Migrating E-commerce Database Applications to an Enterprise Java Environment

Terence C. Lau
Centre for Advanced Studies
IBM Canada Laboratory
Lautc@ca.ibm.com

Jianguo Lu
Department of Computer Science
University of Toronto
jglu@cs.toronto.edu

Erik Hedges
Department of Electrical & Computer Engineering
University of Waterloo
ehedges@swen.uwaterloo.ca

Emily (Xuemin) Xing
E-Commerce Product Development
IBM Canada Laboratory
exing@ca.ibm.com

ABSTRACT

As technology evolves over time, a common problem is the migration of software applications from one technology base to another. This paper is a practical experience report based on IBM Net.Commerce to WebSphere Commerce Suite (WCS) migration. It identifies the problems and issues in the migration of applications using traditional database access (SQL) to applications using the Enterprise Java Bean (EJB) programming model, and presents a practical methodology in facilitating such migration. It also describes a tool built on this methodology that has been released on IBM's alphaWorks site. From the experience so gained, this paper points to a number of future enhancement areas in the methodology and associated technology research.

Keywords: E-commerce, Migration, Database re-engineering, Enterprise Javabean, SQL, Net.Data, JSP, Relational-object mapping.

1 Introduction

Many new large distributed applications are being developed using the Java 2 Enterprise Edition (J2EE) platform[1]. The J2EE platform allows developers to create robust three-tier applications by providing middle tier services to communicate with a variety of clients and backend services.

A middle-tier server such as the EJB server has significant advantages over a classical two-tier client/server model. The two-tier model requires the client to have extensive knowledge of how to access backend systems (e.g. a relational database) and forces the client to implement any necessary business logic to manipulate data retrieved from the backend system. The middle-tier is designed to shield clients from interaction with backend system and allows for the development of thin-clients that don't have to perform any heavy processing.

J2EE also has significant advantages over creating your own middle tier. Developing a middle tier server without the J2EE framework would require the developer to worry about client connectivity, database access, and transaction management among other issues. J2EE provides services to take care of these issues and lets the developer concentrate on implementing business logic.

The J2EE platform is also popular because it is based on open standards (XML, Java, Java Naming and Directory Interface etc.) and is being adopted by a variety of third party solution providers (Broadvision, IBM, etc.) in their web servers and distributed application development platforms.

We have examined the problem of migrating clients of client/server applications that use SQL statements to retrieve data to a new Enterprise Java based package with a developed programming model to access similar data. This problem becomes significantly more difficult when extensive schema changes are made between the old and new versions. Without significant knowledge of the developed client components and the schema changes it would take the developer a long time to figure out which component contains access to similar data in the new version. To figure out which component to use the developer would have to consult system documentation and potentially dig into the programmed source of the components to understand relationships. This is not an ideal situation when the goal is to have simple client components that present a layer of transparency to the developer.

