Bachelor of Technology

Information Technology Project

Data Integration

Shawn D'souza
sdso010@auckland.ac.nz
University of Auckland

June 2012

Abstract

Data integration is a pervasive challenge faced in applications that need to query across multiple autonomous and heterogeneous data sources. My project for BTech this year is to design an interface that will allow users of the application to query and import data from various data sources. In this report I will describe my project, I will explore possible solutions to solve data integration issue and explain why my proposed solution is apt for this project.

Acknowledgements

I would like to thank my Industrial mentor Frederik Dinkelaker for guidance & support in developing the solution. I would also like to thank my academic supervisor for supporting with the research of my project.

Contents

List of Figures

Introduction

Integration of application and business processes is a top priority for many enterprises today. Very few business applications are being developed or deployed without a major focus on integration, essentially making integratability a defining quality of enterprise applications [1]. For my project I will a add feature to the existing application to integrate data from other source into the application. In this report I will talk about what technique companies can use to integrate data from multiple sources. How information from multiple data sources can be queried. And what is the most efficient way of doing it.

Project Description

Company Information

The company I have chosen to work with for my BTech project is Torque IT Solutions. It is a start-up company that began in June 2011. Torque I.T.S is a finance company that primarily focuses on enabling car dealerships make the most profit from their business transactions, by providing them with the technical tools required to monitor and audit sales transactions in the company. The company has many software applications under development, and will be working on one of the applications, namely DPMS (Dealer performance management system). The application allows the Car dealer managers to manage the sales made within in the dealership, by allowing the staff to input transactional data into the system.

Figure 1: Torque ITS Logo

Problem Background

The Dealer Performance Management System is an application designed to allow mangers of car dealerships to identify their profit potential. This is done by allowing the business manager of the car dealership to monitor the car sales and services. This is done by the staff at dealership (may it be sales personal or finance mangers) to input their sales details into the application. The car dealership companies have their own point of sale and vehicle databases in which they store the information pertaining to the sales transactions and vehicle details. DPMS allows the business manager of a car dealership to monitor those sales made by the sale persons, finance managers, and aftersales staff. Before the business manager can make use of DPMS, he or the staff first need to enter the data into the system. This re-entry of data from the POS system is a major draw-back of DPMS. To resolve this issue, the application needs to be able to extract/ import the information from other information systems such as POS. For my project I will need design an application that will allow the user of DPMS to integrate the data from their Point of Sales system. Also users of the system might also want to include data from there on site vehicle database, or query vehicle information from the vehicle transport database. The application needs to be flexible and uniform so that data from any number for data sources can be queried and data can be imported with minimal code change.

Project Motivation

The purpose of the project is to reuse the existing data from external systems, as to reduce the time for data entry and eliminate human error. Allowing the option of user import their existing data into the system will make the application compatible with existing greatly increase the appeal of the system and increase the sales more car dealerships to buy the DMPS product. Having the form fields auto-populated from the data bases will eliminate any human error that could occur.

Project Goal

The goal of the project is to provide a uniform query interface to a multitude of data sources, thereby freeing the casual user from having to locate data sources, interact with each one in isolation and manually combine the results.

The project

I have been given the possible ways the data could be extracted. The data that needs to be imported into the system can come from several data sources and formats. It is up to me how I design the interface to allow for the flexibility to import any data source in a consistent format. However the data sources can be split into major categories based on the frequency of data updates. The first type of data source contains data that changes frequently (i.e. every hour); this data is stored on the point of sale system. The DPMS application needs to be able to query these changes as soon as they have been made. The second category is data whose value if changed will not affect the overall outcome the application. This data is not mission critical and doesn't need to be updated constantly

Project Specifications

An application need to be developed that will allow Users choose which databases, and applications to search the data from. The User of the system must be able to query all these data sources and obtain the results in a consistent format so that the data can be imported into the DMPS application.

Specifications

- Users must be given the choice for which external data source they want to pull data from
- Users must be able to import parts of a log entry (or specific columns of data), from any data source they desire.
- The web service will provide you with a number of methods you can call. The search parameters offered will be known.
- Each data source will offer different parameters
- The search parameter will be run over multiple data sources; some of the sources will not have the same parameters for the search.
- List of alternatives should appear in case of multiple results (i.e. two existing quotes in POS for same vehicle).

