Bringing Business Objects into Extract-Transform-Load (ETL) Technology

Huong Morris[*], {Hui Liao, Sriram Padmanabhan, Sriram Srinivasan},
{Phay Lau, Jing Shan, Ryan Wisnesky}[**]

IBM T. J. Watson Research, 19 Skyline Drive, Hawthorne, NY 10532

{*thm, huiliao, srp, sriram*}*@us.ibm.com, ptaclau@gmail.com, jshan@ccs.neu.edu, wisnesky@stanford.edu*

Abstract

Business objects represent the key concepts that a business needs to operate such as people, services, products, etc. but transforming these objects to and from existing data models can be difficult. Business objects have traditionally been represented in a backend data store using relational databases, and techniques for transformation must work with these stores. But such access may violate the encapsulation these business objects require, and so conventional approaches may not provide an adequate solution. In this paper, we examine how to use Extract-Transform-Load (ETL) tools to provide business object transformations. We show how to solve some of these issues by using pluggable components and introduce customized operators for ETL tools. We demonstrate our solution using a commercial ETL system that allows access to several Product Data Management systems and illustrate the use of our technique with several case studies drawn from various industries.

1. Introduction

As business systems have consistently expanded throughout the enterprise, the need for multiple systems with different architectures to inter-operate becomes ever more important. To make this happen, enterprises are turning to higher level software, such as the solutions provided by SAP, Oracle's PeopleSoft, Siebel, and the IBM WebSphere Product Center (WPC), to manage their *business objects* directly. Existing work in this area mainly focuses on how to design business objects in business systems and how to use them in business processes [2, 3, 9, and 10]. Despite advances in information integration techniques [5, 7]; access to heterogeneous data sources remains a challenge. The goal of efficient management of distributed information has become progressively more difficult for several reasons [7]: 1) the data volume is growing due to increased digitization of sources, 2) data is coming from a greater variety of these sources, and 3) virtual marketplaces and global partnerships are requiring integration efforts which stretch across the boundaries of previously siloed systems and individual corporations. In addition, the customer challenges are also growing due to: 1) the complexity of integration of multiple data sources within and between applications; 2) the need to include non-traditional data sources including sensors, multimedia, etc, 3) the need to combine structured and unstructured data, 4) time pressure to deploy new applications, and 5) people and skill shortages to develop new applications. Finally, as discussed in [4], the major trends of Enterprise Information Integration (EII) and Enterprise Application Integration (EAI) are overlapping and that creates one giant integration problem.

Every important entity in a business can be represented as a business object. Business objects are capture the semantics of business concepts and are directly useful for business processes. They represent the key concepts that a business needs to operate such as people, services, and whatever is sold. Business objects are different from simple bits and bytes data embedded inside software; they are used directly by business developers to implement business functions. Master data is used to define key data that uniquely defines business objects. Examples of commercially available product data management (PDM) systems that manage business objects were given above. However, because of their intrinsic complexity, these products do not interoperate with each other. Even within a PDM system, transforming the business objects to and from existing data models, as might be

[*] This work was carried out when the author was at the IBM Almaden Research Center, California.
[**] This work was carried out when the authors were at the IBM Almaden Research Center as Extreme Blue interns.

978-0-7695-3395-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ICEBE.2008.72

required by a merger or new business relationship, can be difficult. Most PDM systems represent business objects, in terms of a master data management (MDM) system, in a backend data store using relational databases. Any techniques for transformation must be able to access these backend data stores. But naïve access to these objects may have unintended consequences, caused by the semantics of the data and the relationships and constraints the data must abide by. As a result, conventional approaches to integration may not provide an adequate solution. Furthermore, one of the key problems that arise as an enterprise attempts to round up its data is that of "semantic reconciliation", that every user and application see a consistent and persistent interpretation of these key business objects.

