Paper 143-26

# Integrating SAS® with an Open World: Java, JSP, LDAP, and Oracle

Jay L. Stevens, Whitehurst Associates, Inc., Atlanta, GA
Brian Santucci, THINKologies, Inc., Atlanta, GA

## ABSTRACT

The world of business technology is no longer the sole domain of proprietary systems and protocols; it is now a world of open standards and collaboration. While it has its roots in the "big Iron" of the mainframe world of yesterday, SAS has more than kept up with the incredibly rapid rate of technological change accompanying the Internet revolution. By providing industry-standard interfaces between its software and open technologies such as Java and LDAP, SAS have given businesses new ways to deliver information. THINKologies develops cutting-edge technology-based solutions to identify, measure and predict the dynamics of business relationships, helping companies monitor and enhance their relationships with customers, employees, and business partners. THINKologies required world-class data warehousing, reporting and analysis tools that integrated with the latest open technologies available. This paper will review the architecture and some of the technical details behind THINKologies' SAS-based online reporting system co-developed and designed with Whitehurst Associates. The system tightly integrates SAS with Java, Java Server Pages, Weblogic™, LDAP, and Oracle(tm) through the use of Base SAS®, SAS/Graph®, the IOM object spawner, Integration Technologies, SAS/Warehouse Administrator® and SAS/Access to Oracle®.

## INTRODUCTION

SAS' capabilities in dealing with large volumes of data in data warehousing environments in all industries is well documented. As many SAS professionals know, there is the misperception in the industry (largely propagated by would-be SAS competitors) that SAS fits nicely into a box.

"SAS is a mainframe package."
"SAS is a statistical package."
"SAS is a mainframe statistical package."
"We don't compete with SAS, they're not big into data warehousing."
"SAS is on its way out. No one will be using it in 5 years."

These are all actual quotes from actual potential SAS customers (or vendors who were trying to sell them non-SAS solutions). With the explosion of the internet, SAS has not remained motionless, a monument to the days when IBM dinosaurs ruled the earth. Instead, SAS has kept pace (mostly) with the new techologies sweeping the IT landscape. Words like Java, CORBA, LDAP, JSP, and others like it that had no meaning just a few years ago, are now part of the business technology vernacular. This paper will review, via an implementation case study, how well SAS speaks the Open language of today's technology.

## CASE / PROJECT BACKGROUND

Early in the year 2000, THINKologies.com began a rigorous vendor search process to find both software and consulting partners to provide tools for and to assist its internal staff with the development of the Online Reporting Module for version 2.0 of their flagship product, the ePortcard[sm] online application. Version 1.0 was developed under NT using an ASP framework accessing SQL Server database back end. The ePortcard[sm] 2.0 architecture called for a Unix-based JSP (Java Server Page) implementation of the presentation layer (replacing the NT-based ASP architecture).

## PROJECT REQUIREMENTS

Brian Santucci, Business Intelligence Systems Manager at THINKologies and co-author of this paper, presided over the vendor selection process. The project had the following objectives:

- Build an Online Reporting Datamart - Create a repository with data structures appropriate for decision support applications.
- Manage the Warehouse - Develop and manage extraction, transformation, and loading (ETL) processes to populate the data warehouse.
- Offline Reports - Purchase analytic tools that will allow for development of sophisticated statistical models of ePortcard data.
- Online Reports - Purchase of online analytic tools that give clients controlled interactivity with collected ePortcard data.
    - Parameter-driven queries
    - OLAP (click-able graphs and data that allow drill-down)
    - Static HTML delivery of models developed offline
- Integration with content and application development to facilitate proper procedural movement of data and code to the production environment.

### SOFTWARE / CONSULTING PARTNER SELECTION: THINKOLOGIES' PERSPECTIVE

As a startup company, we at THINKologies sought two elements to meet the above outlined objectives. We needed to find a vendor, or vendors, that offered software that would help us meet the objectives and we needed help implementing the software to meet those objectives.

When we approached the marketplace, we found that vendors typically fit into one of two categories: Warehouse Management tools and Reporting and Analysis tools. Our primary focus was on selecting the right reporting and analysis tool for our needs and our secondary focus was to find a warehouse management tool that would work well with our reporting and analysis software.