Project Research

Data Integration universal challenge faced in applications that needs to query across multiple autonomous and heterogeneous data sources. Today's reality is that a large percentage of a data warehouse's total cost of ownership (TCO) is related to post development integration costs—that is, the on-going costs of loading source data into the data warehouse and distributing data from the data warehouse to downstream data stores. In the IT industry we are often presented with the problem of having data sets ready to be used, and yet being unable to use them. This happens due to several facts: the coded information does not satisfy some of the actual needs or the resource format is not appropriate. These situations may lead to incompatibilities between the data used by two applications that perform the same kind of task, preventing the cross reusability of those data sets or even their combination to form a richer set of data.

For my project research I have explored possible solutions to the problem described. I will look what data integration means, what challenges are faced by businesses when they attempt to integrate data with external systems. I will briefly go through common data integration patterns used currently in the IT sector. Finally I will compare and contrast these patterns and validity to my project.

Data integration is responsible for moving, cleansing and transforming set-based data—often very large data sets—from source(s) into the Production data area and then into the Consumption data area as shown in Figure 2.

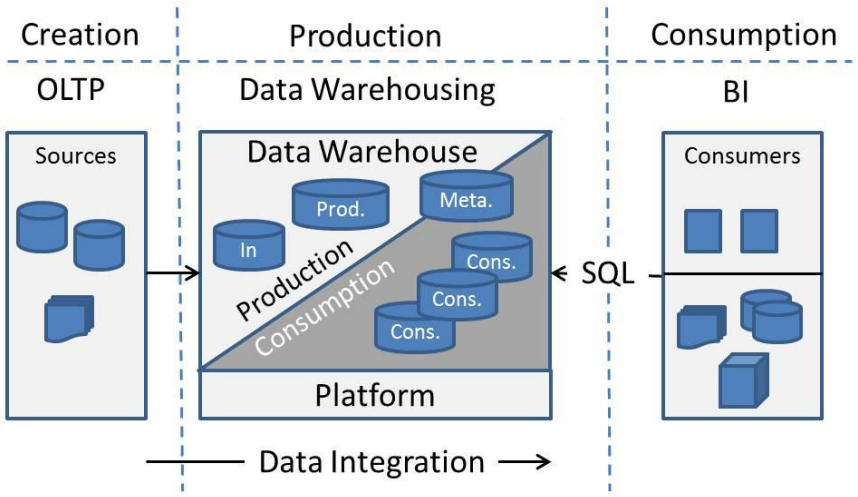

Figure 2: The role of data integration in a data warehouse project

Requirements for the data integration

The requirements for the data integration component include:

- **Trust** – Business consumers must be able to trust the results obtained from the data warehouse.
- **One version of the truth** – Consolidating heterogeneous sources into an integrated view supports business consumers' need for an enterprise-level view of data.
- **Current and historical views of data** – The ability to provide both a historical view of data as well as a recent view supports key business consumer activities such as trend analysis and predictive analysis.
- **Availability** – Data integration processes must not interfere with business consumers' ability to get results from the data warehouse.

Challenges for the data integration

The challenges for the data integration team in support of these requirements include:

- **Data quality** – The data integration team must promote data quality to a first-class citizen.
- **Transparency and auditability** – Even high-quality results will be questioned by business consumers. Providing complete transparency into how the data results were produced will be necessary to relieve business consumers' concerns around data quality.
- **Tracking history** – The ability to correctly report results at a particular period in time is an on-going challenge, particularly when there are adjustments to historical data.
- **Reducing processing times** – Efficiently processing very large volumes of data within ever shortening processing windows is an on-going challenge for the data integration team.

Data Integration Patterns

The industry has several well-known data integration patterns to meet these requirements and solve these challenges, and it's important for data warehouse practitioners to use the correct pattern for their implementation.

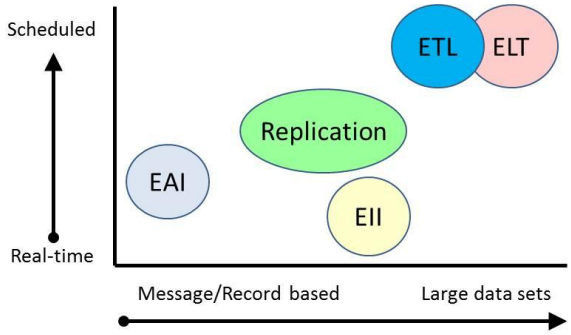

Figure 3: Positions the different integration options that are available.

The two axes in Figure 3 represent the main characteristics for classifying an integration pattern:

- **Timing** – Data integration can be a real-time operation or can occur on a scheduled basis.
- **Volumes** – Data integration can process one record at a time or data sets.

