ERP Application Development Frameworks: Case Study and Evaluation

Nattanicha Rittammanart*, Wisut Wongyued†, and Matthew N. Dailey*
*Computer Science and Information Management Asian Institute of Technology, Pathumthani, Thailand
†GismoGears Co., Ltd., Chiangmai, Thailand
Email: Nattanicha.Rittammanart@ait.ac.th, wisut.wongyued@gmail.com, mdailey@ait.ac.th

***Abstract*—Enterprise Resource Planning (ERP) software brings many benefits to an organization, but commercial software is expensive. Smaller organizations may prefer free and open source solutions but are faced with an enormous array of options from different vendors. One of the most difficult decisions to make is whether to develop software from scratch or to adopt a free and open source ERP framework. To answer this question, we evaluate one of the most popular enterprise software development frameworks, JBoss Seam, and one of the most popular open source frameworks for ERP development, Apache Open for Business. Using a simple real-world application integration problem involving an asset management system and an accounting system as a case study, we compare the frameworks along a number of dimensions. We find that the ERP framework, OFBiz, is the better choice for simple ERP development problems typically encountered in smaller enterprises.**

***Index Terms*—Enterprise applications, Enterprise resource planning, Application frameworks, Enterprise application integration, OFBiz, JBoss Seam.**

I. Introduction

Modern organizations require many information systems to support their business processes. Enterprise Resource Planning (ERP) software attempts provide a standard, unified, and consistent interface to common functionality. Commercial ERP software brings many benefits to an organization, but it is extremely expensive. Due to cost factors, smaller enterprises wanting to streamline their business processes may be faced with two options: develop their own software to integrate existing standalone systems, or adopt a free and open source business automation solution such as OFBiz, OpenBravo, or Tiny ERP.

Regardless of whether they develop new software or adopt an existing solution, organizations need a platform that makes it easy to integrate disparate information systems and rapidly customize the application logic. Which approach, new software, or open source automation framework, is best? Currently, there is relatively little information available to guide organizations in making a decision. For this reason, in this paper, we compare one of the most popular enterprise software development frameworks (JBoss Seam) with one of the most popular open source frameworks for ERP development (Apache Open for Business).

In general, an application framework is a set of abstract classes forming a reusable design [1], a set of state-of-the-art development tools and practices defined by an independent standard body [2], or a reusable skeleton for an application that targets a specific domain [3]. A good framework provides reusable solutions for difficult, recurring problems.

Among the most difficult recurring problems faced by ERP developers is the need to integrate multiple systems in order to automate a business process. Although every application development platform has platform-specific means to accomplish this goal (e.g. JMS and Java RMI for the Java platform), XML-based Web services are rapidly emerging as an effective and platform-neutral technology for application integration.

In this paper, to limit the scope of the evaluation, we focus on relatively straightforward integration problems in which a simple point-to-point synchronous request-response (remote procedure call) model is appropriate. Under these circumstances, SOAP is an ideal Web service protocol. The service provider exports a WSDL specification documenting the required format of the XML request and response messages necessary for invoking services. The client uses the WSDL specification to construct a request message, sends the message, and (in the simplest case) blocks until the response is received.

Given this common integration problem, one of the main specific criteria we use to compare frameworks is the ease with which developers can deploy and interact with SOAP Web services using the framework. In addition to this specific criterion, we compare frameworks on a number of more general criteria, such as customizability and modifiability, to be introduced in Section II.

The general enterprise software development framework we chose is JBoss Seam, and the open source ERP framework is Apache Open for Business (OFBiz). Both frameworks are based on Java EE, both provide numerous features useful for ERP software development, and both ship with an integrated Web server. To motivate the comparison, we use as a case study a simple integration problem that arose in the context of developing an asset management system for Haadthip Public Company Limited, a beverage distributor located in southern Thailand.

