request to the last step of deploying the product.

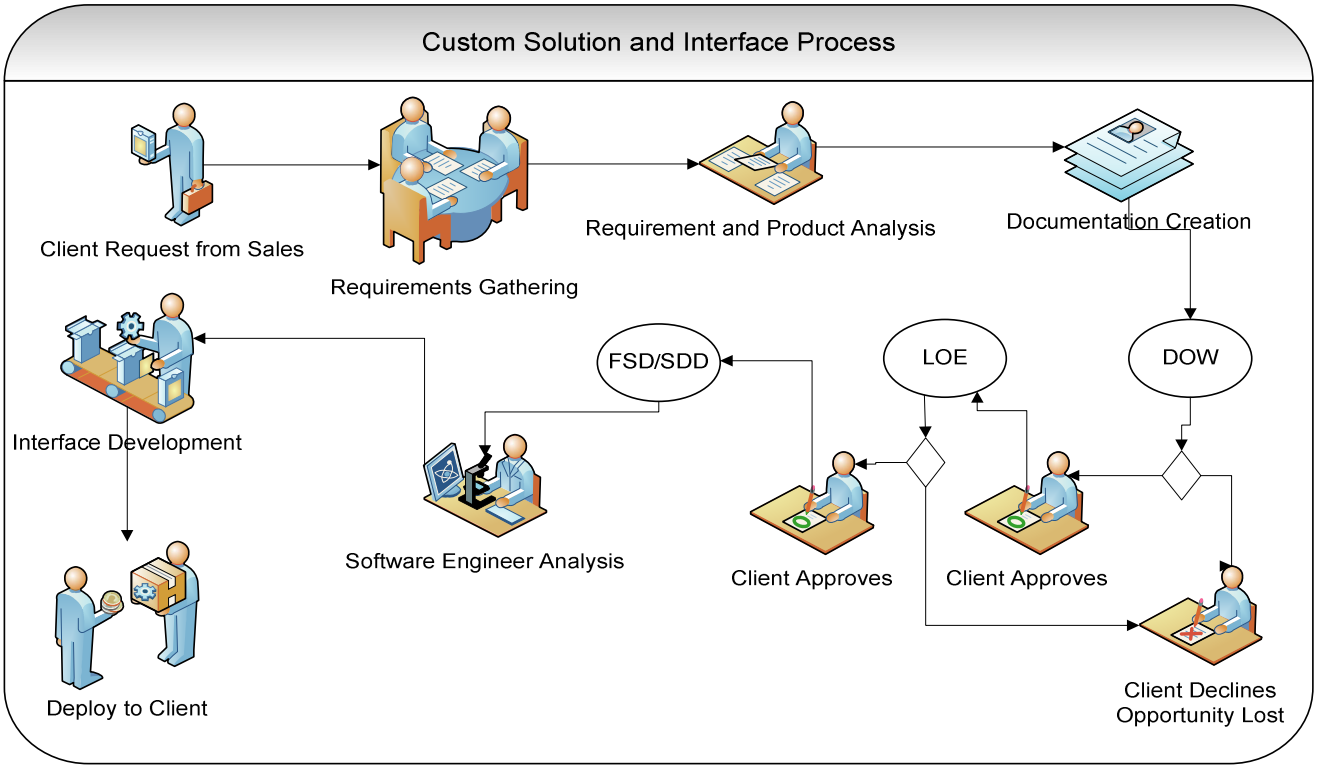


Figure 7 Custom Solution and Interface Process

Custom Integration Interface Type Definitions

There are two different types of interfaces within the company that will be discussed within this paper; the type depends on the use of the product and its resell-ability.

"Productized" Interface – this is an interface that requires no customization to either the client's product or the baseline 3rd party product; this is accomplished through standardization. This should also be considered the perfect solution to an interface quote, due to its reusability.

Custom Solution – this is a custom solution interface that is a "one-off" creation between two systems. Like all integrated interfaces, all custom solutions have potential to become productized interfaces.

Custom Solution and Interface Process

The following is an example process in which an interface gets created; the repository will make these steps more efficient.

Client request from sales - The client starts off by contacted the sales department at the participating 3rd party. They setup with an account manager if not done before hand; after which they discuss their requirements with the sales staff.

Requirements gathering - The sales staff is usually not the most technical savvy staff; but due to their people skills they get first contact with the client. The end development engineers hardly get any client interaction so it is imperative that the sales are efficient with their time with the client. The sales department will try to get as much info out of the client as possible; pertinent information includes 3rd party information, product information and a functional work-flow including interface specific requirements (transmission type, file format, logging, etc.). All the requirements will determine the interfaces functionality, ETL.

Requirements and 3rd Party product analysis - After the initial sales call, the requirements gathered are forwarded to the product management staff, here they analyze the current work-flow and requirements and determine how to integrate with the 3rd party in the most efficient yet economic way. Sometimes the workflow can be tweaked in order to use one of the "universal" interfaces or maybe an old interface, this can be determined through comparing the requirements to interfaces within the repository.

Document Creation

Create a DOW - Once product management is confident with their work-flow breakdown and functional analysis they will create a DOW. This is a Description of Work document that explains what engineering work needs to be done; it is high level in nature. When completed it is then forwarded to the client for verification. When verified and signed the document is sent to a senior engineer for a calculated man-hour estimate. The creation of the DOW takes man-hours as well, therefore certain parts of old interface DOWs should be reused. The repository hopes to boost the reuse of documents through potential storage and file retrieval of these process documents.

Create LOE from DOW for client - The Level of Effort document is an extension of the DOW; it calculates the man-hours needed for all labor, this includes all testing, development, etc. based on the hours specified by the LOE, there is a final price for development included with the final LOE document for the client's acceptance or declination.

If approved for development - Create the FSD, SSD. If a client agrees with a LOE, the software engineers and development team will create a more in-depth version of the DOW. This Functional Specification Document includes low-level details on functionality from the clients stand point. The Software Specification Document specifies the software's functionality from the developers stand point. Just like the DOWs these documents can be reused within the company after some requirement tweaks and revisions to old interface documentation.

Develop/Deploy - After the FSD and SSD are created, development finally occurs. This process can take anywhere from 6 to 18 months based on the integration and number of products involved.

VisionAIR Company Information

VisionAIR is a public safety software company located in Castle Hayne, NC; it was established over 20 years ago and operates with 100+ staff. The company is involved in many parts of the public safety automation market and has clients all over the United States. A few of their products include police and fire department Record Management Systems (RMS), Mobile Field Based Reporting (MOBILE/FBR), Computer Aided Dispatch (CAD), and real time data Dashboards (DASHBOARD). Even though VisionAIR has many products out they primarily focus on record management systems and computer aided dispatch systems. VisionAIR was the inspiration for this repository. Through working in their integration department I saw many opportunities for improvement which could not be provided by their current tool set. This section describes the company at a glance.

- **Vision**: To be a company known for its integrity and high moral standard. Encouraging creative solutions and a passion for excellence. A company that respects the contributions of every employee, client and partner. That is an aggressive competitor adding value to all aspects of our presence in the Public Safety marketplace.

VisionAIR is involved within many different areas in the automated public safety business. Some products include: Agency Record Management Systems, Computer Aided Dispatch, Fire and Emergency Dispatch and Alerting, Mobile and Field Based Reporting, and many others. The products involved in the demo will be the VisionRMS (Record Management System), VisionCAD (Computer Aided Dispatch).

Within VisionAIR there is the need to interface with 3rd party vendors consistently, this is due to the number of products developed and maintained in house. The following chapter will describe the work flow from the initial sales department contact to the completed development of an interface.

Document Control and Customer Relations

As within many companies the store and handling of documents are critical to business processes. Therefore there needs to be a strict document control process. In order to retain order and control with documentation VisionAIR tries to keep a document control process in motion. This proves to be harder said than done, becausethe documents are created within different departments in the company. VisionAIR has used CRM, Sharepoint and StarTeam as a document storage process but document retrieval and storage is still not efficient. This is another advantage of the repository.

There are multiple documents that exchange hands within the interface creation process. These documents includethe DOW, LOE, FSD, and the SSD. These documents serve important roles within the interface process; therefore all of these documents need to be stored in a central location within the company. There should also be some business logic which relates particular documents to the interfaces they define, allowing for interface and document ownership. This will allow users of the repository to find documents for different interfaces related to the requirements they provide, facilitating document reuse. The document reuse and retrieval will aid in the creation of new documents and in overall interface analysis.

Current Control Process

The current process at VisionAIR has proved to be insufficient due to old practices and the lack of collaborative efforts between departments. The locations of files are mainly unknown, but usually they are located locally on employee computers, or buried within star-team, CRM, orSharePoint.

CRM processing and storage

Within all software development companies there is a customer relations department. This department's responsibility is to keep track of all client or vendor records, including contact information, previous orders, and even potential leads on new clients. In order to aid in this effort the department will often use a client repository, which holds all client related information.

This customer relationship management system contains all customer created data and also aides in the storage of client/contact information, agencies, 3rd parties/vendors, sales data,

etc. VisionAIR keeps all their client information within the CRM system and ultimately stores the information into a common SQL database. The repository will take advantage of this by referencing specific sales documentation within the CRM database and presenting the information to the repository user.

Related Work

The need for integrated systems has been existent since the creation of the database. More specifically focused on database integration, this is because the database serves as the source for information within systems. Due to these separated database systems there is a need for system integration, this is the focus of this paper and other efforts in the IT field. More specifically an open source software solution for supporting data management called Phynx.