Integration patterns

The primary integration patterns are:

- **Enterprise Information Integration (EII)** – This pattern loosely couples multiple data stores by creating a semantic layer above the data stores and using industry-standard APIs such as ODBC, OLE-DB, and JDBC to access the data in real time.
- **Enterprise Application Integration (EAI)** – This pattern supports business processes and workflows that span multiple application systems. It typically works on a message-/event-based model and is not data-centric (i.e., it is parameter-based and does not pass more than one "record" at a time).
- **Extract, Transform, and Load (ETL)** – This pattern extracts data from sources, transforms the data in memory and then loads it into a destination.
- **Extract, Load, and Transform (ELT)** – This pattern first extracts data from sources and loads it into a relational database. The transformation is then performed within the relational database and not in memory.
- **Replication** – This is a relational database feature that detects changed records in a source and pushes the changed records to a destination or destinations. The destination is typically a mirror of the source, meaning that the data is not transformed on the way from source to destination.

Overnight Data Integration patterns

Data integration, which frequently deals with very large data sets, has traditionally been scheduled to run on a nightly basis during off hours. In this scenario, the following has held true for the different patterns:

- EII is not commonly used in data warehouses because of performance issues. The size and data volumes of data warehouses prohibit the real-time federation of diverse data stores, which is the technique employed by the EII pattern.
- EAI is not used in data warehouses because the volume of the data sets results in poor performance for message-/event-based applications.
- ETL is the most widely used integration pattern for data warehouses today.
- ELT is seen mostly in legacy data warehouse implementations and in very large data warehouse implementations where the data volumes exceed the memory required by the ETL pattern.
- Replication, used to extract data from sources, is used in conjunction with an ETL or ELT pattern for some data warehouse implementations.
    - The decision to use replication can be based on a variety of factors, including the lack of a last changed column or when direct access to source data is not allowed.

Real-Time Data Integration

The growing need for real-time or near real-time reporting outside of the line of business database; organizations are increasingly running some data integration processes more frequently—some close to real time. To efficiently capture net changes for near real-time data integration, more and more companies are turning to the following solutions:

- Replication to push data out for further processing in near real time when the consumer requires recent data (replication is also useful when the source system doesn't have columns that the ETL or ELT tool can used to detect changed records)
- Relational databases' additional capabilities to detect and store record changes, such as SQL Server 2008 Change Data Capture (CDC) which is based upon the same underlying technology used by replication.
- Incremental change logic within an ETL or ELT pattern (as long as the source table has a date or incrementing column that can be used to detect changes)

Which Pattern Should I Use?

Typically, a data warehouse should use either ETL or ELT to meet its data integration needs. The costs of maintaining replication, especially when re-synchronizing the replication process is required, makes it a less attractive alternative for extracting data from sources. However, hybrid approaches such as ETL/ELT combined with source system net-change detection capabilities may be required for near real-time data.

Data Integration Paradigms (ETL and ELT)

ETL products populate one or more destinations with data obtained from one or more sources. The simplest pattern is where one source loads one destination, as illustrated in Figure 4.

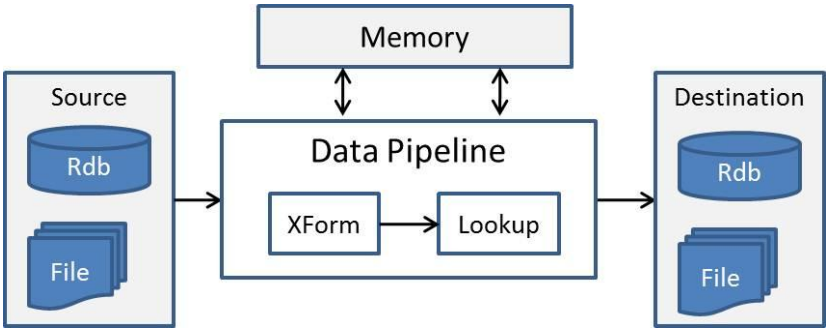

Figure 4: Simple ETL data flow

Processing steps in ETL data flow

The processing steps are as follows:

1. The ETL tool retrieves data sets from the source, using SQL for relational sources or another interface for file sources.
2. The data set enters the data pipeline, which applies transformations to the data one record at a time. Intermediate data results are stored in memory.
3. The transformed data is then persisted into the destination.