The rest of this paper describes the frameworks and the case study scenario in detail then evaluates the two frameworks according to their suitability for ERP system development. We find that for simple integration problems such as those needed in our case study, OFBiz is a better solution than JBoss Seam according to nearly every evaluation criterion.

II. Experimental Design

This section describes the frameworks we selected for comparison, the case study implementation, and our criteria for evaluating the frameworks.

A. JBoss Seam Framework

JBoss Seam [4] is a lightweight enterprise application development framework. It builds upon Java EE, with the aim of unifying Java Server Faces (JSF) with Enterprise JavaBeans (EJB) 3.0 to create a single consistent programming model.

Seam applications are organized according to a standard three-layer enterprise architecture (presentation, business logic, and persistence). The presentation layer is based on the Model-View-Controller architectural pattern; developers use JSF to to implement views and controllers. In addition to JSF, Seam supports its own tag set, Ajax integration, localization, and themes in the presentation layer. The business logic layer of a Seam application is based on EJB3 session beans. The container also provides security services as well as business process automation via the Drools business rule engine and the jBPM workflow engine. Business logic layer services can easily be exported as SOAP Web services. The persistence layer uses the Java Persistence API (JPA) object-relational mapping technology.

B. OFBiz Framework

Apache Open For Business (OFBiz) [5] is an open source enterprise automation system providing, among many other features, ERP functionality. The system attempts to provide reusable modules for common business functions and a variety of paths to develop custom business logic and connect with external systems.

Similar to Seam applications, OFBiz has a three-layer architecture. The presentation layer is based on the Model-View-Controller pattern and makes extensive use of the Decorator pattern [6] for reuse of design elements. OFBiz integrates with many presentation-tier technologies, including Tomcat or Jetty as the Web server, the Freemaker template engine, the JasperReports report engine, the JavaPOS point-of-sale device API, the XML User Interface (XUI) rich client platform, and the BeanShell Java scripting language. The business logic layer utilizes the Service Oriented Architecture pattern in which application developers organize the business logic as a set of reusable services. Services are implemented as scripts using an OFBiz-specific XML scripting language, Java via BeanShell, or other languages such as Python via the Bean Scripting Framework (BSF). Services can be easily exported as SOAP Web service or Java RMI endpoints. The persistence layer provides a database-independent entity persistence engine to the other layers using an XML specification of the database mapping and an API following the Table Data Gateway [7] pattern.

C. Implementation

As a case study to drive the comparison of the two frameworks, we selected a simple integration problem we encountered while developing an Asset Management System (AMS) for Haadthip Public Company Limited, a beverage distributor based in southern Thailand.

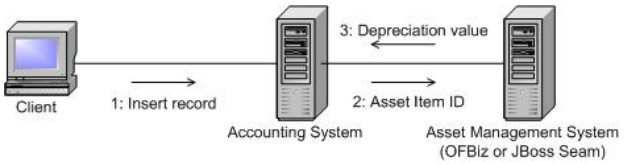

Fig. 1. Communication diagram for case study implementation.

One of the requirements for the AMS is that it needs to supply information about assets to a separate legacy accounting system connected to a Foxpro database. We selected a simple use case for this study in which a user of the accounting system client inserts an accounting record related to a particular fixed asset into the accounting system database. In this situation, the accounting system needs to retrieve asset-specific depreciation information from the AMS before inserting the new record into the accounting system database.

Our design is shown in Fig. 1. First, the client inserts an accounting record including an accounting ID, an asset item ID, and related fields. Before the accounting system inserts the data, it passes the asset item ID to the AMS to get a depreciation value. The AMS looks up the asset item, applies the necessary business logic to calculate the depreciation value, and returns it to the accounting system. The accounting system inserts the new record including the depreciation value, commits the transaction, and notifies the client of success or failure. This design maintains loose coupling between the accounting system and the AMS and keeps the business logic related to asset management local to the AMS. Since the use case is interactive, the communication is best implemented using a synchronous request-response protocol such as SOAP.