The Phynx system was a system developed to provide a flexible and economical solution for user level review of information from databases. This system was developed on the Flex platform and uses many of the components within this repository, all of which are shared within all Adobe Rich Internet Applications. This particular application consists of an Internet client communicating with a Tomcat Java Application Server. Tom cat serves as the data provider and connectivity monitor between the application and the database. The system proved to increase patient data validity though user input and participation. Below is the structure of the application. [3]

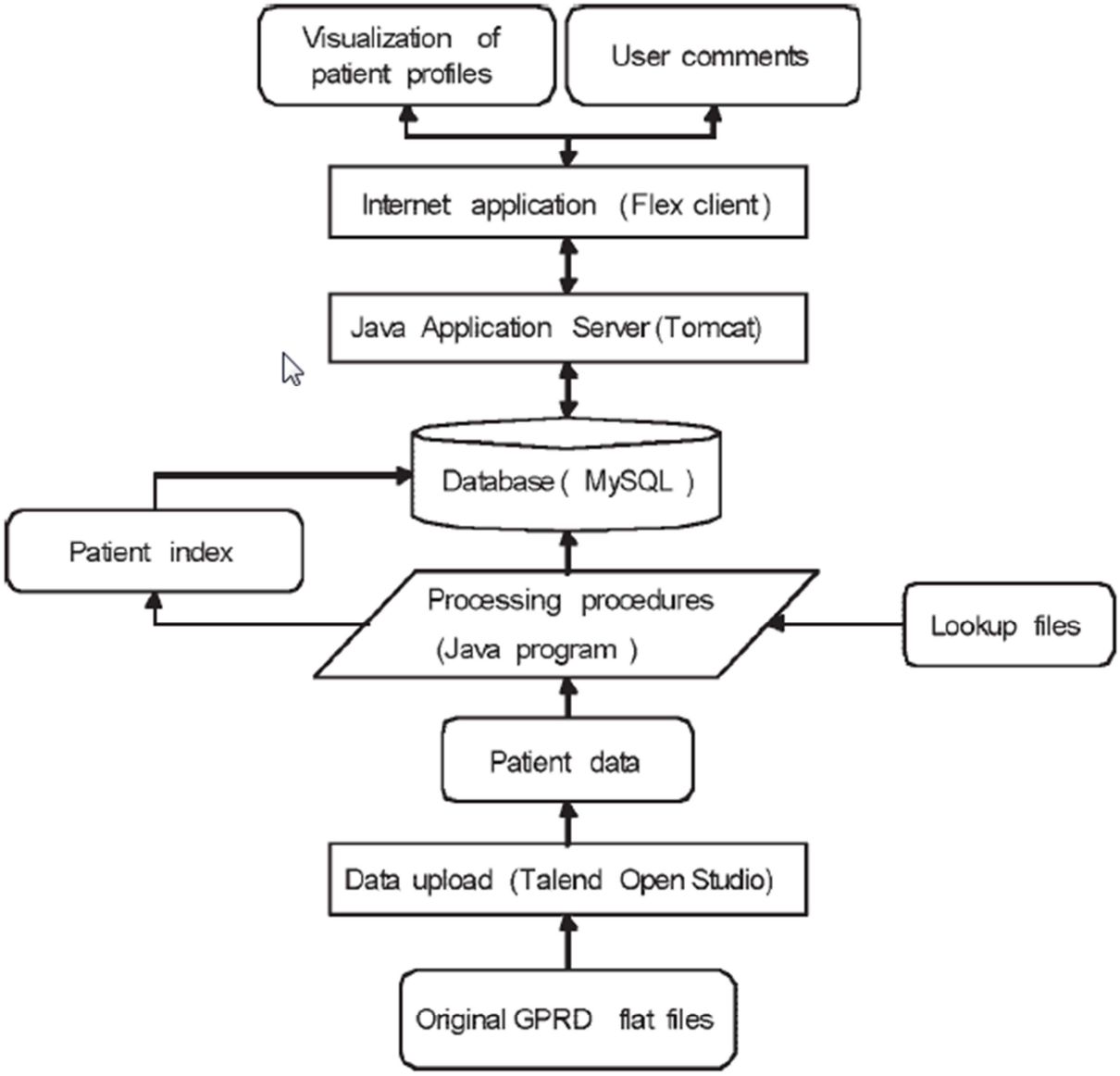


Figure 6: The Phynx system's architecture [3]

The Phynx system created allowed users to efficiently browse through existing data on a database using a rich user interface. Through the use of a similar integration approach; a business, like VisionAIR, could benefit from business data storage and the centralization of the data. Custom Integration and Business

Chapter 3: Research Method

There was a need for a central repository for all developed/conceptual interfaces and their related documentation with an associated storage/access process for all the custom solutions and Interfaces. This project has completed the necessary requirements and delivers a useable application; this application will be the repository. The repository includes all solutions and interfaces; the data contains all interface data and documentation. Through the use of this repository the staff will have automated custom solution and interface data mining.

The Interface repository which will serve as the centralized repository for all interface information, it is essentially the combination of CRM data and local database information. The collaborative use between different departments, such as sales and integration, will ultimately boost the repository's impact within the business. This impact can be measured by different performance metrics within particular sections of the interface creation workflow. An example metric could be viewed as the turn-around time between initial client contact and custom interface quote delivery, time is money so a goal would be to decrease the time and work involved within this step. The following section describes in more detail the goals of the repository.

Short term goals

- ***Organization*** - Each interface is to be stored within the repository with a list of generic attributes for each interface. The repository will take advantage of these attributes by facilitating interface navigation, storage and document retrieval.

- ***Document Reuse*** – With a central location for searchable interface documents, work spent on documentation will decrease. The repository will provide the user with shortcuts to retrieving and reusing documents.

Long term goals

- ***Interface Reuse*** - The repository will give users a different look at their interfaces in the hopes of spotting previously developed interfaces with the same requirements as current interfaces. This is uncommon due to the many different versions of software used by agencies.
- ***Interface Standardization*** - This is considered the ultimate goal of all integration practices, it eliminates any new work on interfaces most importantly development. A standard interface will be less expensive as previous interfaces allowing the company to be agile in the competitive automated public safety market.
- ***Automated Documentation*** - The automation of interface quote documentation may be possible with more time and use of the repository. This is only possible with the quotation process because of the high level nature of the DOW. The mitigation of this document creation process can allow the staff to focus on other tasks.

The repository will achieve these goals through the use of interface specific data, local document locations and database integration. In order to organize the interfaces by specific criteria there will need to be a standard template for interface details. Each interface will have a list of standard attributes; these will be needed in order to sort and navigate through

interfaces. The interface attributes are for repository indexing; every interface must have the following required attributes in order to be included into the interface repository database.

Interface Attributes

- Solution Name (text)- name of the solution
- Description (text) – brief description of the interface in question
- Developed (Boolean) – is the interface developed or conceptual
- DOW (Boolean/link) – DOW? Link to its location within the company file system
- VisionAIR product (text) – what product is being integrated
- 3rd Party required (Boolean) – involved 3rd party?
- 3rd Party details (text) – details about the 3rd party, name
- CS Unit Price (money) – custom solution unit price as quoted
- Total cost (money) – cost of development
- FSD (Boolean/link) – similar to DOW
- Interface type (lookup) – is the interface an export, import, etc.

These attributes will be utilized by the repository through the use of a graphical user interface. The repository will be a client side application ran within the internet explorer. The following section describes this in further detail.

The Application

This application consists of a front end interface to allow the user to browse and search though the repository for specific interfaces. The interfaces will be stored with certain

attributes within the database, these attributes will be leveraged when indexing and searching through the repository. Below is a table from the database for interfaces:

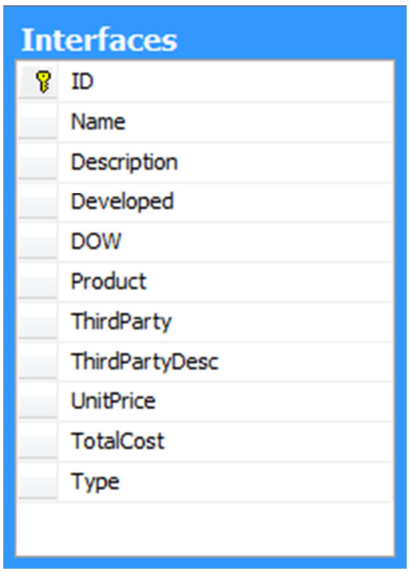


Figure 8 Interface Database Table

From looking at the diagram you will see that all interfaces will have a unique identifier (ID), this will individualize every interface and allow for retrieval from the repository. With future integration and development of the database, it will include other tables, such as: Agency, Client, 3rd Parties, etc. This will increase the interface repository's use and role within the organization. The following diagram shows the use cases related to primary application functions between the user and the databases.