We evaluated a number of vendors for reporting and analysis including Business Objects®, Cognos®, Hyperion®, Hummingbird® and SAS. Since our site was converting to a JSP architecture we quickly narrowed the field to Business Objects and SAS since the others did not offer tight integration with JSP.

With respect to warehouse management tools, SAS offered their complementary product Warehouse Administrator while Business Objects offered tight integration with Informatica's PowerCenter™ / PowerMart™. We invited parties involved with both of these solutions to install evaluation software on our servers to assist us with our decision.

With the software in hand, we quickly discovered what we needed to know to make our decision. SAS offered a flexible architecture that opened up the power of SAS to a JSP site through the Integration Technologies Java API. Business Objects was in the process of converting an existing product, WebIntelligence, to a Java API. It was unclear what we would be able to do with the Business Objects

API other than replicating the Web Intelligence product. With the SAS Integration Technologies Java API, the opportunity to develop customized solutions that met our needs was extensive.

Having made the decision on reporting and analysis software, the warehouse management tools selection became a simple matter. SAS offered similar functionality at a much lower price point than Informatica with a clear integration advantages with other SAS modules. Also, Informatica's data manipulation tools were primarily SQL-based. SAS offered that and more, leading us to believe that our ability to manipulate data in complex ways would be better facilitated using SAS software.

Once the decision about what software to select was made, we decided that we needed assistance putting it all together. We evaluated the alternatives and settled on a contract with Whitehurst Associates. THINKologies selcection of a consulting partner was based on the excellent reputation that Whitehurst has established working with "dot-coms" in the Atlanta area and the local proximity of their leading technology experts. In addition, they had been involved with the evaluation process from the beginning, assisting with SAS software installation and configuration, etc. and were familiar with the challenges we were facing.

## THE SOLUTION

Based on the requirements, Whitehurst recommended the following products for the THINKologies project:

- **Base SAS** – the core of the SAS System.
- **SAS/Graph** – for generation of graphics.
- **SAS/Access for Oracle** – to access the production databases.
- **SAS/Stat** – for analysis.
- **SAS/Warehouse Administrator** – to manage the warehouse ETL processes.
- **SAS Integration Technologies** – to provide integration with Java.
- **SAS Enterprise Guide** – to provide point-and-click desktop analytical reporting access.

## SAS INTEGRATION TECHNOLOGIES

"Integration Technologies" is the product name that SAS® has given to a group of open technologies including: Component Platforms (support for programmable object interfaces to SAS® services), Application Messaging Platforms (support for MSMQ, MQSeries, etc.), Enterprise Directory Platforms (support for directory services such as LDAP and Active Directory Server). In building the ePortcard Online Reporting application, Whitehurst and THINKologies used 2 of these platform areas: Component Platforms (via IOM) and Enterprise Directory Platforms (via LDAP).

### INTEGRATED OBJECT MODEL (IOM)

"SAS Integration Technologies delivers a base service component hierarchy with an integrated object model (IOM) that provides access to the SAS software procedural scripting language, data, file system, results content, and formatting services." [1]

-SAS Integration Technologies Whitepaper

SAS® Integration Technologies provides a standardardized common middleware platform and object interface to the core modules of the SAS system. This enables diverse clients (from custom-built C++ or Visual Basic applications to Java Servlets/Applets to SAS® shrink-wrapped products like Enterprise Guide) to access SAS services and objects via a common object interface. **Figure 1** at the end of this paper provides and overview of the IOM object hierarchy.

This integrated object model (IOM) is surfaced via the IOM server, a separate process similar to SAS/CONNECT's ® spawner or TCP daemon which facilitates and manages the startup of client to server TCP/IP conversations. The root object in IOM is the SAS® Workspace. This is roughly equivalent to a SAS® session. As shown in the chart below, once a root Workspace object has been created other classes are available under the workspace including:

- DataService, which offers Libname assignments and access via JDBC.
- FileService, which allows Filename assignments.
- Utilities, which allows access to HostSystem functions, SAS System options, access to SAS formats in the Java context, and access to the ResultPackage service (including HTML results).
- Language Service, which allows programmers to submit SAS language statements directly to the workspace or to run SAS StoredProcesses (specially configured SAS programs stored in a file) to allow parameters to be passed to the program via Java.