We describe Callisto, which uses ETL tools to transform or merge business objects efficiently and effectively. Examples of previous ETL work focused on the modeling and managing of the ETL processes can be found in [2]. The paper is organized as follows. In the next section, we explain why the use of ETL tools can be useful in access, aggregate and manage business objects. In section 3 and 4 we describe the Callisto project and its implementation with two realistic use cases.

2. ETL Technology and Business Objects: are they 'apples' and 'oranges'?

Accessing disparate business objects can be complex due to business objects being compound versions of the data embedded inside many databases and unstructured data sources. Current IT technology does not directly support needed functionalities against these objects, such as data transformation, analysis, integration etc. For example, business intelligence (BI) is an important building block in an enterprise nowadays. It helps to make better business decisions.

Integration is usually tackled using one of four main techniques: transformation tools (as in ETL), replication, database gateways, and virtual data federation. Extract-Transform-Load (ETL) tools are pieces of software responsible for the extraction of data from several sources, cleansing the data and customized insertion of the data into a data warehouse as depicted in Figure 1.

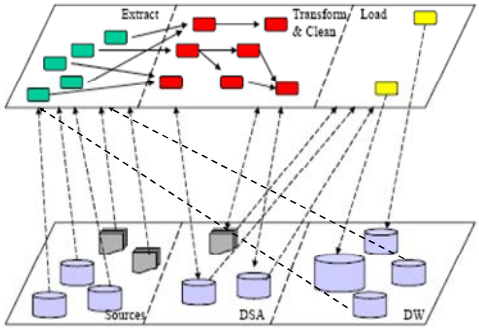

Figure 1: ETL Processes. Derived from [11] with flows added.

At the lower right layer representing data store layer, we have targeted data warehouse after the loading activities have been performed at the upper right hand layer. On the left hand lower layer, data come from various sources (e.g. relational tables and files or from the data warehouse). These data sources are extracted (left hand upper layer) by extraction routines, which provide either complete snapshots or differentials of the data sources. Then these data are propagated to the *Data Staging Area (DSA)* where they are transformed and cleaned before being loaded to the data warehouse.

ETL tools have become a standard technology that aims at easing the pain of data transformation. The user of ETL tools can focus on the semantic mapping from a data source to a data target and let the ETL tool to take care of the underlying transformation details. This idea is a good fit to what is needed in business objects management systems. But current ETL technology only supports the lower level software data (e.g. data inside a DBMS). In short, there is an "impedance mismatch" between business object aware software and conventional systems involved in business processes.

When business objects are hidden inside these applications, it is hard to inter-operate without a deep understanding of the semantics of each of the applications, and this understanding often requires an understanding of the implementation. This makes such business process interaction hard and complicated [6]. Callisto addresses this problem by integrating Business Objects as a component of ETL toolset.

3. Integration Challenges

Current ETL technology supports relational formats, such as relational database tables, CSV files etc. To represent business objects inside of the ETL, we must find a way to describe business objects in a relational format without resorting to examining how the business objects are implemented and stored. In essence, we must create *custom ETL operators* that

expose the required information. This is not an easy task because business objects are usually semi-structured or unstructured. In our project, the business objects in WPC are semi-structured.

The key challenges are: 1) relational presentation of a business object must be as rich as the original object. That is, information about the business object should not be lost when the object is represented in a relational way. In addition, the information presented in the relational view must be presented in a way that is useful. 2) in many business-oriented systems, there is no clear boundary between data and metadata. An ETL system requires operators to expose metadata while a dataflow is designed, and to manipulate the data during runtime. 3) different business objects of the same type may not share properties, so that there is not necessarily a common relational representation for different instances of a type of business object. For instance, a retail *Category* business object may be represented as a table with columns for '*name*' and '*price*', but another *Category* object may require '*UPC*' and '*description'*. However, both are called *Category* objects, and so we cannot always decide on relational representations for an entire class of such objects. 4) business objects and their relational views must relate to each other in a consistent, complete, and useful way. For instance, it is common for one business object to reference another; say, for a *person* object to reference a *department* object, thus capturing the relationship that the *person* is employed by the *department*. Thus, when multiple business objects are represented in multiple relational tables, if one objects references another, that information must be suitably and consistently encoded wherever it is represented in the relational tables.