Advantages for ETL data flow

Advantages to this process are that:

- Procedural programming constructs support complex transformations.
- Storing intermediate results in memory is faster than persisting to disk.
- Inserts are efficiently processed using bulk-insert techniques.

Disadvantages for ETL data flow

However, the disadvantages include the following:

- Very large data sets could overwhelm the memory available to the data pipeline.
- Updates are more efficient using set-based processing—meaning using one SQL UPDATE statement for all records, not one UPDATE per each record.

Figure 5 shows an example of an SQL Server Integration Services (SSIS) data flow that performs transformation processes (joining, grouping, calculating metrics, and so on) in the pipeline. This data flow has the advantage of leveraging the memory resources of the server and can perform many of the transformation tasks in parallel. However, when memory is limited or the data set needs to entirely fit in memory, the processes will slow down.

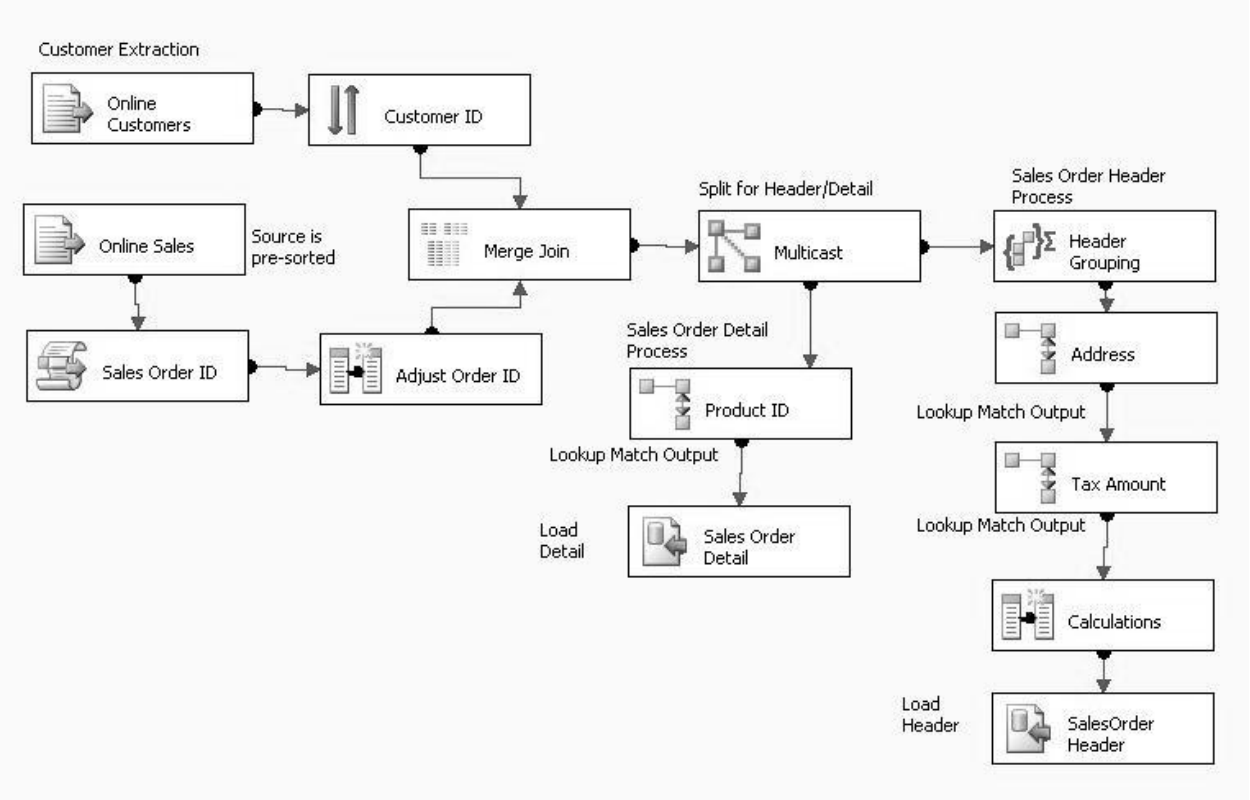

Figure 5: SSIS data flow example

Extract, Load, and Transform—also moves data from sources to destinations. ELT relies on the relational engine for its transformations. Figure 6 shows a simple example of ELT processing.