The same IOM server can handle requests from both COM (Windows Component) and CORBA (Java) clients. This allows the data to take on many different "presentation personalities" while keeping the data and SAS processing components centralized. So, for example, let us suppose we have a SAS Stored Process (a SAS program) that analyzes some data and generates summary statistics. Further, this program needs to be run every time a new request is made because of the nature of the report. The IOM server enables multiple and diverse clients to connect to SAS, utilize SAS services and retrieve the resulting data. So a Microsoft Excel ® spreadsheet macro could be written that would invoke the StoredProcess and retrieve the results to a spreadsheet. The same process could be invoked by a Java thin client application, or a Java servlet, or a Visual Basic application, or an ASP page. With the advent of the IOM server, SAS services and SAS data are now open.

### LDAP - LIGHWEIGHT DIRECTORY APPLICATION PROTOCOL

LDAP is a protocol designed (as the name implies) to be lightweight and speedy. Ostensibly its primary mission is to provide cross-platform directory services. For example, a company might store its entire user database (along with each user's access levels, name, address information, etc.). Using a standard API, webservers, fileservers, and applications could all use the centralized LDAP store to authenticate and validate users. LDAP, however, can be used to store any kind of hierarchically organized information. The open interface allows many diverse clients to utilize the services.

## THE EPORTCARD APPLICATION

THINKologies ePortcard application allows businesses to collect feedback from business process stakeholders (i.e. customers, suppliers, partners, employees) via multiple input sources (Web, Kiosks, WML, PDA wireless, etc.). These data are loaded to Oracle directly. ePortcard[sm] customers can then use online reporting tools to get information and analysis on the incoming feedback data.

### ENTERPRISE JAVA BEANS ARCHITECTURE

There are 3 user bases that interact with the ePortcard application: administrators, feedback providers, and information consumers. The architecture for handling administration and feedback provision utilize a transaction system built using J2EE's (Java 2 Enterprise Edition) EJBs (Enterprise Java Beans). Administrators define elements of the system such as channels (e-mail, URL, kiosk, IVR), forms (a.k.a. surveys), and launches(specification for form distribution and data collection) which define the experience of the feedback provider. The EJB tier handles the creation of these entities and persistence of the associated data in an Oracle database. It also handles the form distribution and data collection.

The third type of user, the information consumer, interacts with a portion of the application that is built around SAS Institute's Integration Technologies module. This type of user's first interaction with the system is likely to involve accessing a report. Access to management reports is handled via LDAP, specifically the freely available OpenLDAP implementation. A list of all users and all available reports is stored in the LDAP database.

### LDAP INTEGRATION WITH SAS

This LDAP listing of users is constantly updated via SAS by using the new CALL routines ldaps_open, ldaps_search, and ldaps_entry. In addition to updating and loading new data, there are ETL programs that load all new ePortcard users to LDAP that have been recently added to the Oracle user tables. This synchonization between Oracle and LDAP was done completely with SAS.

When a user accesses the reports webpage, the JSP that handles the request uses a JNDI-based JavaBean (developed by Whitehurst and THINKologies) to search the LDAP database for group membership for that individual and creates a group list. The bean then searches for reports that are available to all groups in the group list and returns the results to the requesting JSP. The reports available to the user range from static html or pdf documents created by THINKologies analysts using SAS Enterprise Guide to real-time dynamic reports that utilize SAS and a custom JavaBean architecture to return real-time, dynamic results.

## CUSTOM BEAN INFRASTRUCTURE

The heart of ePortcard's online reporting component is based upon the Java API offered within SAS/Integration Technologies. With the introduction of SAS/Integration Technologies, web application developers were given the ability to integrate the power of SAS with Java in an open way. To harness this power and flexibility, Whitehurst Associates created a JavaBean infrastructure, later extended and enhanced by THINKologies, that is customized to the needs of the ePortcard application. These JavaBean components act as custom wrappers to the base SAS® Java classes, extending and customizing their functionality.

### RPTWORKSPACE (SAS WORKSPACE MANAGEMENT)

The workspace bean manages the retrieval and closure of SAS workspaces over the IOM bridge for Java. The SAS workspace class allows the programmer to specify the connection properties in multiple ways: either via a Java Properties object and the WorkspaceFactory or by reference to an LDAP server where the connection information is stored. For connections obtained via an LDAP server, a feature called workspace pooling is available. Workspace Pooling provides for sharing of Workspace objects among multiple clients. So rather than incurring the overhead of starting and stopping a Workspace session for every request, Workspace Pooling allows a Java application to connect to and disconnect from an existing Workspace object without incurring the overhead of startup and shutdown between those requests. The rptWorkspace bean's methods are independent of the workspace access method, meaning that they work for both the workspace factory and pooling methods.