4. Callisto Overview

In this section, we describe how Callisto integrates a typical commercial business object system to another typical commercial ETL toolset, namely the IBM Websphere Product Center (WPC) [12] and IBM ETL toolset called SQL Warehousing (SQW)[1] toolset. We show how on one side, we have business entities which are better represented with hierarchical and multidimensional objects (which is what WPC basically does); and on the other side, we have a product toolset like SQW that understands relational data and business intelligence.

4.1. Callisto Architecture

Callisto, as depicted in Figure 2, is essentially implemented as a set of plug-ins around an ETL system. Our implementation used Eclipse plug-ins to SQW framework. Callisto extracts and loads information into the WPC using the scripting mechanism and a JSP interface. Callisto also transforms information to and from the hierarchical format that the WPC uses by examining a model of WPC business objects. Finally, Callisto presents and receives relational representations of WPC information from the ETL tools set. The ETL tools set provide support for transforming relational information and connectivity to various relational systems, thus allowing WPC information to be integrated into the SQW framework as form of business object operators. This ETL tools set also provides BI and other operators which can then be used by Callisto.

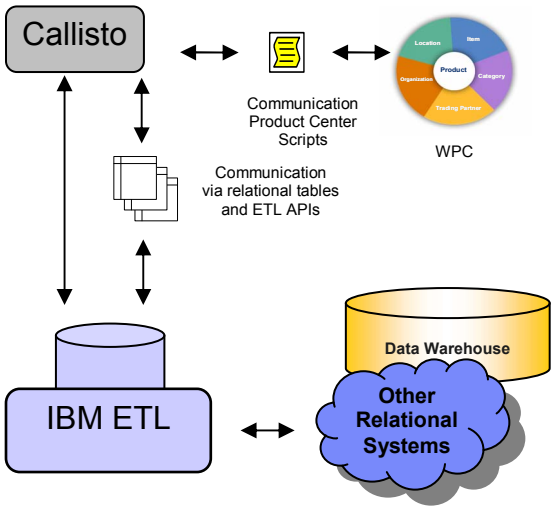

Figure 2: Callisto architecture

4.2. Callisto MDM data model

Callisto MDM data model is based on WPC data model. WPC is a product information repository for an enterprise's master data. This information is maintained in a relational database in the back end, but is represented to the user as business with retail flavor. The "core objects" in the WPC are *catalogs*, *items*, *attributes*, category trees (a.k.a. *hierarchies*) and *categories* (a.k.a. hierarchy nodes). Attributes hold values or group other attributes. Attributes are defined through *specifications* (a.k.a. *specs*).

[1] SQW is one of the toolsets included in the IBM DB2 Data Warehouse Enterprise (DWE) product. Like most commercially available ETL toolset, SQW provides a framework called the *Data Flow*. The data flow is an extensible framework that allows users to build data extraction, transformation and load sequences as a flow of '*Operators'*.

Items make up the primary data element in WPC. They are typically represented as SKU's, individual products, etc. *Catalog* is the containers for items. An item belongs to one an only one catalog. Each catalog has one primary *specification* that defines the attributes that all the items in that catalog share.

Category trees are hierarchical arrangements of categories. This provides users with different "views" into the same set of data (e.g. UNSPSC, UDEX, etc.).

Hierarchies are built and stored separately from items and catalogs. This enables the same hierarchy to be deployed in multiple catalogs, and also allows items in a catalog to be viewed in multiple hierarchies.

Items are mapped to *categories*. Categories defined specific attributes for the items mapped to them through secondary specifications.