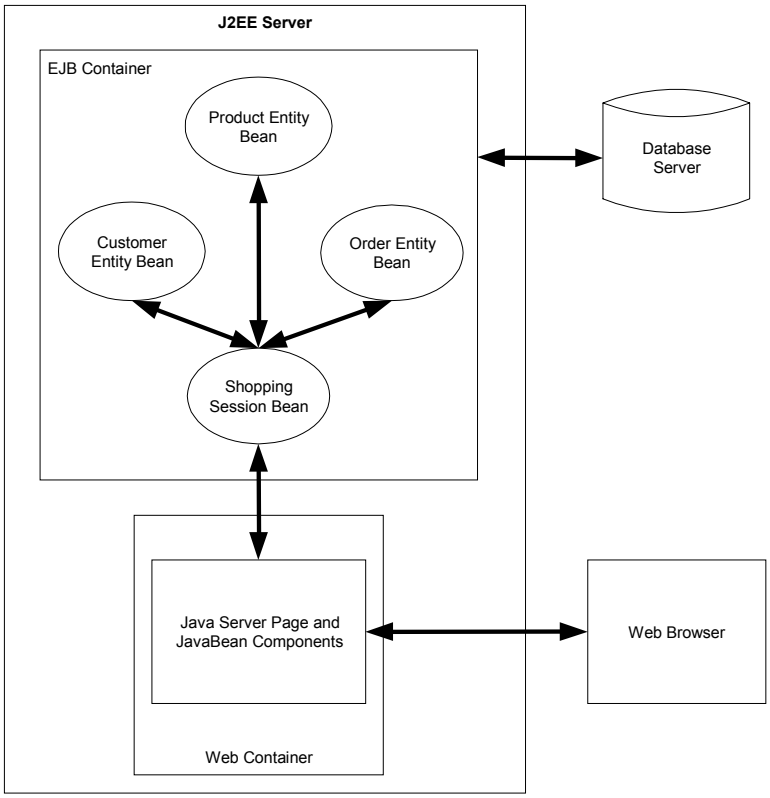

Figure 1: J2EE Architecture Example

We present a practical methodology to aid the client migration problem and our experience in building a Net.Data to JavaServer Page Helper Tool that is currently being used by developers of migrating e-commerce websites developed using IBM's WebSphere Commerce Suite, from its earlier versions to its most recent version (IBM WCS V5.1).

There are several contributions of our work. First, we developed a methodology and a migration tool that facilitate the transition of IBM Websphere Commerce Suite from one version to another. The tool has been released in IBM alphaWorks. Second, based on the experiment, we identified common problems for migrating legacy systems to EJB-based architecture for future works.

2 Background

2.1 Enterprise Java

The J2EE platform is an architecture for developing, deploying, and executing applications in a distributed environment.[1] A general J2EE architecture layout is shown in figure 1.

The J2EE Server provides all the necessary services to enable communication between the clients, the middle tier services and the database server (e.g. HTTP communication enabling clients to invoke a Java Server Page).

Enterprise Beans

Enterprise beans are server-side components that run in an Enterprise Java Bean (EJB) container[4]. The Enterprise Beans control data access and contain the business logic of the server

application. The container manages features important to a distributed application such as transaction management, security, database connection pooling etc. There are two kinds of enterprise beans: session beans and entity beans.

An entity bean is a persistent object that represents an item in a storage system such as a Relational Database Management System. Using the example from , Product Entity Bean represents a row in the product table of the database. The entity bean provides methods to select, add, modify, and delete underlying data. Methods that return a set of rows we call "Finder Methods" (e.g. search for a particular record based on matching the primary key).

Session beans on the other hand are not persistent and may have only one client. They are instantiated at the request of the client and are terminated when the client session terminates. They perform a task for that client and do not directly represent shared data in the database. Session beans can however, access and update such data.

Java Server Pages

Java Server Pages (JSPs) enable Enterprise Java applications to create dynamic content for browser-based clients. JSPs are a presentation-centric method of developing servlets (Java programs that are run by a web server and whose output can be directed to a client browser). JSPs support a reusable component model using JavaBean components and Custom actions (not explored here).

Using JavaBean components web page designer can focus on presentation while an application developer can develop specific components to process data and return data to be used in the page. JavaBean components can also be used elsewhere in the application, as they are reusable and portable.

2.2 Net.Commerce and Websphere Commerce Suite

IBM WebSphere Commerce Suite and its earlier versions (Net.Commerce) [3] are platforms for building E-Commerce applications, supporting functionalities ranging from product catalogue browsing, payment processing, to product promotion, auction, and etc. Started in 1995, Net.Commerce has gone through five major revisions and was renamed as Websphere Commerce Suite (WCS). The most recent revision (IBM WCS 5.1, November 2000) extensively enhanced its underlying technology, from the database schema to the programming model. With emphasis on clear programming structure and conformance with the Enterprise JavaBean (EJB) model, IBM WCS V5.1 has improved significantly from its earlier versions. Existing customers who want to transit to the new version of IBM WCS will need the help of tools and guidance of some methodology to move their legacy e-commerce applications to the new environment.

The IBM Net.Commerce and WCS V.4 Programming Model

The early versions of Net.Commerce and IBM WCS V.4 have a programming model built on C++ and use commands, tasks and overridable functions. Commands are C++ components servicing major functions, like placing an order. Tasks represent subunits of work within a command, such as checking inventory or calculating the price for an order. Overridable functions are the C++ components that actually perform the work of the tasks. Customers usually implement their own business logic by replacing the WebSphere Commerce Suite-provided overridable functions with their own. When results and responses are presented to users, Net.Data macros are invoked to access the database, retrieve the appropriate information and then present it on Web pages.