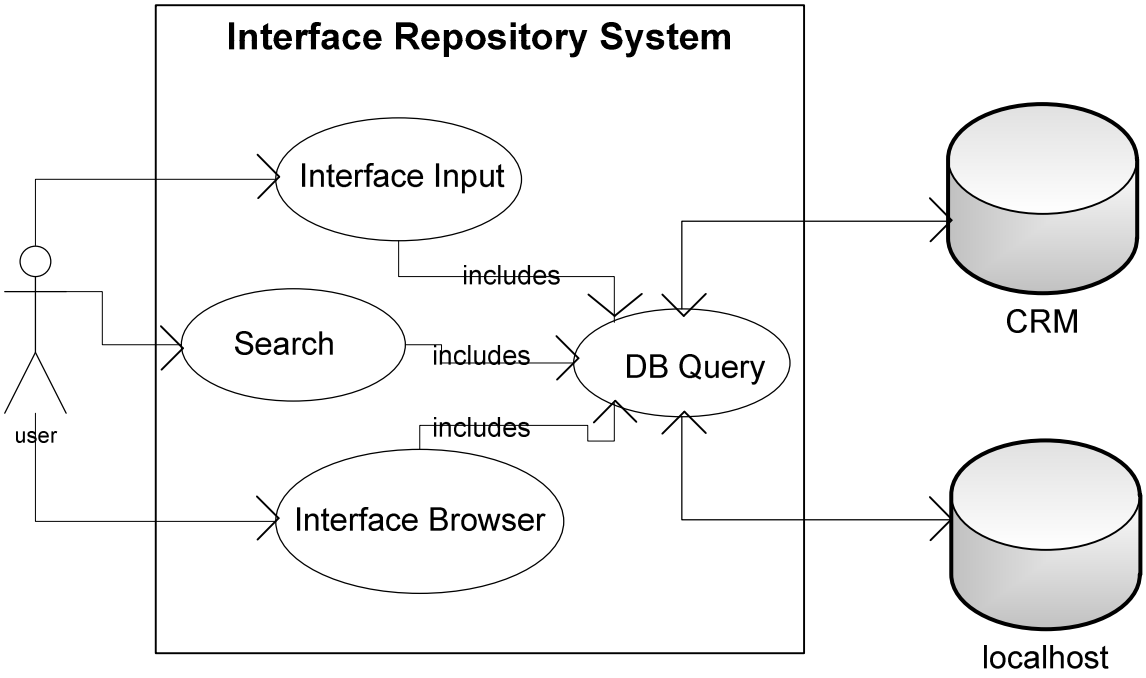


Figure 9 Use Case Diagram

The user will use the interface to input data in order to run queries on the database, the user will also have access to all of the interfaces via Interface Browser which is essentially a 'select all' query on the database.

Database Details

CRM – this systems database is located on the CRM server and any interactions with it will be strictly read only, ensuring consistency.

Local database – this is located locally on the server hosting the interface repository, for keeping track of abstract attributes not previously contained within the CRM database.

There are several integration points to be considered, such as database connectivity. At first the only remote database connection made will be with the CRM database. Possible future development will provide integration with other systems such as StarTeam and Sharepoint.

Local database connectivity will be seamless due to a middle web service and the local SQL instance running on the development machine. This is only for demonstration purposes, the final version will have a dedicated server for all database related queries, transactions, storage etc. The CRM database will be setup with a generic dummy database, this test environment will reduce risk in the developmental stages. The final application will be plugged into the company CRM database to allow for proper repository functionality.

Graphic User Interface

The layout of the interface is very important when developing an application. The easier your application is to use the more productive the user will be, as well as minimizing any user frustration. I decided to have tabbed layout for navigation resulting in a stacked application, meaning the application has several different parts which function differently. The first is the data mining tab, second the interface browser, and lastly the future questionnaire. Currently the questionnaire part of the application is still in its conceptual stage;though user interaction this will evolve and be part of a future effort.The following are graphic representations of what the applicationshould look like after development, they are not the actual system.

Data Mining Interface

This part of the application is for searching through the repository. It includes a panel with several fields to help the user reduce possible interfaces and find exactly what they are

looking for. The result of the users input will be what creates the SQL query for the database. There is also a panel for "Interface Details"; this lets the user see inside of the Interface record within the database. There is also a mini browser underneath the previously explained panels. This holdsthe interfaces currently being searched on by the user, the interfaces included here are determined by the user's input in the search panel. Below is an image representing the first mockup of the Internet Search part of the application.

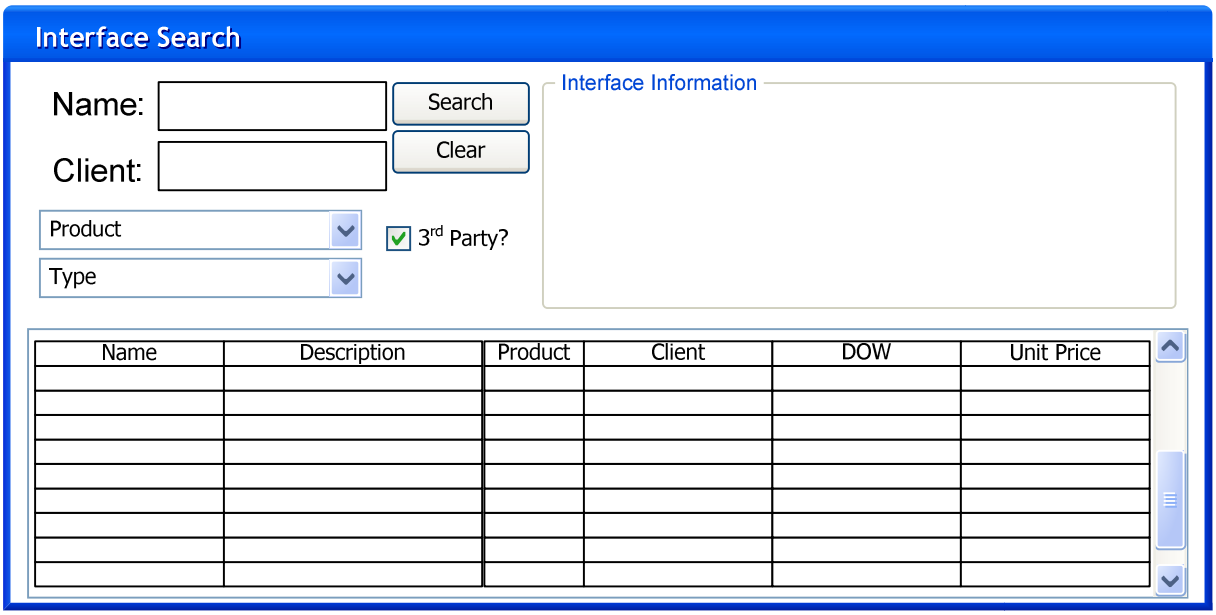


Figure 10 Search Tab of Application

Browser

This part of the application has all the interfaces that are being stored within the database. This will allow the user to browse through the repository, displaying certain attributes to help sort, compare, and gauge the interfaces as a whole. Below is an image representing the first mockup of the browser part of the application.

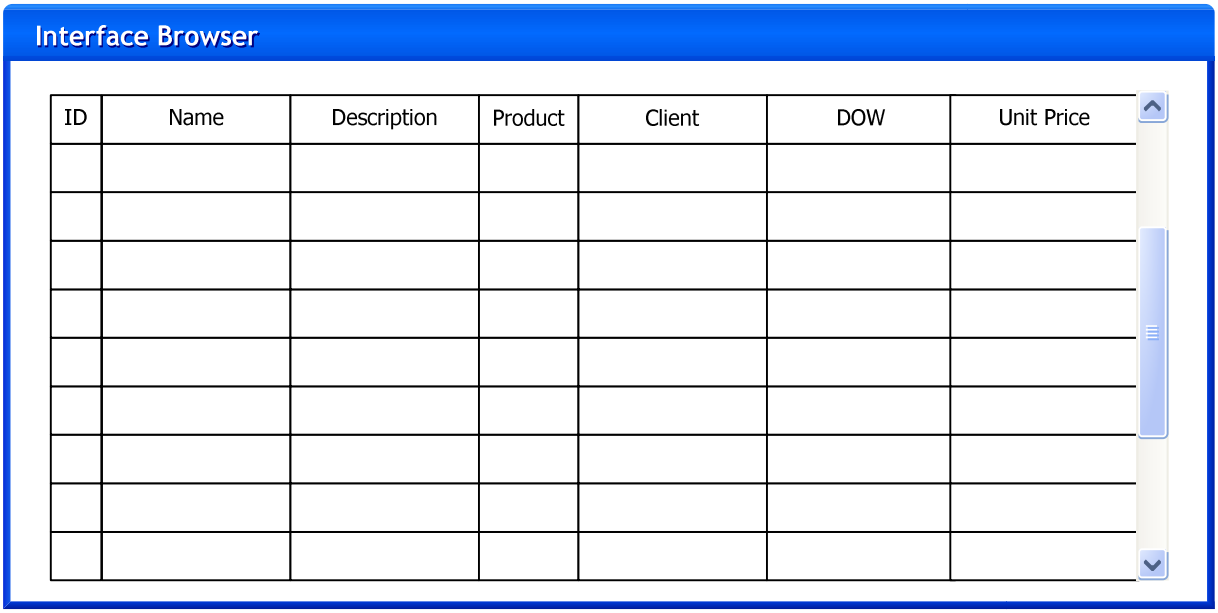


Figure 11 Browse Tab of the Application

Questionnaire

Considered future work... This will be a part of the application which the sales staff will use. It will take several questions from the user, such as: Product used, client, state, etc. After the user is finished answering all the questions; their answers will be compiled and a search will be done on the repository resulting in relevant interfaces or finding a previously developed interface. In a perfect world the user will receive information about a developed interface, this will allow for reuse of the documentation and or the actual/partial implementation of the interface. This tool will become stronger with the increase in the number of total interfaces, as will the repository as a whole.