4.3. Callisto UML data model

In Callisto, we use IBM Rational Data Architect (RDA) to model the ETL process and Rational Rose [8] to model WPC objects. This UML tool defines data models in a higher abstracted level using a set of well-defined graphical tools. Figure 3 shows the overview of this modelling approach. Each WPC object is modeled as a standard class. And their containment relationships are modeled as aggregation relationships in the Rose model.

After modeling the above two steps, these models are exported as EMF '*ecore*' (Eclipse modeling framework core) models. Note that our UML model of business objects is incomplete and is a simplification; however, it is sufficient and simple for us to use in this prototype.

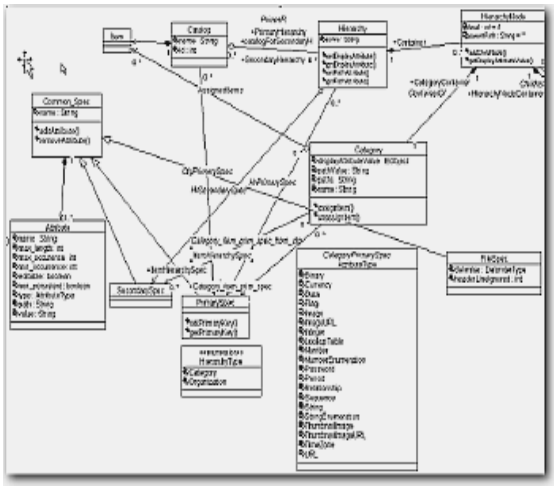

Figure 3: Callisto MDM Model

Also note that because of EMF/serialization constraints, every business object must have a parent container, which is not necessarily how the WPC operates.

4.3.1. EMF code generation. The Eclipse Modeling Framework (EMF) [1] is a Java framework for generating tools and other applications based simple class models. EMF uses these '*ecore'* models and generates customizable Java code that can then be used to manage the life cycle of these business objects, including their relationships as well as provides means of serializing and de-serializing these objects as XML/XMI files. The code generated from the UML and EMF artifacts are what we refer to as the 'WPC model'.

4.4. Callisto implementation

Callisto provides the ability to examine the WPC catalog and to build a metadata model of the catalog information in the Eclipse environment using EMF and conforming to the XML metadata interchange (XMI) standard. Callisto operates by analogy with the existing SQW toolset. Similar to how the SQW design studio allows users to build models of relational tables, Callisto allows users to examine a WPC instance and select relevant *catalog* information. This information is used to populate an EMF model of this WPC instance (i.e. the model provides information that *categories* related; *hierarchies* have these *categories*, etc). This model, which may be serialized to disk (as XML/XMI files) and browsed from within Callisto, is the foundation of the rest of Callisto.

We developed 4 operators, which represent the business objects for *Import* and *Export* functions in WPC:

1. **Item Export**: Export a category of items from the WPC. The items contain the values of their attributes as columns in a table.
2. **Item Import**: Import a category of items into the WPC, with attribute information.
3. **Hierarchy Export**: Export a hierarchy from the WPC, where parent-child relationships are maintained using paths and parent/child columns.
4. **Hierarchy Import**: Import a hierarchy into the WPC, while maintaining parent/child relationships.

With a model in hand, user can drag and drop the Callisto operators into a Data flow. Depending upon the operator chosen, the user would then select different aspects of the WPC instance in order to build the operator's properties. For instance, when using *Item Export,* a user would browse the WPC model and select a particular category of items to export.

Callisto provides a code generator for each operator, which generates script that performs the required WPC operation. For example, an *Item Export* operator causes the generation of a WPC script that involves exporting the description of the item being exported. Finally, Callisto provides runtime component that plugins into the SQW engine, so that data flows build using Callisto operators can be executed. This runtime executes the generated scripts to communicate with a WPC instance, sending or receiving information as required.

5. Scenarios

Use case 1: Our first scenario focuses on master data integration: a typical customer pain-point for most MDM systems. WPC catalog building is a semi-automatic process that can require a substantial amount of skill and manpower to deploy. In this scenario, Callisto's aim to see whether the process integration of new data from and into WPC master catalog can be simplified.