The IBM WCS 5.1 Programming Model

The WebSphere Commerce Suite, Version 5.1 programming model is based on WebSphere Application Server, Java technology and EJB. The Net.Data display functions in Net.Commerce and IBM WCS Version 4.1 are completely replaced by JSP, and the C++ command/ task/ overridable function structure is replaced by a single WebSphere command structure that uses Java technology and EJB.

The Transition Tasks

The transition of Net.Commerce and IBM WCS V.4 to WCS V.5 consists of three parts:

(1) Migrate the software stacks

The software stacks include the system and application software packaged with WebSphere Commerce Suite, Version 5.1 to provide basic system functions and third-party application function. They include the operating system, the database system, the web server, the security system etc.

(2) Migrate the IBM WCS infrastructure
The IBM WCS infrastructure consists of the basic components shipped with the product. Customers build their specific solutions on this infrastructure. The infrastructure includes: the tools for building a website based on WebSphere Commerce Suite, the runtime infrastructure, the IBM WCS database schema, the IBM WCS class library and object entities.

(3) Migrate the customer assets.
Customer assets are those assets accumulated, populated, customized or extended by customers in their commerce sites. These assets include database information, pages designed specifically for the Web site, database schema extensions, and new or customized business logic in commands and overridable functions. [3]

The transition to IBM WCS Version 5.1 involves the conversion of the following major assets:

(1) Data is migrated to the WebSphere Commerce Suite Version 5.1 database format and schema, and accessed using the new object models of EJB and dataBeans.

(2) Net.Data macros are replaced by JSP. The SQL in Net.Data macros are replaced with JSP calls to WebSphere Commerce Suite, Version 5.1 data or business EJB. Business logic within a Net.Data macro should be moved into a command to isolate view from model.

(3) Business logic will be converted from C++ to Java commands: Overridable functions are replaced by EJB task commands. C++ commands are replaced by EJB controller commands.

The work in this paper has a particular relevance in (2) of the above list, namely the migration of the Net.Data macros to JSP.

3 Generalized Process and System

Our process contains two main phases: (A) Information collection and (B) Analysis of Database Application and JavaBean Recommendation.

During Phase (A) the following information is collected and mappings are established between common elements:

(1) Database schema mapping information between the source and target system

(2) Command mapping information between the source and target system

We define a command as a component of the source system that accomplishes a particular task and has a comparable counterpart in the target system (e.g. addUser command).

(3) Target system EJB Object (entity beans) and database relation information

(4) Target system JavaBean components and Enterprise bean relations

(5) Target system command and JavaBean relations

During Phase (B) the following processes take place:

(1) An application unit (e.g. a source file) with database access is parsed and converted into a representation that only stores the SQL statements contained in the application unit

(2) Each SQL statement is analysed to see which tables from the source schema are used