### RPTSESSION (JSP SESSION MANAGEMENT)

With a workspace available to perform SAS services, the ePortcard application begins the task of gathering information from the user about what tasks SAS should perform. The user enters information related to a report request into standard HTML forms. This information is captured and passed to the session bean. The session bean stores this information across page clicks until the point where enough information has been specified to make a request to the SAS server.

### RPTQUERY (JDBC CONNECTIVITY)

The query bean is a wrapper to Integration Technologies' DataService classes as well as standard Java 2 JDBC classes and methods for accessing datasets. The query bean is used by the process bean (below ) or can be called independently with a SQL string as an argument. If a stored process is not necessary for the job, it is possible to write or generate regular SQL and pass that to the query bean. This flexibility makes it very easy to get information from the SAS server.

### RPTPROCESS (EXECUTE SAS JOBS)

The process bean is the workhorse of the ePortcard application. By encapsulating the functionality of several SAS and standard Java classes, it shelters the JSP developer from the complexities of both Java and SAS. The process bean is designed with methods to allow the JSP page to pass arguments to a SAS stored process (SAS program), execute the SAS program, retrieve the results (both images and data), and finally to use the results in the JSP page.

Usage of these beans in a JSP page is shown in the example below:

**Passing Arguments to and Executing the Stored Process**

```
1  <%
2  try {
3  rptWorkspace.open();
4  string attribute=rptSession.getValue("attr")
5  rptProcess.addParam("ID",ID);
6  rptProcess.addParam("attribute", attribute);
7  rptProcess.execute("testprocess");
8  }
9  %>
```

In line 3, the call to the open() method of rptWorkspace establishes the SAS workspace (session) with the server. Line 4 contains a variable assignment. We are retrieving a value from the JSP session (set earlier) that will be used as a parameter for the stored process. In lines 5-6, the addParam() method of rptProcess is called to set the parameters that will be passed to the SAS program. In line 7, the execute() method of the process bean is called. This actually runs the SAS program using the supplied parameters.

**The SAS Stored Process**

As mentioned previously, a SAS Stored Process is nothing more than a SAS program with one special addition.

```
testprocess.sas

1  %let ID=;
2  %let attribute=;
3  %let USERID=;
4
5  *ProcessBody;
6
7  proc sql;
8  create table ds1_&userid
9  as
10 select *
11 from table
12 where id=&id
13 and attribute=&attribute;
14 quit;
```

All of the parameters that can be passed to the StoredProcess are specified at the top of the program (lines 1-3). The special comment "*ProcessBody;" indicates that the SAS program proper is about to begin. In the ePortcard Application, any datasets or images created by Stored Processes follow a naming convention. This is to allow

the Java beans to retrieve them via JDBC. So if another dataset had been created in the program above it would have been named sequentially ds2_&userid. The &userid (the web user's unique login id) is passed behind the scenes by the rptProcess bean when it submits the other parameters (ID and attribute). This interface allows any number of datasets or images to be created by the stored process for later access by Java.

**Retrieving the Results**

Once the stored process has executed, the results are then available to be used in a JSP page as shown below:

```
1  <%
2  rptProcess.doQuery(1);
3  while(rptProcess.next()) {
4  string ID=rptProcess.getValue("id");
5  string attr=rptProcess.getVAlue("attribute");
6  %>
7  <tr><td><%=ID%></td><td><%=attr%></td></tr>
8  <%
9  }
10 %>
```

The doQuery(1) method of the rptProcess bean on line 2 above retrieves the dataset from SAS via JDBC and makes it available to the bean. The argument (1) tells the process bean to retrieve the dataset named ds1_&userid. In lines 3-9 a loop is set up using the next() method of the rptProcess bean which returns true as long as there are rows from the result data to process. The getValue() method is used to retrieve values from the resultset via column name.

In addition, the process bean also maintains a HashMap cache which tracks result sets and images that have already been created by a particular user. If the user requests another view of the same data, the process bean is able to recognize this and retrieves the stored process results that are cached in memory rather than re-running the program. This output caching mechanism substantially increases performance over repeated calls to the SAS server.