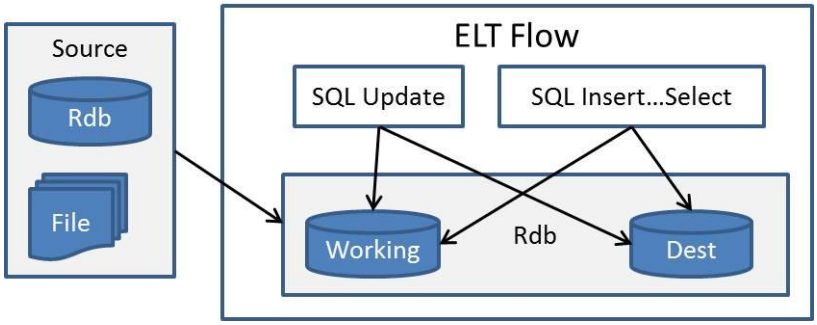

Figure 6: Simple ELT data flow

Processing steps in the ELT data flow

The processing steps in the ELT data flow are as follows:

1. Source data is loaded either directly into the destination or into an intermediate working table when more complex processing is required. Note that transformations can be implemented within the source SQL Select statement.
2. Transformations are optionally applied using the SQL Update command. More complex transformations may require multiple Updates for one table.
3. Transformations and Lookups are implemented within the SQL Insert…Select statement that loads the destination from the working area.
4. Updates for complex transformations and consolidations are then applied to the destination.

Advantages of ELT Data Flow

The advantages of this process include the following:

- The power of the relational database system can be utilized for very large data sets. Although, note that this processing will impact other activity within the relational database.
- SQL is a very mature language that translates into a greater pool of developers than ETL tools would.

Disadvantages of ELT Data Flow

However, you need to consider these disadvantages:

- As just noted, ELT places a greater load on the relational database system.
- You will also see more disk activity because all intermediate results are stored within a table, not memory.
- Implementing transformations and consolidations using one or more SQL Updates is more inefficient than the ETL equivalents, which make only one pass through the data and apply the changes to the destination using a single SQL statement rather than multiple ones.
- Complex transformations can exceed the capabilities of the SQL Insert and Updates statements because transformations occur at the record level not the data set level. When this occurs, SQL cursors are used to iterate over the data set, which results in decreased performance and hard-to-maintain SQL code.
- For a given transformation, the processes applied are often serialized in nature and add to the overall processing time.

Figure 7 shows the SSIS control flow used in more of an ELT-type operation. You can identify ELT-type operations by their multiple linear tasks, which perform either Execute SQL Tasks or straight data loads using a few working tables.

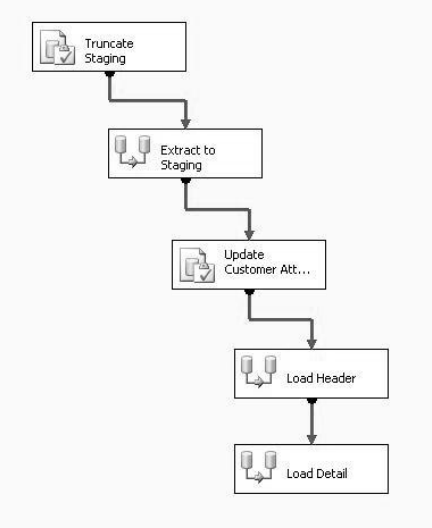

Figure 7: SSIS control flow ELT process

Which Should I Use for My Implementation?

The decision about whether to use an ETL or ELT pattern for a SQL Server data integration solution is based on the following considerations.

Use ETL when…

- Working with flat files and non-relational sources. ETL tools have readers which can access non-relational sources like flat files and XML files. ELT tools leverage the SQL language which requires that the data be first loaded into a relational database.
- This is a new data integration project or the current first-generation implementation is hard to manage and maintain. The visual workflows for tasks and data flows make the process easier to understand by non-developers.
- The transformations are complex. ETL's ability to apply complex transformations and business rules far exceeds the abilities of one set-based SQL statement. Many legacy ELT solutions have become unmanageable over time because of cursor-based logic and multiple Update operations used to implement complex transformations.

Use ELT when…

- The data volumes being processed are very large. Huge data sets may exhaust the available memory for an ETL approach. Remember that the ETL data pipeline uses in-memory buffers to hold intermediate data results.
- The source and destination data is on the same server and the transformations are very simple. A SQL-centric ELT solution is a reasonable choice when the current database development team is not trained on SSIS. But keep in mind that complex transformations can easily translate into poorly performing, unmaintainable data integration code.