For the purposes of the current experimental comparison, we implemented two versions of the AMS, one on Seam and the other on OFBiz. Both systems use a PostgreSQL database for persistence, share the same database schema, implement the same business logic, and export their functionality to the accounting system via a SOAP Web service endpoint.

Fig. 2 shows the basic technologies used at each architectural layer in our Seam and OFBiz AMS prototypes. In the presentation layer, the Seam AMS relies on JSF, whereas OFBiz utilizes a widget-based structure and the Freemaker template engine. At the domain logic layer, the Seam AMS uses EJB3 session beans, whereas OFBiz uses simple XML scripting language. Finally, at the data source layer, the Seam AMS uses the JPA object relational mapping whereas the OFBiz AMS prototype uses the OFBiz Entity Engine, which implements a Table Data Gateway [7].

D. Comparison Criteria

Frameworks can be compared using many criteria. We use the following criteria specifically for comparing ERP application development platforms in our case study:

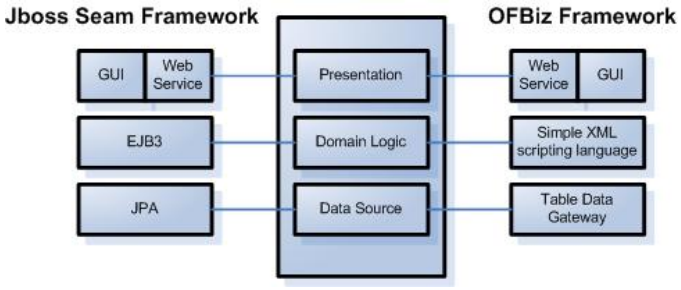

Fig. 2. Architectural layers of the Seam and OFBiz AMS prototypes.

- *Technology complexity*: are the technologies used to create applications on the framework easy to understand?
- *Ease of service exposure*: what are the steps needed to expose a Web service on the framework.
- *Ease of administration*: is installation and application deployment simple?
- *Resource utilization*: are the memory and disk space requirements reasonable?
- *Ease of presentation implementation*: are the presentation layer patterns flexible and easy to use?
- *Ease of business logic implementation*: is it easy to develop and modify the business logic?
- *Ease of database administration*: is it easy to enable and change the database configuration?

In addition to the specific ERP framework comparison criteria listed above, Gerdessen [3] proposes the following general criteria for comparing frameworks in any domain:

- *Availability*
- *Customizability*
- *Modifiability*
- *Integrability*
- *Flexibility*

In the next section, we evaluate Seam and OFBiz according to both the ERP-specific and general criteria.

III. Framework Evaluation

This section evaluates the Seam and OFBiz frameworks according to the ERP-specific and general criteria listed in Section II-D.

A. ERP-Specific Criteria

1) Technology complexity: We find that JBoss Seam requires developers to clearly understand Java EE technology, but OFBiz does not require deep knowledge of any one technology. A Seam developer need skills in JSF, EJB3, and JPA. In contrast, for OFBiz, a basic knowledge of the architecture of the framework is sufficient for getting started. In the data source layer in particular, OFBiz's Entity Engine is straightforward for any developer familiar with relational database technology, but JPA requires in-depth knowledge of object technology and object-relational mapping. Although Seam's complexity may be necessary for a large project, OFBiz technologies are easier to understand and well tailored to smaller ERP projects.

2) Ease of service exposure: In our prototype, we found that OFBiz provides simpler means for exposing business logic as SOAP Web services. As already mentioned, OFBiz forces developers to adopt a service-oriented interface to the business logic layer. Once an OFBiz service has been defined, it is a simple matter of adding the attribute `export="true"` to the XML service definition:

```
<service engine="java" name="calculateDepre"
    location="org.ofbiz.ams.DepreciationServices"
    invoke="calculateDepre"
    default-entity-name="AssetExts"
    auth="false" export="true">
        <!-- code for service here-->
</service>
```

With the `export` attribute turned on, OFBiz automatically creates a WSDL specification for the service. In Seam, to export existing session beans' methods as Web service operations, developers must create a separate Java interface declaring the methods that should be exported then add two annotations: `@Remote` and `@WebService(name = "...", serviceName = "...")`. Like OFBiz, Seam automatically creates the appropriate WSDL specification. We conclude that although the Seam solution would be better when developers want a coarse-grained remote interface and a fine-grained local interface to the same business logic, the OFBiz approach is simpler.