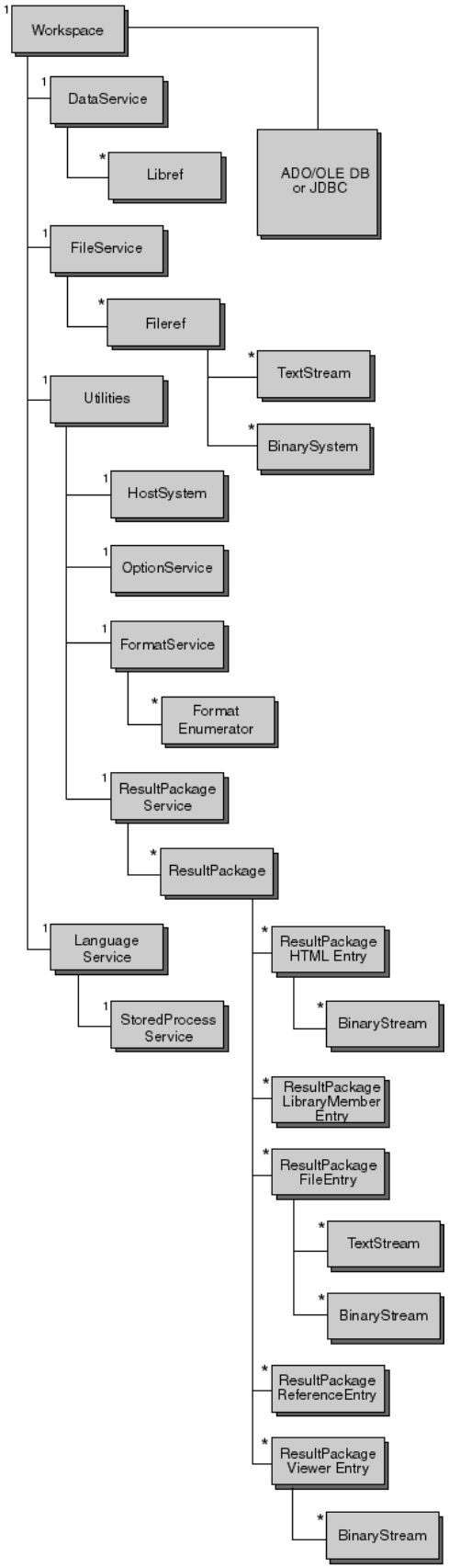
## ETL PROCESSES

SAS plays a pivotal role in a part of the application that the user never sees: ETL (extraction, transformation, and loading). The Oracle transaction database that supports the EJB layer is highly normalized to optimize transaction performance. Recognizing that this structure would be sub-optimal for reporting and that long-running aggregation queries would be unwelcome on the transaction database, THINKologies sought a tool to assist them in the development of an ETL process that would create a reporting data mart. SAS Warehouse Administrator was chosen as the tool for the task.

Warehouse Administrator is responsible for managing the code base that extracts data from the Oracle Transaction database using SAS/Access to Oracle libname assignments, performs transformations using SAS functions and programs, and then reloads the information into a separate Oracle database using a schema that is optimized for reporting.

To avoid contention between database updates and user requests, two copies of the reporting database are maintained. The application and the ETL both dynamically look up which database on which to work.

**FIGURE 1: SAS INTEGRATED OBJECT MODEL**



## CONCLUSION

With the advent of SAS Integration Technologies, SAS has given application developers (web-based and otherwise) a powerful new tool for integrating SAS with custom-built or existing applications. For the first time, developers now have the option and ability to use industry-standard programming languages and protocols in seamless combination with the world class analytic and data management strengths of the SAS system.

## REFERENCES

[1] SAS® Integration Technologies Overview - http://www.sas.com/rnd/itech/papers/oviewSUGI24.html

## ACKNOWLEDGMENTS

The authors would like to thank the following individuals who greatly assisted in our understanding and application of the technologies discussed in this paper:

Joel Gardi
Don Chapman
Steve Harris
Biff Beers

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Jay L. Stevens
Whitehurst Associates, Inc.
341 Eureka Drive
Atlanta, GA 30305
678-358-5734
jay@whitehurst-associates.com
http://www.whitehurst-associates.com

Brian Santucci
THINKologies, Inc.
1111 Peachtree St.
Atlanta, GA, 00000
404-111-1111
brian@thinkologies.com
http://www.thinkologies.com