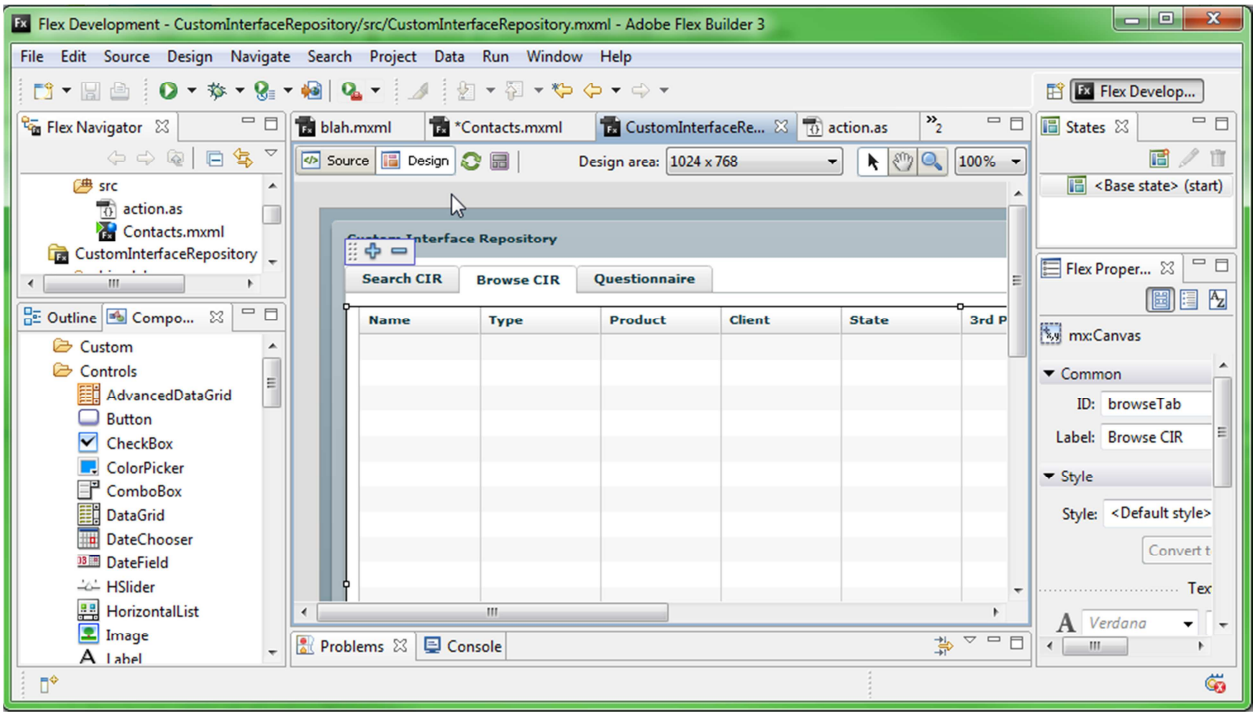
Chapter 4: Research Results

The following is a list of items completed of the project.

1. Setup development environment. Complete
    - Setup SDK.Complete
    - Setup IIS. Complete
    - Setup Databasemanagement software. Complete
2. Setup and configure the databases. Complete
    - Local database for application specific information. Complete
    - Simulated CRM database for business related information. Complete
3. Create dummy data for databases and import. Complete
4. Develop and create database connectivity. Complete
    - Create web service for database/application communication. Complete
    - Connect web service to application. Complete
5. Create user interfaces. Complete
    - Interface browser. Complete
    - Interface search. Complete
6. Integrate data grids to display database information. Complete
7. Create database queries (SQL queries/procedures).Complete
8. Create use cases for application use.Complete
9. Create simulated 'real' data for use case testing.Complete
10. Gather results..Complete

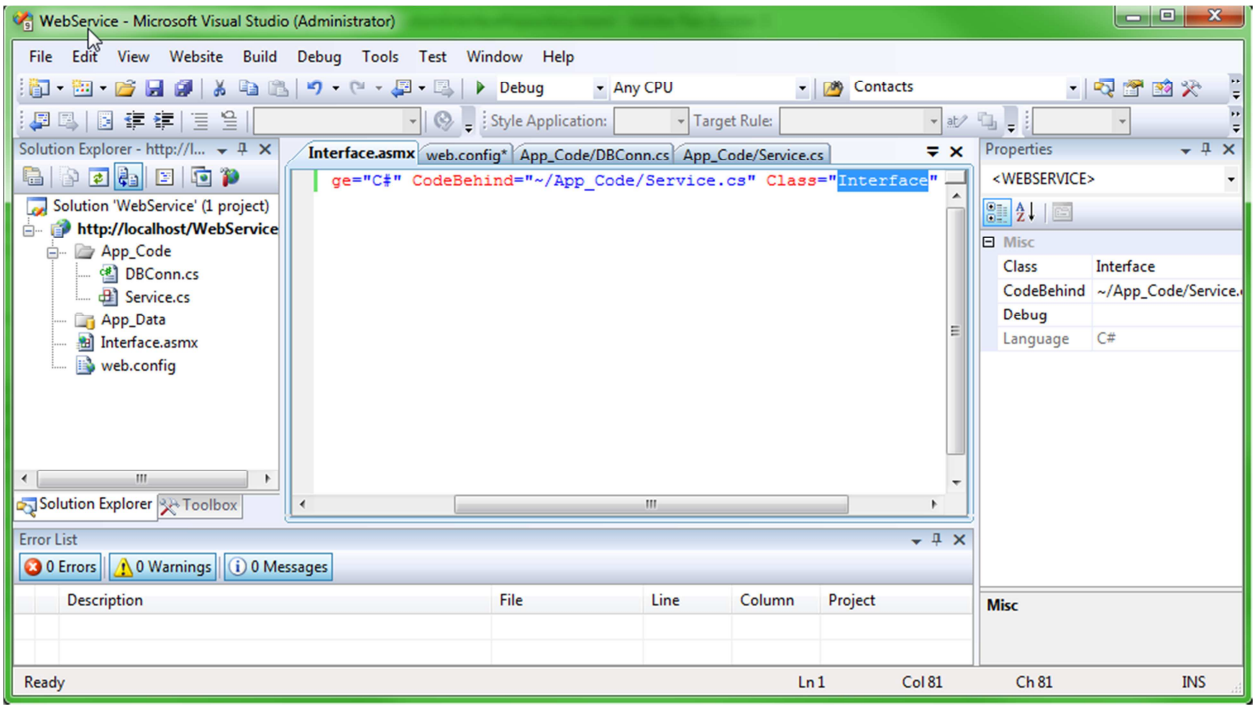
4.1. Setup development environment.

- Adobe Flex Builder - The following picture shows the developing environment in which the frontend application was developed. The Navigator lets you navigate through different projects, so if you are using one as a reference it is easy to compare and edit as needed between the two projects. It also has a feature to view the source and design on the fly so you can visually see the code in design mode and add functionality to the visual objects via the source mode. It also recognizes all objects created within a project meaning, when you click an object (text field, button, pane) it will automatically show you the properties and relationships in the properties panel.



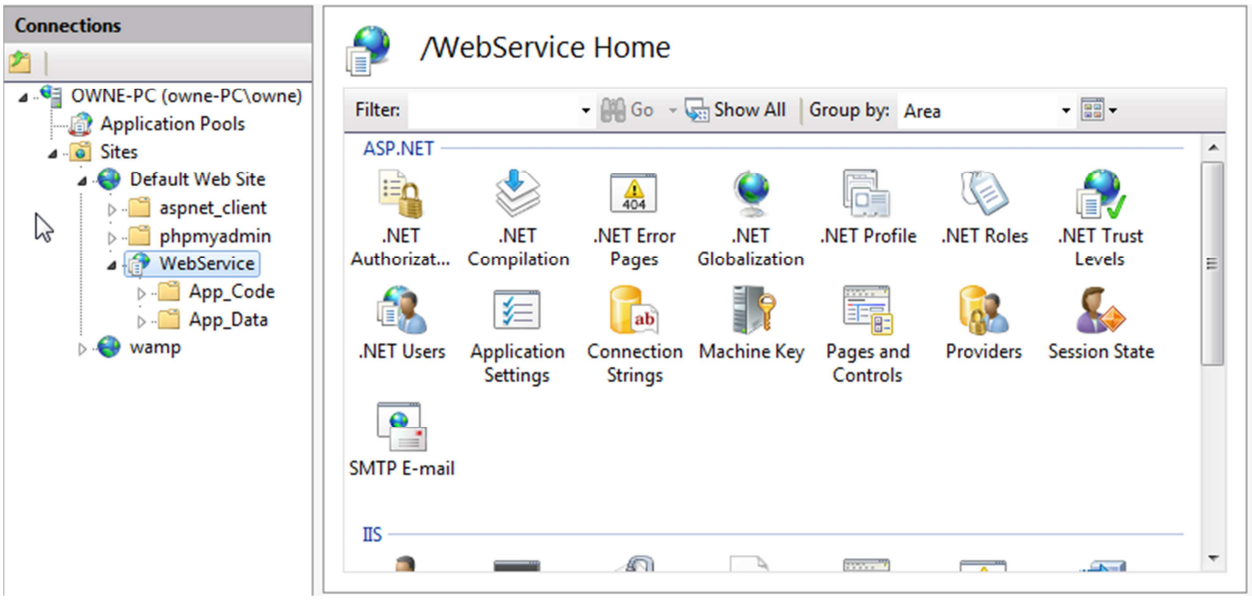
Adobe Flex Builder

- Microsoft Visual Studio – This software operates much like the Adobe Flex Builder. But has one advantage in that it was created in order to functionally operate with other Microsoft components (i.e. SQL Server, IIS, etc.); this is critical in the effort to integrate different services offered by Microsoft. In this case I integrated an IIS web service with a database, and created functions or methods in order to get data from one to the other.