Situation: In a fictitious example, AceMart, a large retail chain plans to expand its product portfolio by acquiring BetaMart. The acquisition needs to be completed by integrating AceMart's product catalog with BetaMart's various data sources.

System Environment: AceMart uses the IBM WPC to centrally manage its product catalog information. BetaMart's product, suppliers, stores, and pricing information are scattered throughout different systems and suppliers' databases.

Limitation: AceMart's upper management has required that the integration of BetaMart's product information into AceMart be completed in three months. However AceMart's systems group estimates that this could take much longer. Before Callisto, the procedure to integrate BetaMart's catalog into AceMart's would be to transform all BetaMart's catalog into AceMart's WPC; load data into WPC using WPC import script and this can be error prone and time consuming.

Solution using Callisto:

1. Define Data Source by selecting WPC *Item Source* Operator. This will open up a WPC instance with BetaMart's catalog item and category that user can select. In our demonstration as shown in Figure 4 below, we select the "*Drill"* category in BetaMart's Makita catalog.
2. Define Target Source by selecting WPC *Item Target* Operator. This will open up a WPC instance with AceMart's catalog. We select the AceMart's Drill catalog for Item Target.
3. Complete Dataflow in SQW design studio

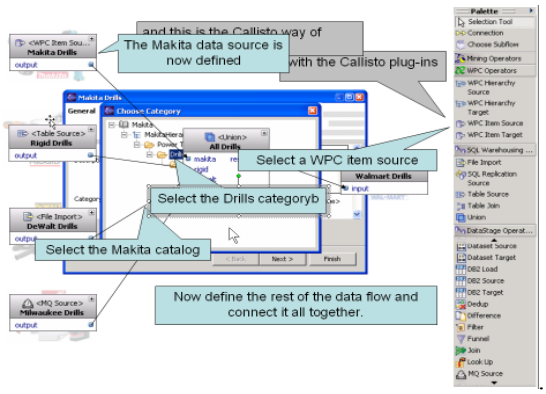

Figure 4: Use case 1: MDM integration

4. Load Data into AceMart Master Catalog by clicking on "Run" in the SQW design studio. Internally, Callisto will move all *Drills* from BetaMart's catalog to AceMart's catalog. Thus complete the migration of *Drills* item catalog from BetaMart to AceMart.

Use case 2: Our second scenario shows how Callisto can enable WPC master data to be integrated into a Data Warehouse to be analyzed by Business Intelligence (BI) tools. Traditionally, BI tools were limited to analyzing transactional data. Callisto adds a new dimension, master data, into the Data Warehouse that provides a whole new capability for BI.

Situation: A fictitious online bookseller, Books4Sale.com, found that the sales of its Harry Potter books grew two-fold when it changed the category from "Children" to "Fantasy". Changing categories can dramatically boost sales. Books4Sale.com would like to analyze the past trends of its sales to determine the optimal category for its products.

System Environment: The bookseller uses the IBM WPC to centrally manage its products and categories. WPC manages the Books4Sale catalog but it does not have the ability of a BI tool such as reporting and analysis for evaluating product trends.

Limitation: Conventional data export techniques are too slow to react to the high volume of daily catalog changes. Failing to spot a bad branding or categorization can lead to a huge loss. In addition, peak sales trends may be missed by slow, conventional techniques

Solution using Callisto:

1. Define WPC *Item Sources* operator. Again, as in Use case 1, this will open up an instance of WPC's Books4Sale catalog. From this catalog, the user selects "*Fantasy*" book category and can see that the category is defined by 2 attributes: *id* and

name. When the data flow is run, this operator will connect the WPC instance and present this selected item source, namely *Fantasy*, in the WPC category as a relational table to the next SQW operators.