In my research I have found that the most suitable data integration pattern would be ETL, for the same reasons described above. In my final report the precise steps taken to implement this pattern. My future research on this topic will consist of detecting when changes have been made, how to make the data integration into a scientific manageable process. I will need to research about batch processing so that I will be able to integrate data from sources that don't direct access to live data. Further study needs to be done in ensuring that data quality is maintained: so data profiling, cleansing and reconciliation will also be area I will look into.

Development Plan

Extracting the Data

There are two ways the data can be extracted from external systems. The car dealerships can offer a web service to query their point of sale of dealer management database. The second option is that the dealerships, car manufactures, and other vehicle databases daily offer a database dump of the database. The option is preferred but however is not always possible for dealerships to create a web services to access their system due to the additional cost of development.

1. Ad-hoc Web Service

When a sales person in the car dealership makes a sale the transaction is stored in the POS. These kinds of changes need to be incorporated in the DPMS as soon as it happens. A solution to this is pulling the data from the external system via AD Hoc web service. In this approach the underlying P.O.S (Point of Sales) system will have a web service which returns data. For example when the user enter a registration number or VIN number the system will automatically trigger a web-service that will send a request to the external POS system, This query will then run on the systems database thus providing the user latest information. Which in turn will respond with corresponding vehicle detail, this data can then is used to populate the rest of the fields.

2. Using Data Dump

The second means of gathering data is through an overnight process where the external system in our case Auto link, which a data interface tool used by dealer to upload mass vehicle information to databases, this is same system currently being used by Trade me. The plan is use the data dump created by the vehicle database, Auto link to update our database. This means that all the latest vehicle details can be imported to into our system automatically without any user input; this eliminates any human error or data inaccuracy. In this case when the user enters an identifier the vehicle details will be looked up within the local data store and populate the remaining fields.

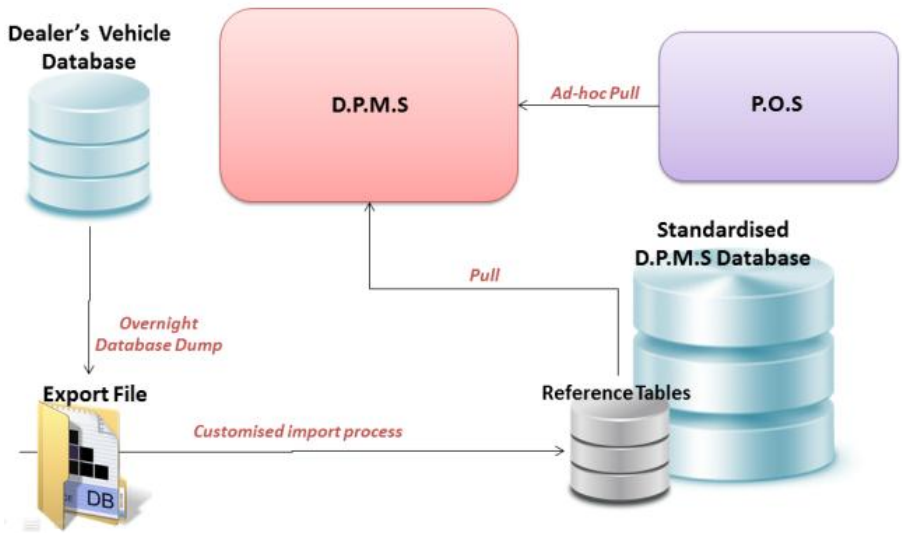

Figure 8 shows how the data from the different data sources will be extracted into the application.

Figure 8: DPMS Data Integration Architecture

Import Interface Design

Design & Implement the Visual Interface

For the importing the data, I created a design of how the interface should look. The interface will allow the user to specify whether they want to import or manually enter the, Depending on the user configuration the system will use one of the two ways described above will used to retrieve the vehicle details.