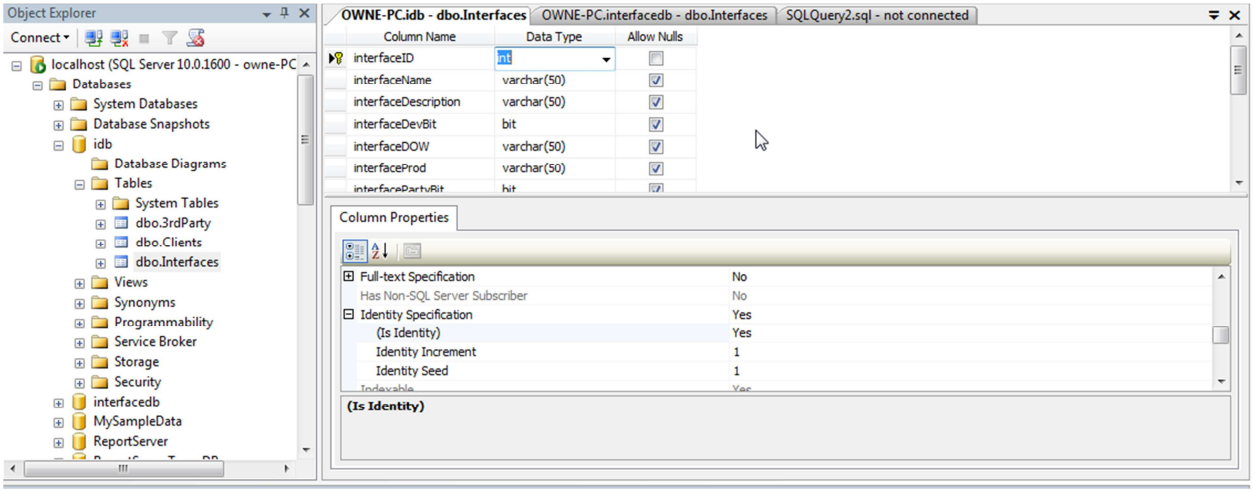
Microsoft Visual Studio

- IIS – This is a feature included with most Professional versions of the Microsoft operating system. It allows a user to locally host a web site or service in order to connect locally (LAN or VLAN) to the hosted websites using various protocols and ports to control access to different services.



Internet Information Services

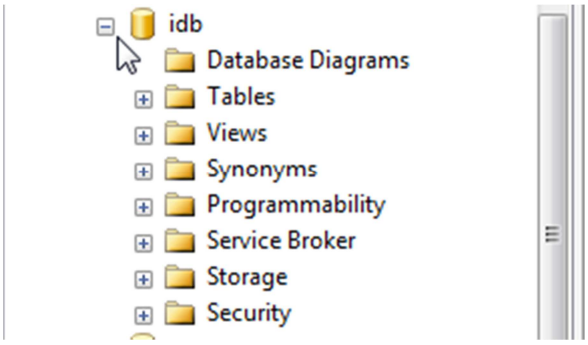
- Database Management Studio – This software allows developers to have a user interface to their database. Meaning they get access to all the functionality their database has to offer through an intuitive interface. It allows you to manipulate and access data via clicking on representational objects instead of using the command line (SQLPlus) to execute commands on the database.



Microsoft SQL Server Management Studio

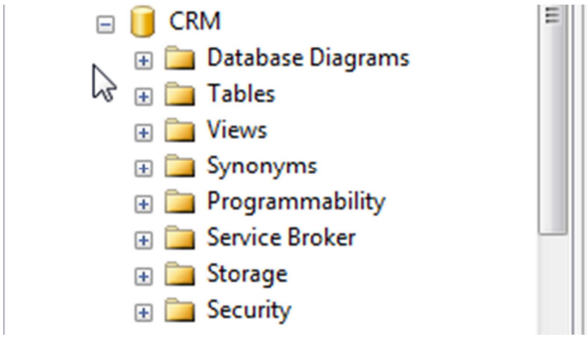
4.2. Setup and configure the databases.

- Local database for application specific information. This database is used only to store data specific to the Repository system, it will not have any value outside of the system. It will contain certain aspects of business logic, such as a count on particular interfaces developments. Through the use of reporting, this count can be leveraged along with other metrics in the effort to standardize certain interfaces.



Interface database

- Simulated CRM database for business related information. This database will only be used to gather business side information (sales information, document locations, 3rd Party and client information).

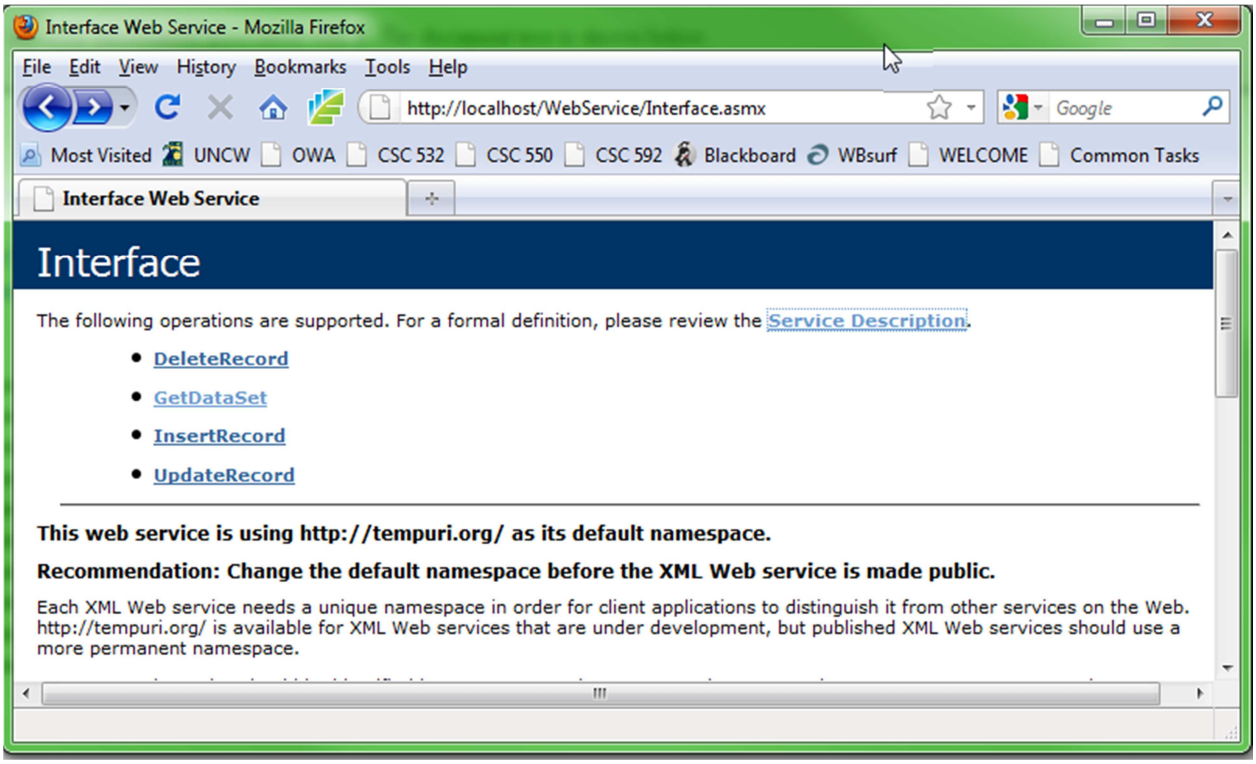


CRM sample database

4.3. Create dummy data for databases and import.

4.4. Develop and create database connectivity.

- Create web service for database/application communication. This web service will be utilized in the effort to extract data from the databases. This particular web service has 4 “WebMethods” they can delete; get all data, insert, and update records within a connected database. It uses the NT web service user when connecting therefore the user will need to be given privileges within the database being queried.