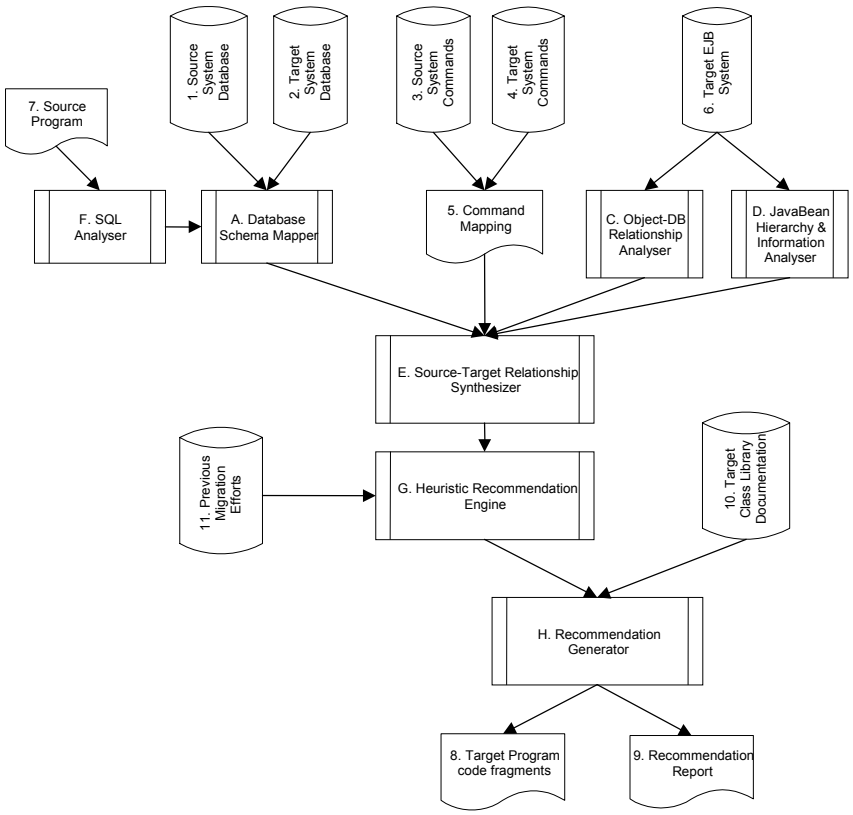

Figure 2: System Diagram

(3) Identify what those columns map to in the target schema from (A1)

(4) Identify EJB Objects that use those columns from the target schema using (A4)

(5) Identify JavaBeans that make use of the objects from (B4) using (A4)

(6) If there are commands that run before or after the SQL execution in the source system, find the equivalents of these commands in the target system using (A2)

(7) Using this information (B6) the set of JavaBeans can be reduced if the user can identify which JavaBeans are used by these commands in the target system.

Figure 2 depicts in detail the various data sources and processes involved in our migration-aid system. We will now explain the purpose and relation between the various elements in the system diagram figure 2.

Prior to the first use of the tool the target application source code is analysed. The Object-DB analyser (Item C) extracts information from the target application to determine relations between Entity Beans and the underlying database. This is accomplished by parsing through relevant source code from the target application and needs only to be run once providing the underlying source remains static. In general, an Entity Bean will exist for each table in the database and provide read/write access to data elements as well as to perform searches on the data. We call this relationship the Primary relationship between the table and the related Entity Bean (a secondary relationship exists if the Entity Bean uses data from other tables in any search methods). All of the "finder methods" of the entity beans are also recorded.

We then examine the client JavaBean Components to determine their usage of entity beans (Item D) and any containment relationships that exist between the JavaBean Components. This process is also run once and needs only to be

re-run when the underlying JavaBean code changes.

During initialisation of the migration tool all the underlying reasoning data necessary is read into the system. The Database Schema Mapper (Item A) process accepts as input the source-to-target database schema mapping relationship as specified in a configuration file. This relationship states what target database table/column(s) map to what target database/column. The process generates an internal structure representing this mapping relationship. The relationship between the source database tables and target database tables can be 1-to-1, 1-to-many, or many-to-1.

The Source-to-Target Relationship Synthesizer (Item E) amalgamates the underlying reasoning data so that it is possible given a table from the source application to determine all the possible JavaBean components from the target application that could have access to similar data in the target system.

After initialisation the tool accepts as input a source file and extracts all the SQL statements contained in the file. The SQL Analyser (Item F) extracts the column(s) and tables used in a particular SQL statement. The following example illustrates the recommendation process:

Given the following SQL statement:

```
SELECT A,B,G,H
FROM X,Y
WHERE X.A = Y.G
```

The SQL Parser extracts the tables from this SELECT statement (X,Y). Applying the database schema mapping information to our tables we get:

```
(source tables) X,Y
-> X',Y',Z' (target tables)
```

We then determine the entity beans associated with the target tables:

```
X',Y',Z' (Target system table)
->  X_EntityBean,
    Y_EntityBean,
    Z_EntityBean
```

We then determine the set of JavaBean components that use these entity beans:

```
X_EntityBean, Y_EntityBean, Z_EntityBean
-> A_JavaBean,
   B_JavaBean
```

The number of entity beans that a particular JavaBean component uses is used to rank the recommendations. The logic being that the more data coverage that the component has, the better chance that it will have a combination of methods that will return data similar to the data retrieved from the SQL statement.