2. Add the *Current Time* operator. This operator is provided by the SQW, will add time stamps information to the information from WPC.
3. Define BI *Item Target* operator. Again, this is another SQW existing operator. It is a business intelligence operator and it takes the input data from the previous operators, namely the *WPC Item Source* and the *Current Time* operators; merges it with existing data in the data warehouse.
4. Complete Dataflow. Once the execution of this flow is complete, the new information will be available for analysis in next step.
5. Analyze Results using any reporting and analysis too. In our experiment, we use IBM Alphablox and data mining tools.

Figure 5 shows the SQW design studio screen shot once steps 1 to 4 completed.

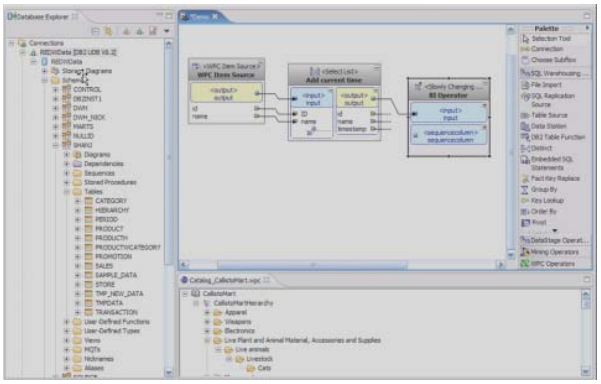

Figure 5: Use case 2: Bringing BI into MDM – Screen shot of SQW Design Studio for steps 1 to 4.

9. Conclusion

We have demonstrated that by adding some conceptually simple Java-based operators to a transformation tool, business objects can be integrated, assembled or disassembled. Our approach has been relatively simple and makes use of commonly available technology: we use UML and EMF modeling, which capture the key constraints between objects, to generate Java code. The java code is used to present relational representations of selected business object instances based on the object's state. Finally, custom operators use these Java objects to present clean relational table schemas (virtual tables) to the rest of the SQW transformation framework.

While this is a prototype, and we are short of a complete system, our work provides some encouragement that existing business intelligence tools can be mediated in their use of data residing in relational database systems. Further experience and development of more and richer operators for different master data management systems such as SAP, Siebel and PeopleSoft would validate the approach.

6. Acknowledgements

The authors thank the IBM Software Group, especially the ETL and the Websphere Product Center development groups for providing resources to this project. The authors also thank the anonymous reviewers for a thorough and constructive set of reviews of this paper.

7. References

[1] Eclipse Modeling Framework (EMF), http://www.Eclipse.org/emf/.
[2] P. Eeles and O. Sims, "Building Business Objects", *Wiley Computer Publishing,* 1998.
[3] G. Gillibrand, "Essential business object design", *Communications of the ACM, 43, 2*, 2000.
[4] A. Halevy, N. Ashish, D. Bitton, M. Carey, D. Draper, J.Pollock, A. Rosenthal, and V. Sikka, "Enterprise Information Integration: Successes, Challenges and Controversies", *ACM SIGMOG 2005: 778-787.*
[5] J. Madnavan, and A. Halevy, "Composing Mappings Among Data Sources", *VLDB 2003:572-58.*
[6] A. Maier, B. Mitschang, F. Leymann, and D. Wolfson, "On combining business process integration and ETL technologies, *BTW 2005.*
[7] H. Morris, S. Lee, E Shan, and S. Zeng, "An Information Integration Framework for Product Lifecycle Management of Diverse Data", *ACM JCISE 2004, Vol 4, No 4.*
[8] IBM Rational Rose, IBM Rational Data Architect, http://www.ibm.com/software/rational
[9] O. Sims, "Business Objects, Delivering Cooperative Objects for Client-Server", ***McGraw-Hill*** *Book Co.,* 1994.
[10] J. Sutherland, Business Objects in corporate information systems", *ACM Computing Survey*, 27, 1995.
[11] P. Vassiliadis, A. Simitsis, and S. Skiadopulos, "Conceptual modeling for ETL processes", *DOLAP,* 2002.