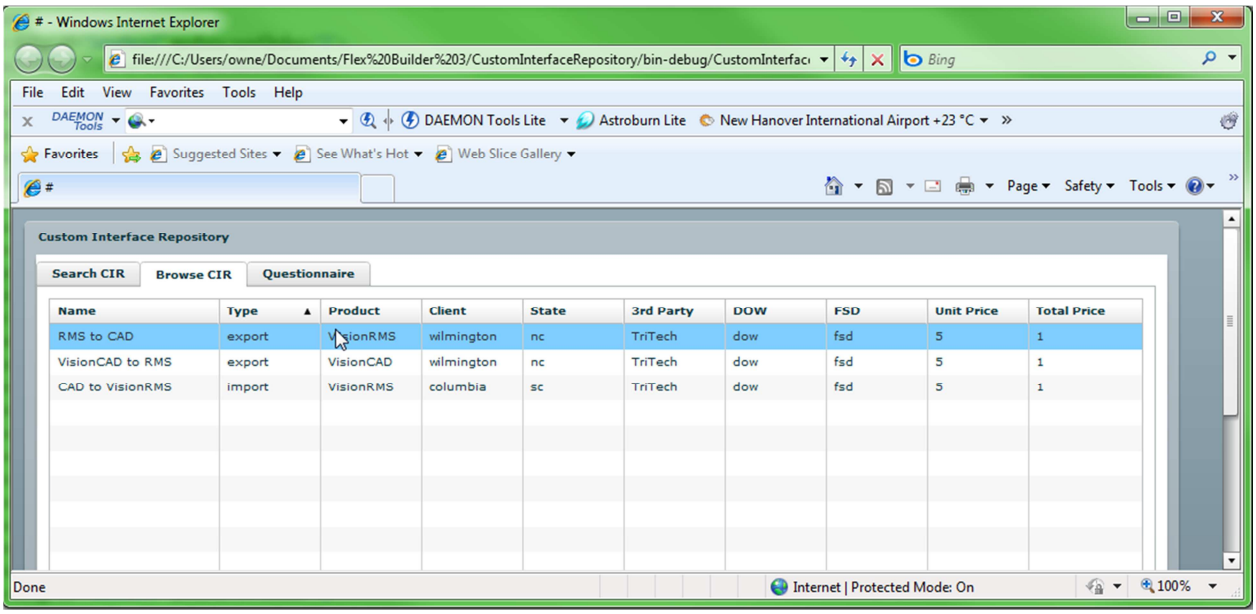


Web Service root

```
- <NewDataSet>
  - <Interface diffgr:id="Interface1" msdata:rowOrder="0">
      <interfaceName>RMS to CAD</interfaceName>
      <interfaceType>export</interfaceType>
      <interfaceProd>VIsionRMS</interfaceProd>
      <interfaceClient>wilmington</interfaceClient>
      <interfaceState>nc</interfaceState>
      <interfaceParty>TriTech</interfaceParty>
      <interfaceDOW>dow</interfaceDOW>
      <interfaceFSD>fsd </interfaceFSD>
      <interfaceUP>5.0000</interfaceUP>
      <interfaceTP>1.0000</interfaceTP>
    </Interface>
  - <Interface diffgr:id="Interface2" msdata:rowOrder="1">
      <interfaceName>VisionCAD to RMS</interfaceName>
      <interfaceType>export</interfaceType>
      <interfaceProd>VisionCAD</interfaceProd>
      <interfaceClient>wilmington</interfaceClient>
      <interfaceState>nc</interfaceState>
      <interfaceParty>TriTech</interfaceParty>
      <interfaceDOW>dow</interfaceDOW>
      <interfaceFSD>fsd </interfaceFSD>
      <interfaceUP>5.0000</interfaceUP>
      <interfaceTP>1.0000</interfaceTP>
    </Interface>
  - <Interface diffgr:id="Interface3" msdata:rowOrder="2">
      <interfaceName>CAD to VisionRMS</interfaceName>
```


Sample Data Retrieve

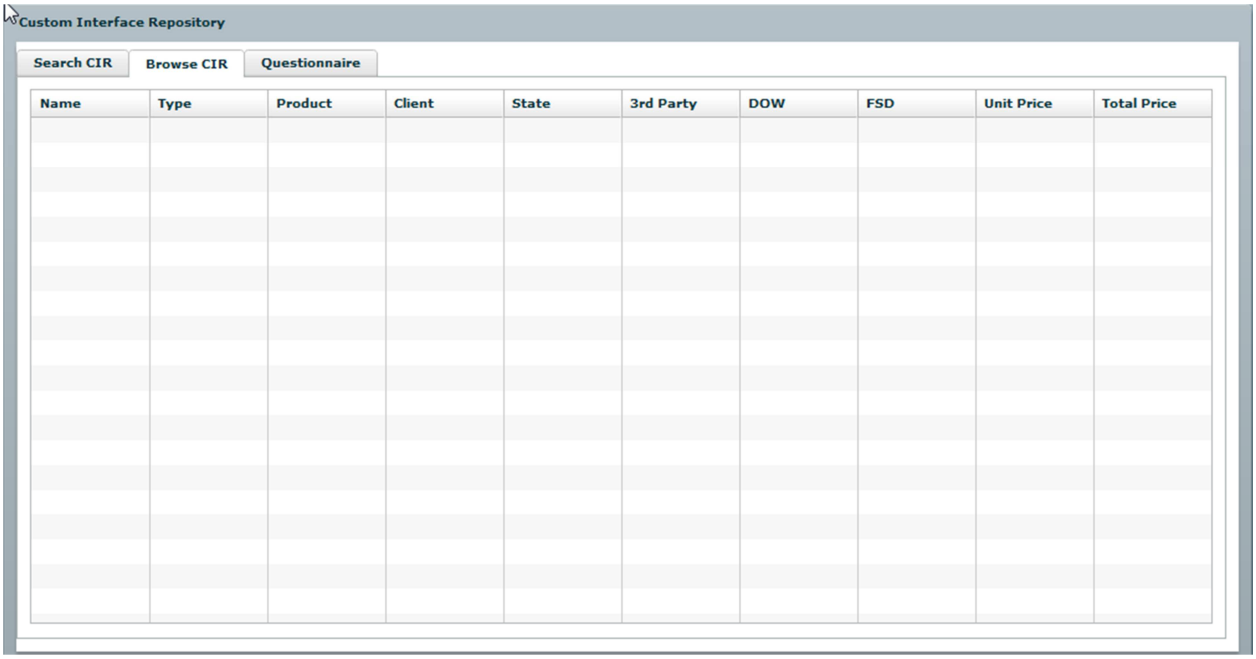
- Connect web service to application. Once integrated with the application, the web services XML output will be transformed and loaded into the application for user viewing.



Browser with database data

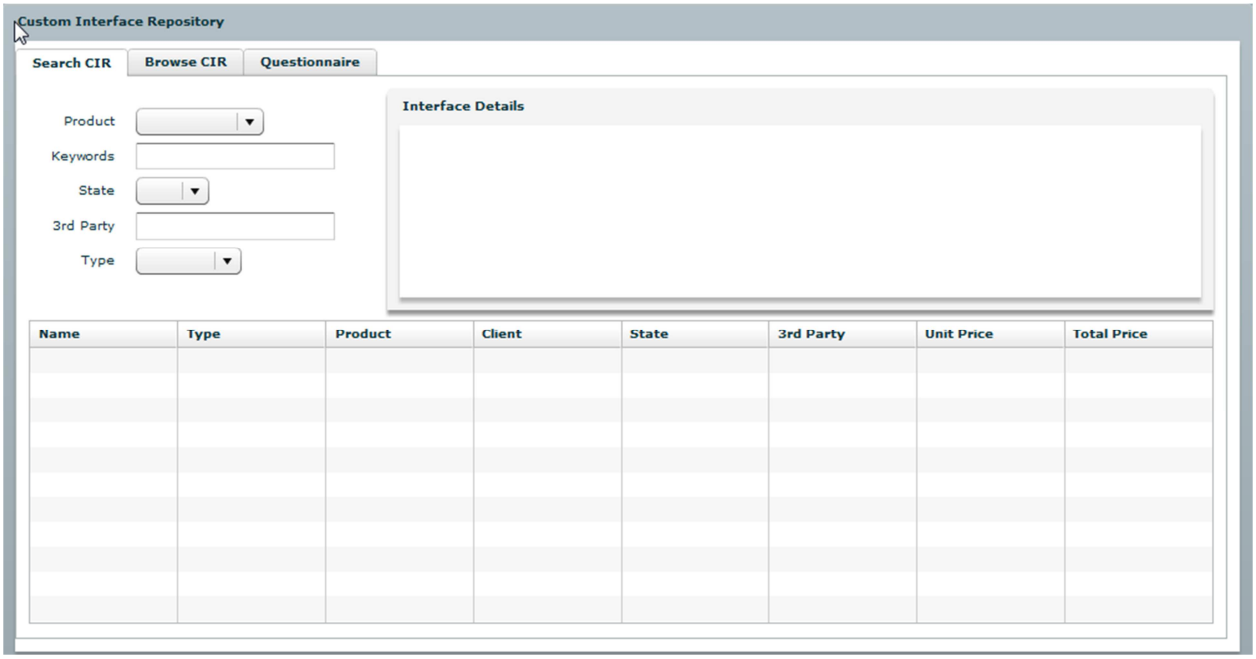
4.5. Create user interfaces.

- Interface browser. This is the browser in which the user can click and sort through different interfaces contained within the local Interface database.



Interface Browser UI

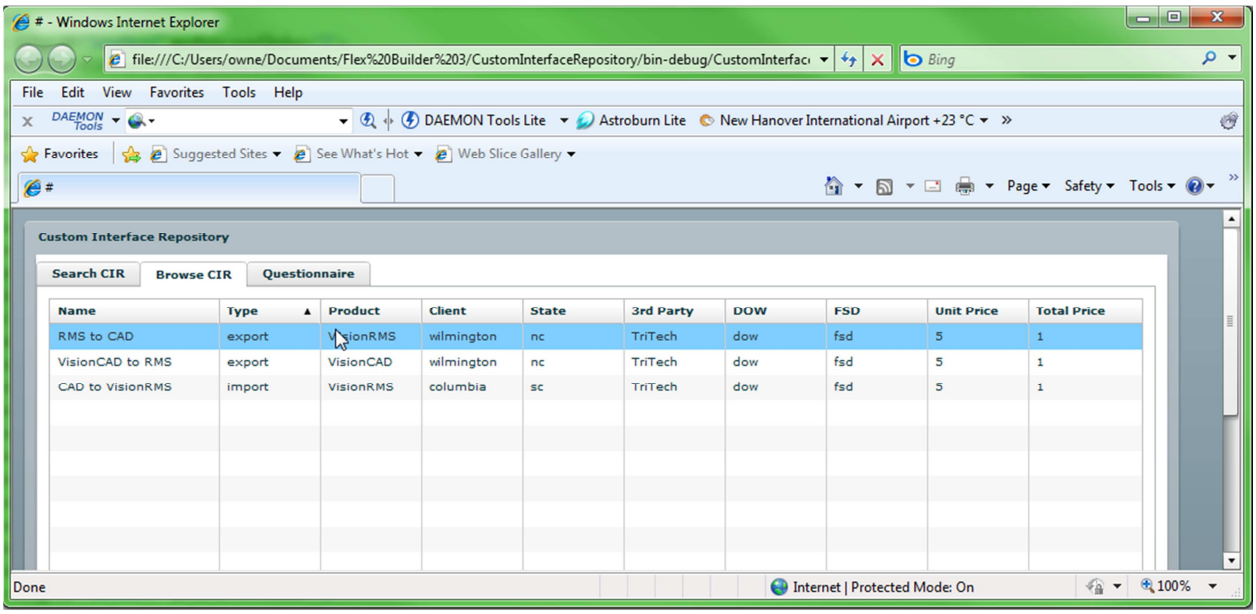
- Interface search. This will allow the user to search through the interfaces contained in the database and display the resultant interfaces' information.