This set of ranked JavaBean components is passed off to the Recommendation Generator (Item H) that produces the output to the user in the form of a report or to a GUI.

4 Net.Data to JSP Migration Helper Tool

We have developed a tool to aid the migration of Net.Data macro files used in client e-commerce projects to Java Server Pages. The tool was developed following the architecture and processing model outlined in the previous section.

IBM Websphere Commerce Suite provides a complete set of tools and interfaces to develop a large-scale e-commerce project (both business to consumer and business to business applications). The most recent version of IBM WCS (V5.1) released this year adopts a new technology base that represents a commitment to an open architecture based on the Java and Enterprise Java programming model. Previous versions of IBM WCS were based on an architecture using C^{++} commands and Net.Data macros for presenting HTML to the user. [3]

The process of converting a commerce site using Net.Data macros to a site using JSP templates is by no means an automated task. There are many design decisions and code conversions to be carried out by the developers responsible for the migration (particularly when developing business logic components). Thus Net.Data Migration Helper Tool is a reference tool that helps the development team plan their Net.Data migration and gives them guidance on where to begin. The tool can be used at two points in the migration process:

(1) When the migration planners are trying to determine how much work needs to be done on each file in order to create a project schedule for migration. The tool will give a high level view of how many functions each

Net.Data file has, and how many standard and customised tables are involved.

(2) When the developer is working on creating a particular JSP template to replace a Net.Data file. The tool acts as a reference for the mapping between IBM WCS Version 4 and Version 5 tables, for the functions in the file and recommended beans that should be used to perform the same function in a JSP.

4.1 Net.Data Scripting language

IBM's Net.Data product enables developers to create dynamic web pages using data from relational database systems and other back-end systems. Net.Data has a macro language that enables a developer to specify the layout of Web pages, calls functions that are defined by the

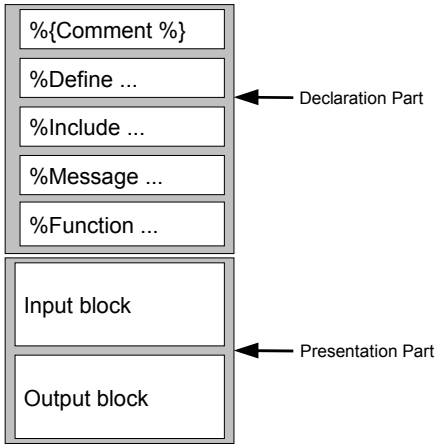

Figure 3 - Net.Data language

macro, and defines variables and functions.[2]

Net.Data macros contain two parts: the declaration part and the presentation part. Figure 3The illustrates the structure of a Net.Data macro.

The function block is what we are most interested in. A common function block has the following layout:

```
%function(dtw_odbc) name(){
  SELECT distinct safname, samname,
salname, shrfnbr, satitle
  FROM shopper, shaddr
  WHERE sashnbr=shrfnbr and
shlogid='$(SESSION_ID)' and
sanick='$(SESSION_ID)'
  %REPORT{
    %ROW{
      <center>
        HTML formatting information for
rows returned from the SELECT statement
      </center>
    %}
  %}
%}
```

The SQL statements that we analyse are generally found in the function blocks of a macro.

4.2 Objective of the Tool and Rationale

Because of the significant differences between Net.Data and JSP, and the fact that customers might want to make functional changes during the migration, the objective of the tool is not to perform complete automatic conversion of one Net.Data macro to a JSP. Rather, it is a "helper tool". Although the conversion process will require human involvement, the tool will significantly reduce the total effort of the developers. Developers' experience indicates that a good recommendation list could save significant amount of effort and time during the first part of the conversion process.

The rationale for this approach is two-fold:

First, from an implementation feasibility point of view, an automated translator from Net.Data to Java is extremely difficult. Net.Data macros and JSPs are very different in form and implementation. It is not easy to pinpoint ahead the mapping from an SQL in a Net.Data macro to a particular data bean. Allowing the users to participate in the mapping will make the tool much more useful.