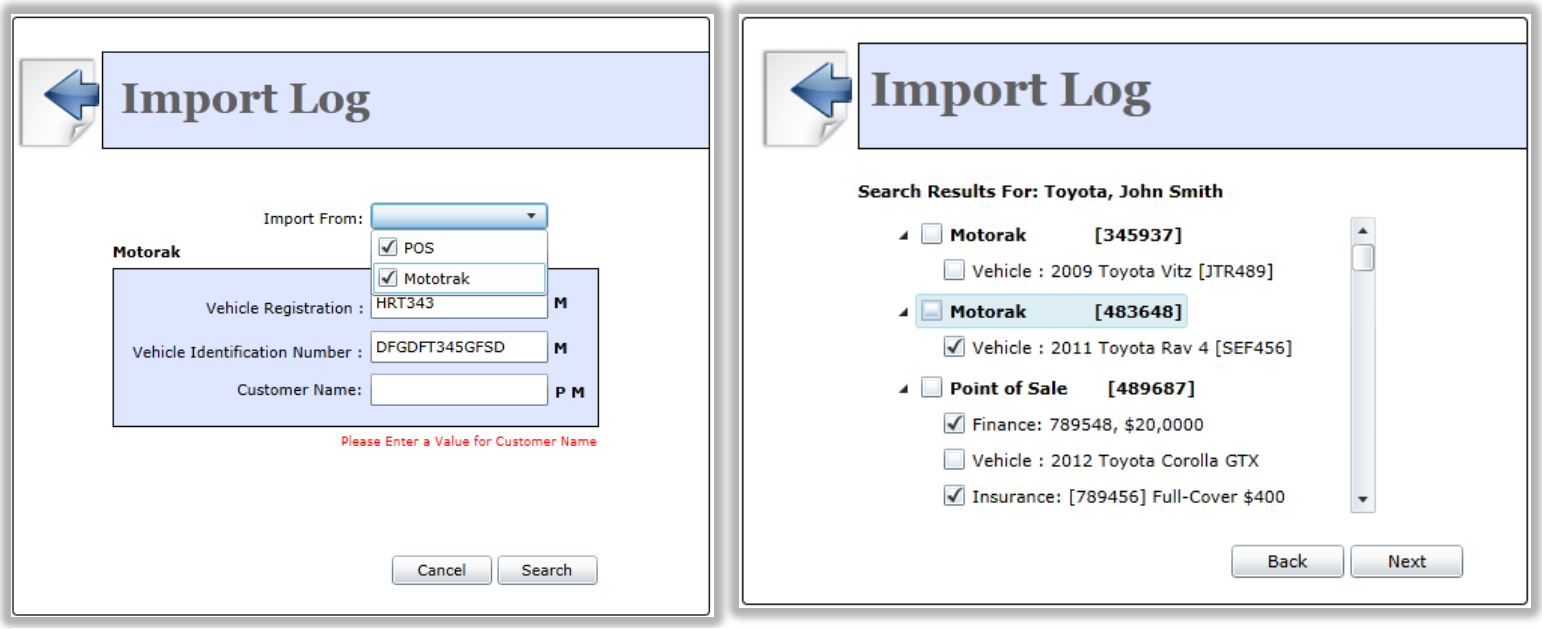

Figure 9: Import Interface: First two mock screens

Use Case 1: Creating and Editing Sales / Finance Entries

The user will create / Edit Entry they will be prompted with a popup, giving them the option whether to import data; the user will have the option to click cancel. If they choose to import (and most cases this should be the case), the user will go through a small wizard which will allow them to pick the entries they want and from where to pick the data from.

1. The user will pick the data sources they want to search from; they may choose either to search entries from e.g. Auto Link or POS or both.
2. Depending on what data sources they choose the identifiers' fields will be displayed*. The identifiers entered will be passed to POS web Service & Auto Link and cumulative results will be returned in a grid.