Interface Search UI

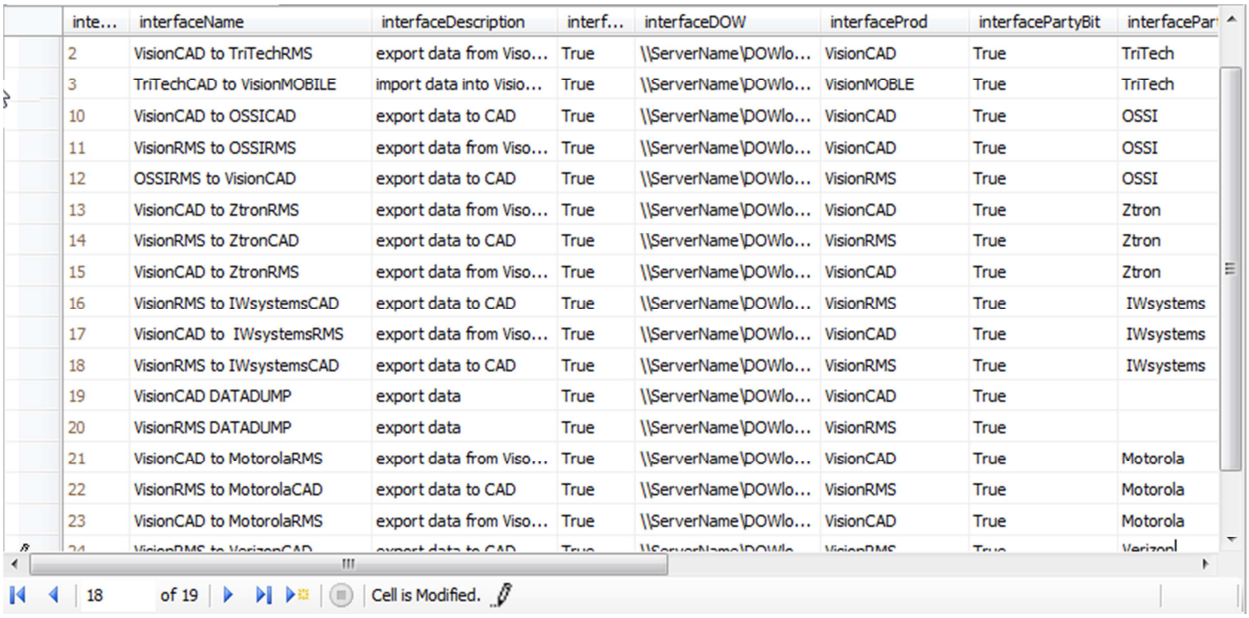
4.6 Integrate data grids to display database information.

- Browser with data - This shows the integration of the output XML from the web service into the Browser.



Browser with database data

- Testing Database: This database contains 20 'dummy' records to simulate records within a production system. This set of data is for demonstrative purposes only.

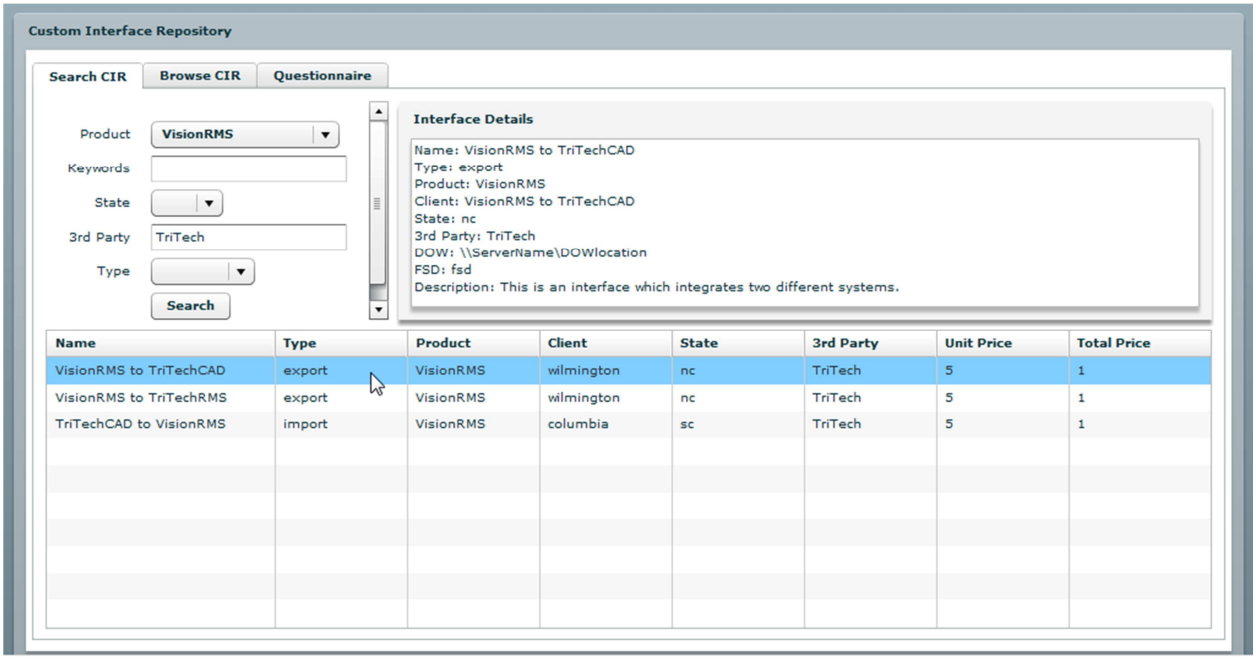


inte...	interfaceName	interfaceDescription	interf...	interfaceDOW	interfaceProd	interfacePartyBit	interfacePar
2	VisionCAD to TriTechRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	TriTech
3	TriTechCAD to VisionMOBILE	import data into Visio...	True	\\ServerName\DOWlo...	VisionMOBLE	True	TriTech
10	VisionCAD to OSSICAD	export data to CAD	True	\\ServerName\DOWlo...	VisionCAD	True	OSSI
11	VisionRMS to OSSIRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	OSSI
12	OSSIRMS to VisionCAD	export data to CAD	True	\\ServerName\DOWlo...	VisionRMS	True	OSSI
13	VisionCAD to ZtronRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	Ztron
14	VisionRMS to ZtronCAD	export data to CAD	True	\\ServerName\DOWlo...	VisionRMS	True	Ztron
15	VisionCAD to ZtronRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	Ztron
16	VisionRMS to IWsystemsCAD	export data to CAD	True	\\ServerName\DOWlo...	VisionRMS	True	IWsystems
17	VisionCAD to IWsystemsRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	IWsystems
18	VisionRMS to IWsystemsCAD	export data to CAD	True	\\ServerName\DOWlo...	VisionRMS	True	IWsystems
19	VisionCAD DATADUMP	export data	True	\\ServerName\DOWlo...	VisionCAD	True	
20	VisionRMS DATADUMP	export data	True	\\ServerName\DOWlo...	VisionRMS	True	
21	VisionCAD to MotorolaRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	Motorola
22	VisionRMS to MotorolaCAD	export data to CAD	True	\\ServerName\DOWlo...	VisionRMS	True	Motorola
23	VisionCAD to MotorolaRMS	export data from Viso...	True	\\ServerName\DOWlo...	VisionCAD	True	Motorola

18 of 19 Cell is Modified.

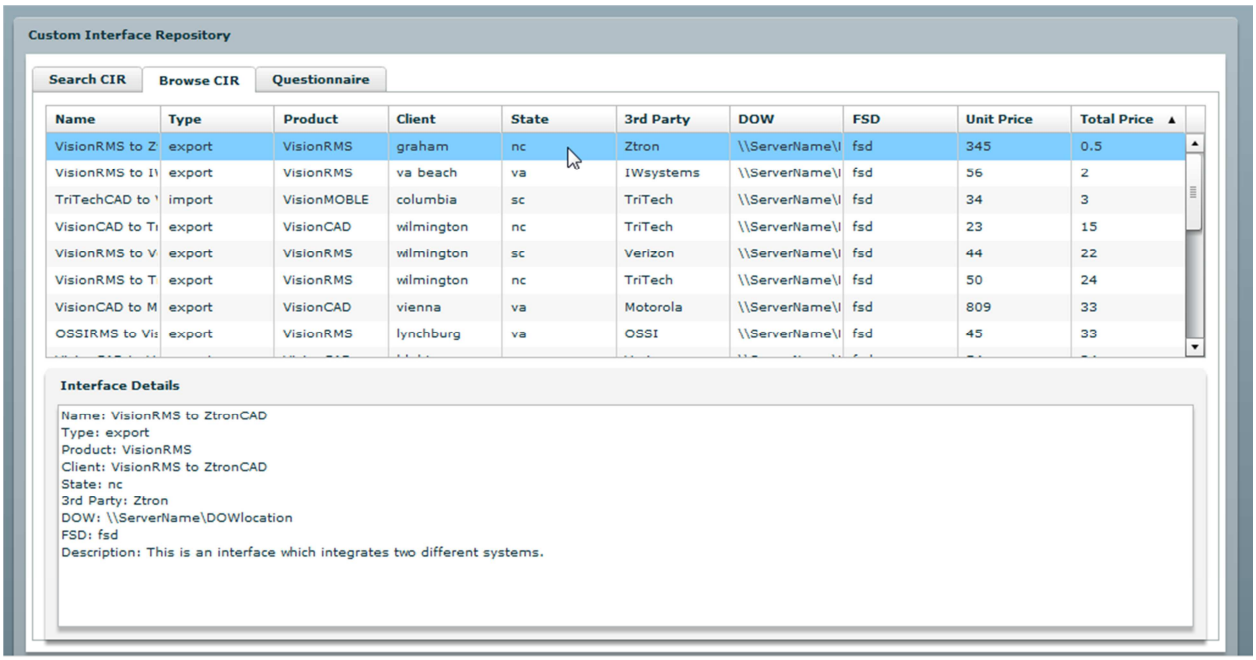
- Interface Search Query– this shows the use of a single search on the database and the data returned as well as displayed in the details box.