Secondly, because of the many enhancements in the WCS V5 that customers want to take advantage of, they very likely prefer not to perform an automated translation of the Net.Data. The database schema has extensive enhancement from IBM WCS V4 to V5, as well as the function and the design of some major components such as catalog/product display, pricing, and ordering is quite different in V5. It is neither too practical nor useful to capture and fix this information in one shot, because customers very likely will customize the product and need to modify this information. A more practical approach is to let the helper tool to provide more general recommendation initially, and allow users to add

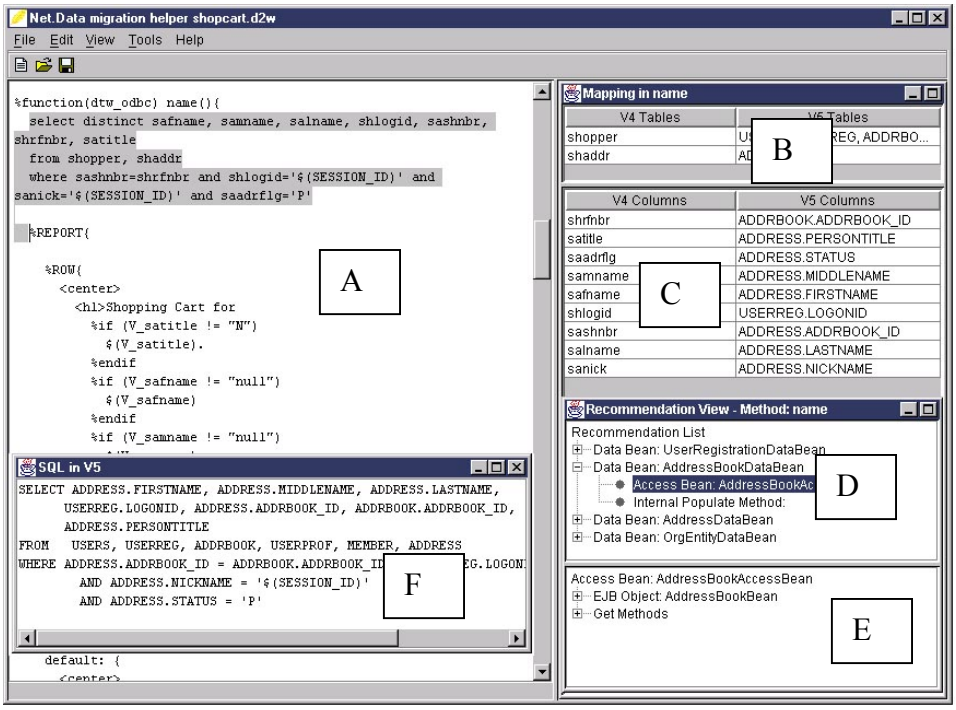

Figure 4: Tool Window View

in their "wisdom" to build up the tool's knowledge.

This approach also paves the way for a potential powerful function, namely a learning capability such that as the customers use the tool and indicate their choices, the information will be captured so that more specific recommendations can be suggested in future uses. Customers usually have collections of similar Net.Data macros. After a lead developer trains the tool for one macro, it can then give out very useful recommendation for the rest of the collection to other developers. Similarly, this tool can be used to capture the experience of one customer situation and pass it on to other similar situation. The learning capability is not included in the first release of the tool.

4.3 Tool Features

Figure 4 is a screen-shot of the tool.

The left-hand pane is a full editor in to which the Net.Data macro loads. The tool enables the user to cycle through each of the SQL statements found in the macro. The right-hand pane shows the table and column mapping information and the recommended set of data beans of the active SQL statement.

The tool can produce a report for the loaded Net.Data macro that lists all the SQL statements in the macro, the function it came from, and the recommended data beans and their related access beans. It also points customised tables (not in the standard IBM WCS schema) used in the SQL statements (used for project sizing).

As an example, when a Net.Data page is loaded into the tool the macro is parsed to extract all the SQL statements. After that the SQL statement is parsed and relevant information is extracted, such as columns and tables that are used. A typical sequence of activities by a user is as follows:

(1) Select the SQL statement in the Net.Data macro (as depicted in the top-left pane).

(2) Note the V4 -V5 table correspondence in B

(3) Note the V4 -V5 column correspondence in C

(4) Open recommendation pane, note related DataBeans in D.

(5) Go through the list, select DataBeans of interest in D and note detailed information of related DataBeans in E

(6) Using information in F, compare against getter methods in E to narrow down the DataBean selection.