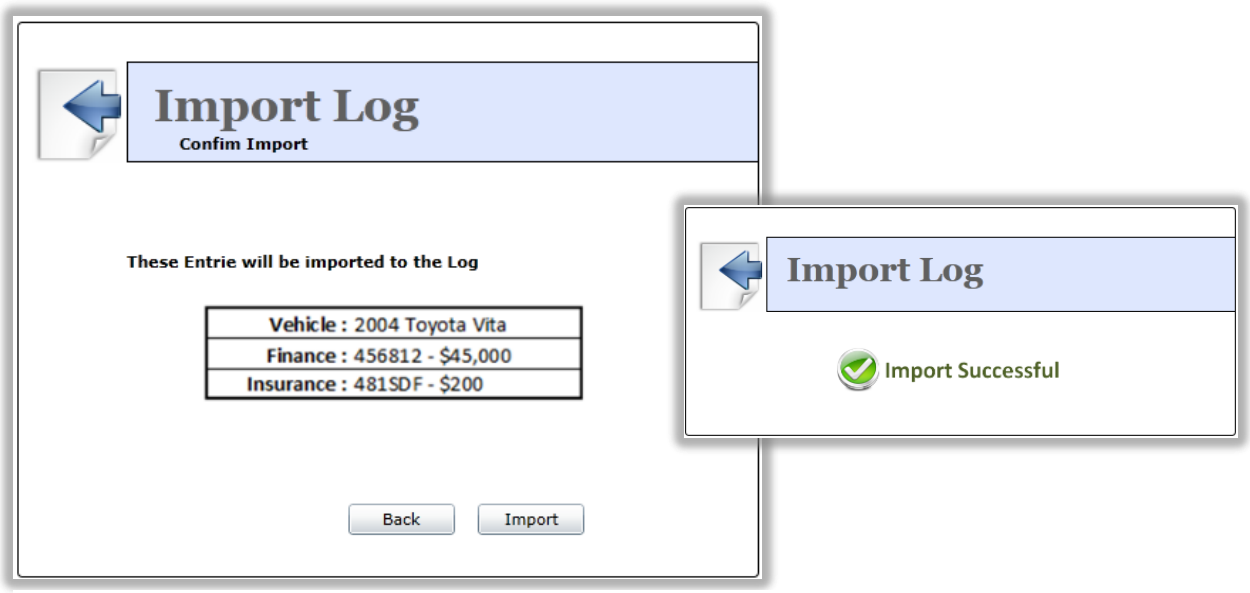

Figure 10: Import Interface design screens

3. From the tree the user will select one entry, the user has the option to select the whole entry or select sub entries that relate to the entry.(maybe even individual fields as a future improvement)
    a. The grid shows the data sources that returned matches to the query. Each data source has sub-entries that match the query. The user can only select one sub-entry, from the grid. For example, the user may only select vehicle information coming from Auto Link or the Point of Sales system. The user will not be allowed to select a vehicle entry coming from Point of Sales if a vehicle entry from Auto Link is already selected. Once the user is satisfied with their data import selection, they must click 'Import'.
4. In this screen they will get a final chance to go back and make any changes, before all the data is imported into the log entry Form.
5. If the import is successful, the users will be shown the screen on the right. The Screen on the right will appear as popup that disappears automatically after a second or so.
6. The selected entries will be imported into our current interface (i.e. fields on user screen will be populated according to imported data) where it can be further edited and saved. Only those input fields which are not disabled/read-only can be imported. Other values are ignored.

Work Done

Date	Work
Week 1	Chose the project I will be working on this year
Week 2	This week I met with the company, we discussed about the company background, and a brief overview of the project
Week 3	Discussed the requirements of the project, and more details of the system were discussed
Week 4	The week I had my introductory presentation for BTech, which I discussed my project
Week 5	I worked the specifications document that outlined the scope and finalised the requirements of the project
Week 6	This week I was working the proposal the import interface and research some possible solutions for integration of data
Week 7	Been working on the proposal and more research was done with regards to MVC.NET
Week 8	This week I have been working on cases and
Week 9	I have started giving thought the UML diagrams required for the project
Week 10	Been working on the User interface designs for the import interface
Week 11	Started Implementing the Class diagrams & began work on the interface for the web service
Week 12	Working on Developing the interface and classes

Future Work

So far I have completed the class designs and Interface designs. My main focus and work in this project so far has been working creating the interface to which the existing system will extract data from the external systems via a web service.

For the first half of the second semester I will hard on completing the user interface and front end for the application. Once I am able to consume and display data form the web service, I would have the basic workflow for the application completed. I will the work on writing the scripts that will convert the excel and CSV flat files into the reference tables have the basic work flow working

- The next step for me is to write to write scripts that will convert data from excel files and CSV files and import them into SQL server.
- The scripts will need to run ETL (extract, transform, load) on data so the data is a standardised format reconcile
- Finally, I will need to test whether the data is seamlessly integrated with the current system.

Bibliography

[1] G. Hophe, “Enterprise Integration Patterns,” 2002.

[2] [Online].

[3] “Enterprise Information Integration: A New Definition,” [Online]. Available: http://www.information-management.com/news/1009669-1.html.

[4] “Understanding Enterprise Application Integration - The Benefits of ESB for EAI,” [Online]. Available: http://www.mulesoft.org/enterprise-application-integration-eai-and-esb.

[5] “A Flexible Model for Data Integration,” [Online]. Available: http://msdn.microsoft.com/en-us/library/bb245674.aspx.

[6] R. Yin, “An Effecient Data Service Layer”.

[7] R. Ribeiro, “How to Integrate data from different sources”.