Sample Test Case: Find all interfaces developed using the VisionRMS product and in collaboration with the 3rd Party vendor TriTech.



- Interface Browser – this shows the use of sorting through the interface Browser and how it can be helpful.

Test case: Find the cheapest interface developed for NC. Result = Graham.



Developing environment

- Microsoft Windows 7
- Adobe Flex SDK
- Adobe Developer
- Microsoft Visual Studio 2008
- Microsoft SQL Server 2008
- SQL Management Studio
- Microsoft Dynamics CRM

User environment

- Any OS
- Microsoft Internet Explorer (7+)

Chapter 5: Discussion and Implications

Change Control - An issue brought up during the initial proposal was a change control process for modifying the database. This includes adding attributes for interfaces; these attributes would be requested by the actual users. I propose that the interface repository's database be edited only by the database administrator. This results in a change control process in which the following would happen:

1. User declares the need for a new attribute
2. User submits a change request to the development team
3. The team reviews the request and declares whether the attribute is realistic and useful
4. If it passes, it gets added by the database admin and tests are initiated in a test environment before rolling out the new database version.

File Paths - Something else to consider: the DOW's should all be in an UNC path format. This means the path should include the ServerName and location within the server. These links will be 'clickable' in order to open the DOW for viewing, saving, editing, etc. This is very important because this system's efficiency and effectiveness hinges on the ability to freely view these files.

Security – The application's access security will not be a problem with the as-is system, as it is totally internal to the business. In later releases it may become available for remote access. In this case you will need to implement several different security precautions. Such as the following:

- ***SSL*** – a Secure Socket Layer would be essential for any publically accessible web application.
- ***Firewall*** – a firewall put between the outside world and the application server would help cut down inadvertent or malicious access.

- ***SQL injection*** – there will also be a need to sterilize the SQL inputs. This can be done though adding a black list of words to the input (ex. Delete, create, etc.) or a stored procedure approach could be implemented.

Standardization workflow – Standardization is one of the long term goals to be achieved through the use of this repository. In order to achieve an "Interface Standard" or "Universal Interface" there must be a workflow in which one could follow; this workflow would be as follows:

- ***Realization*** - There will need to be a metric in determining an interface candidate for standardization. Some examples would include: the amount of inquiries on a specific interface (ex. Number of clients wanting a specific interface: CAD to RMS) or the required inputs and outs for a specific state (maybe all of NC police departments can be standardized). These metrics will need to be analyzed and determined so that the repository can use the logic and eventually produce reports.
- ***Business Case*** – This phase will require a business case in order to start in house effort and development. It would be presented to executives in the effort to get research and development funding. If accepted the workflow will proceed.
- ***Analysis*** – This stage would be where system engineers determine what is *really* needed in order for clients to functionally use the interface. This stage would involve researching all of the previous interfaces and their data in between. If there is a need for a custom field outside of the "standard" additional fees can be added, but the functionality of the interface (source code) will essentially be the same.
- ***Development*** – Using the previous stage's results, the actual interface development will begin.

- ***Testing*** – the interface will be put through a thorough testing phase. In which, the interface will encounter many different systems including other 3rd Party vendor's products (simulated virtually).
- ***Marketing*** – This phase is beyond the software development's scope but is very important. This is because the main selling point of this standard interface is its price. Inexpensive functionality that meets unlimited client's requirements would be the selling point.

Automated Documentation – The topic of automated documentation and how it can be accomplished is an important consideration for the future evolution of this project. This could be done though using an advanced search on the interfaces according to the client's interface requirements. This search would take in the requirements and search the database for 'like' interfaces, of which there exists documentation. If there is a one-to-one correlation between the previously created interface and the user's requirements then the system could use the same document with a different reference for the client. Or if the interface being inquired is implicit enough then the system could gather the attributes stored in the database already and create a document itself on the fly. This would mitigate the step usually taken by integration managers in which they would look through previously created interfaces and manually open and edit the documents, saving time and money.

Chapter 6: Conclusion and Future Work

In conclusion, the system functions as required by the business processes. I believe the system will build its true value after the integration of the two databases within the business and the collection of all historical data. The amount of data is very important in that it makes the use of the system more powerful. The use of all this data will help in the effort to standardize interfaces; the system will more than cover its worth with the standardization of even one interface.

As with every project there are goals that the end product is meant to help achieve. The short term goals for this project are organization of internal documentation and interface storage and document reuse. These goals should be realized in the first 6 month of use. Organization of internal documentation and interface storage will be the first to be achieved. This is because with a centralized system all data will reside in one area, limiting the possibility of lost interface data (XML examples, email conversations, Process documents, etc.). Due to this new found organization the documents will be centrally located and readily accessible, therefore boosting the practice of document reuse and reference.

The long term goals of the developed system are: interface reuse, interface standardization, and automated documentation. These goals will not be realized before at least one year plus of consistent use of the developed system. The first two goals go hand in hand. This is because through the use and knowledge of interface reuse, product managers can analyze previous interface reuses in order to determine standardized interface candidates. After which these candidates will enter the standardization work flow as described in the previous chapter. These goals would be the most important when determining value of the overall system, this is because the standardization of an interface can generate actual revenue, revenue of which would not be realized if the system was not in place. The last goal and most intriguing would be automated document creation, the creation of

interface specific documents. This can be achieved only when the system has enough data, enough data to effectively create documentation is unknown at the moment but will be met eventually in the years to come. The following is a list of future work to consider in the lifetime of the system.

1. 'Dress' up the interface search and browser to make it easier to read and view data.
2. Develop the questionnaire portion.
3. Automate the DOW suggestion component based off the answers to the questionnaire.
4. Integrate with other systems within the business, slowly trickle into different departments.
5. Create similar systems for different departments...

References

1. Kevin Hoffman, Patrick Eugster. Towards reusable components with aspects: an empirical study on modularity and obliviousness, Proceedings of the 30th international conference on Software engineering (ICSE), Leipzig, Germany, May 10 - 18, 2008, pp. 91-100.
2. R. Senthil, D. S. Kushwaha, A. K. Misra. An improved component model for component based software engineering, SIGSOFT Software Engineering Notes, 32(4), July 2007, ACM Press.
3. Egbring, M,Kullak-Ublick, GA,Russmann, S..Phynx: an open source software solution supporting data management and web-based patient-level data review for drug safety studies in the general practice research database and other health care databases, PHARMACOEPIDEMIOLOGY AND DRUG SAFETY, 19 (1): 38-44 JAN 2010.
4. J. Sun, J.S. Dong, S. Jarzabek and H. Wang. Computer-aided dispatch system family architectureand verification: an integrated formal approach, IEE Proc.-Softw., Vol. 153, No. 3, June 2006.
5. Tina Rauch. Automating the Field with a System Solution: The Next Step in Mobile Dispatch. Electric Light & Power, May-Jun 2009, p. 58.
6. Spotlight: Computer-Aided Dispatch Systems, Urgent Communications,October 09, p. 42.
7. Mathieu Fourment, Mark J Gibbs. The VirusBanker database uses a Java program to allow flexible searchingthroughBunyaviridaesequences, BMC Bioinformatics,2008, 9:83.

8. Steven B. Cannon, John A. Crow, Michael L. Heuer, Xiaohong Wang, Ethalinda K. S.Cannon, Christopher Dwan, Anne-Francoise Lamblin, JayprakashVasdewani, Joann Mudge,Andrew Cook, John Gish, Foo Cheung, Steve Kenton, Timothy M. Kunau, Douglas Brown,Gregory D. May, Dongjin Kim, Douglas R. Cook, Bruce A. Roe, Chris D. Town, Nevin D. Young,Ernest F. Retzel. Databases and Information Integration for the Medicagotruncatula Genome and Transcriptom Source: Plant Physiology, Vol. 138, No. 1 (May, 2005), pp. 38-46.
9. Xuan F. Zha. A web-enabled open database system for designand manufacturing of micro-electro-mechanical systems (MEMS), International JournalAdvanced Manufacturing Technology,(2007) 32, p. 378–392.
10. Jim Rapoza. Flex finds its footing.Review: Adobe's Rich Internet Platform Matures,eWEEK,August 28, 2006, p. 34.
11. MichaelVizard. Rich Internet Apps: Best of the thick and thin, a new breed of tools is making it easier to combine collaboration and legacy apps,BaseLine Magazine, Jan 2008, p. 26.
12. Paul Krill. Adobe Pivots Flex to Take on AJAX Rich Internet application technology upgrade lowers cost for deployments, InfoWorldMagazine,July, p. 8.
13. James L. Parrish, Jr., James F. Courtney.ElectronicRecordsManagementin Local Government Agencies:The Case of the Clerk of CourtsOffice in Lake County Florida,Information Systems Management, 24:2007, p. 223–229.
14. Robert A. Shiff, Arthur Barca. The New Science of Records Management,Division of Research, Harvard Business School, 1947, pp. 36-44.

15. Equipment News,Fire Chief Magazine, November 2010, p. 164.

16. Max Goldman, Robert C. Miller. Codetrail: Connecting source code and web resource, Journal of Visual Languages and Computing 20, 2009 223–235.