After going through the above steps, the possible output JSP code corresponding to the name() function in Net.Data file in Pane A, as deduced from Panes D and E, might be:

```
……
<jsp:useBean id="addressbook"
class="com.ibm.commerce.user.beans.Address
BookDataBean" scope="page" />
<%com.ibm.commerce.beans.DataBeanManager.a
ctivate(addressbook, request); %>
</jsp:useBean>
<!-- HTML content -->
……
<center>
<h1> Shopping Cart for
<%=addressbook.getTitle() %>
<%= addressbook.getFirstName() %>
<%= addressbook.getMiddleName() %>
<%= addressbook.getLastName() %></h1>
……
```

4.4 Tool Deployment

The most current release of the tool was released in June 2001 on IBM's alphaWorks website[17]. Our primary target audience at this time is members of the IBM WCS Service Teams embarking on migration projects. They will be primarily using the tool in the project pricing and planning stages, and in facilitating the Net.Data macros migration. It is also available to anyone on an "as-is" basis.

Our database schema mapping information is based on a stylesheet defined in XSL (Extensible Stylesheet Language) that transforms IBM WCS V4.1 data in XML format to the new V5.1 schema.

Our data collection process extricates the relationships between the data beans, access beans, and entity beans. Also, it establishes the connection between the database and the enterprise beans. Data beans are the JavaBean components that our tool recommends to the user. In order to fully understand how to use the particular data bean and to confirm that it is the right object for the task we provide a link to the JavaDoc documentation of the data bean.

5 Related Work

There are several tools and methodologies to migrate EJB applications from one platform to another[6] [8]. In this paper, we further address the migration of database applications to EJB architectures, and the specific requirements for the WebSphere Commerce Suite.

There are also tools for mapping database applications to EJB architecture[7][5]. However, they are in the level of mapping schemata to Enterprise JavaBeans instead of migrating in the SQL level. Those tools are more in the area of object-relational mapping[11][10][9].

In database reverse engineering and schema mapping [12][13][14], the common approach is to map relational schema to object schema directly. This paper addresses three additional aspects. First, we map the relational schema to another relational schema. Second, we extract and enrich the schema mapping from the hand-coded schema mapping data provided by IBM recoded in XSL format. Third, we use the schema mapping to translate the SQLs.

6 Conclusion and Future Work

We have presented a practical methodology to aid the migration of applications using traditional database access to those using the EJB programming model, and applied it in a tool developed for IBM WebSphere Commerce Suite. This work has also identified a number of areas which we believe will be fruitful for further investigation in the future.

6.1 Improving the Recommendation Algorithm

The recommendation ranking technique is fairly simplistic in the current release of the tool and we believe that it can be improved upon. We are investigating how the following changes to the Recommendation Engine will improve the quality of the recommendations:

(1) Use the column usage information as the basis for the data coverage analysis.

(2) Analyse the containment relationships that between data beans and the inheritance relationships between the data and access beans that we have recorded to see if these relationships relate to actual usage of the beans and if so, modify the ranking scheme to account for this.

(3) Store user experience from previous migration efforts to aid developers encountering similar tasks in different migration projects (: Item 11)

6.2 Going from Recommendation to Translation

Currently the EJB client code in JSP is not automatically generated. We are using query-rewriting techniques to translate the SQLs to fragments of EJB client code. Furthermore, in a more general situation when EJB architecture is not available yet, the EJB architecture and the finder methods as well as the SQLs inside the finder methods will be generated.

Acknowledgements

In developing the migration process and the Helper Tool we had the valuable help and input from the following people:

(1) Kostas Kontogiannis of the University of Waterloo and John Mylopoulos of the University of Toronto provided valuable input to refine our approach.

(2) Members of E-Commerce Engagement Team in the IBM Canada Laboratory, working to migrate customer IBM WCS systems. Roger Cheung, Laurent Chan helped us understand the processes and challenges involved in their migration process. Mark Crowley was instrumental in getting the tool published on IBM's AlphaWorks website.

(3) Jim Caldwell, Mark Hubbard, George Klima, Sharon Lymer, and Sam Wong of the E-Commerce Development Team in the IBM Canada Laboratory provided valuable comments and suggestions on the functions of the Helper Tool.

Authors

Terence C. Lau is a senior research associate at the Centre for Advanced Studies, IBM Canada, with special focus and interest in business-to-business e-commerce. Dr. Lau received a B.Sc. from University of Hong Kong and Ph.D. in computer science from University of Waterloo.

Dr. *Jianguo Lu* is a research associate of University of Toronto, Department of Computer Science and a visiting researcher at the Centre for Advanced Studies, IBM Canada.

Erik Hedges is a visiting research student at the Centre for Advanced Studies, IBM Canada. He is currently working towards a Masters of Applied Science in Electrical and Computer Engineering at the University of Waterloo.

Emily(Xuemin) Xing is a software analyst in the E-Commerce Development group of the IBM Canada Laboratory. Ms. Xing received a B.Eng. from Dalian University of Technology and M.Sc. in computer science in Memorial University of Newfoundland.

References

[1] Nicholas Kassem and the Enterprise Team, *Designing Enterprise Application with the Java 2 Platform, Enterprise Edition*, Sun Microsystems, http://java.sun.com, October 3, 2000.

[2] IBM, *IBM Net.Data Reference*, Version 7,http://www4.ibm.com/software/data/net.data/, June 2001 Edition.

[3] IBM, *IBM WebSphere Commerce Suite, Programmers Guide*, Version 5.1 Second Edition,http://www4.ibm.com/software/webservers, March 2001.

[4] Sun, *Enterprise JavaBeans 2.0 Specification,* Sun 2001.

[5] IBM, *VisualAge for Java 3.5,* IBM, 2001.

[6] TechMetrix, *Moving from IBM WebSphere 3 to BEA WebLogic Server 5.1,* White Paper, TechMetrix Research, September, 2000.

[7] *In2j: Automated tool for migrating Oracle PL/SQL into Java,* www.in2j.com, April, 2001.

[8] *Migration Guide, iPlanet Application Server,* Version 6.0, www.iplanet.com, May 2000.

[9] Andreas Behm and Andreas Geppert and Klaus R. Dittrich, *On the Migration of Relational Schemas and Data to Object-Oriented Database Systems*, in Proc. 5th International Conference on Re-Technologies for Information Systems, Oesterreichische Computer Gesellschaft, Klagenfurt, Austria, J. Gyorkos and M. Krisper and H. C. Mayr, 13--33, 1997.

[10] S. Bergamaschi and A. Garuti and C. Sartori and A. Venuta, *The object wrapper: an object oriented interface for relational databases*, In Euromicro 1997.

[11] Chandrashekar Ramanathan, *Providing Object-Oriented Access To Existing Relational Databases,* PhD dissertation, Mississippi State University, 1997.

[12] J. Jahnke and W. Schafer and A. Zundorf, *A Design Environment for Migrating Relational to Object Oriented Database Systems,* In Proceedings of the International Conference on Software Maintenance, IEEE Computer Society Press, 163--170, 1996.

[13] Kyle Brown, *Handling N-ary relationships in VisualAge for Java*, www.ibm.com/vadd, August 2000.

[14] M. W. W. Vermeer & P. M. G. Apers, *Reverse engineering of relational database applications,* in Proceedings Fourteenth International Conference on Object-Oriented and EntityRelationship Modeling (ER'95), Gold Coast, Australia, M. P. Papazoglou, ed., SpringerVerlag, New York--Heidelberg--Berlin, December 1995, 89--100, LNCS #1021.

[15] L. Yan, R. J. Miller, L. M. Haas and R. Fagin. Data-Driven Understanding and Refinement of Schema Mappings, SIGMOD, May 2001.

[16] R. J. Miller, L. M. Haas and M. Hernández. Schema Mapping as Query Discovery. Proceedings of the Twenty-Sixth International Conference on Very Large Data Bases (VLDB), Cairo, Egypt, Sept, 2000

[17] Terry Lau, Jianguo Lu, John Mylopoulos, Erik Hedges, Kostas Kontogiannis, Emily Xing, and Mark Crowley, *Net.Data to JSP helper*, IBM alphaworks, http://alphaworks.ibm.com/tech/netdatatojsp
.