3) Ease of administration: Both frameworks require the Java Development Kit (JDK). OFBiz ships with an embedded version of the Tomcat application server, but Seam requires separate installation of the application server and some additional configuration to notify the server the Seam should be loaded before it is ready to run. At application deployment time, OFBiz is capable of dynamically loading controllers, screens, widgets, forms, and business logic implemented by scripts. This saves development time and deployment time. OFBiz makes installation and deployment easier.

4) Resource utilization: In the standard setup, Seam runs on the JBoss application server, which provides complete EJB3 support, whereas OFBiz, by default, runs on the more lightweight Tomcat application server. In our case study, we found that the Seam installation required 335.2 MB of disk space and 85808 KB of RAM at runtime, whereas the OFBiz installation required 193.2 MB of disk space and 75820 KB of RAM. OFBiz is therefore more lightweight than Seam in terms of resource utilization.

5) Ease of presentation implementation: Both Seam and OFBiz utilize the Model-View-Controller (MVC) pattern in the presentation layer. For Seam, developers must be familiar with JSF and EJB to implement presentation layer components. OFBiz developers need only know how to use the Entity Engine, Service Engine, screens, form widgets, and the built-in templates. In developing our prototype AMS systems, we found that the learning for JSF is steeper than that for OFBiz. However, experienced Java developers already familiar with JSF but not with the OFBiz framework would presumably find the OFBiz learning curve steeper.

6) Ease of business logic implementation: OFBiz makes business logic layer services available to many clients automatically, leaving the developer with the relatively straightforward task of plugging in small segments of reusable business logic code in Java or a scripting language. In our case study, we found that Seam, on the other hand, sometimes requires more hand coding than OFBiz for the same task.

7) Ease of database administration: Both Seam and OFBiz provide means to connect with several different database management systems. In both frameworks, changing the database is a simple matter of modifying a configuration file to point to the correct data source. However, the OFBiz framework does have the advantage that developers can create, import, export, and seed data in an XML file format using a simple Web interface.

B. General Criteria

1) Availability: We have not been able to find any data on the relative failure rates of the Tomcat and JBoss application servers. However, we do find that JBoss requires server restarts more frequently that OFBiz when deploying new versions of the code. This means that OFBiz provides slightly more availability than Seam.

2) Customizability: JBoss Seam is in a sense more customizable than OFBiz in that it does not restrict the developer in any way. OFBiz developers must adapt to its style. On the other hand, OFBiz provides many customizable features, for messaging, database vendor, or work flow with XPDL standard. OFBiz has many common services that can be invoked in any application. The developer only needs to change a simple XML specification to switch the service endpoint to a different component. Therefore, it is very easy to customize OFBiz if what the developer wants to do fits into the OFBiz framework. Since OFBiz is optimized for the common cases in ERP development, in most cases there will be a fit.

3) Modifiability: In Seam, the application architecture is up to the developer. This means that the modifiability of the final system depends on the developer's skill. OFBiz, on the other hand, requires developers to use a service-based organization that forces them to build a loosely-coupled system. Since OFBiz is loosely coupled by design, it encourages modifiability more than Seam does.

4) Integrability: As previously explained, OFBiz makes it simple to expose any business logic service as a SOAP Web service endpoint. In this simple case study scenario, OFBiz exhibited more integrability than Seam. However, we should note that although OFBiz is the better framework according to this test, we have only tested one small aspect of integration. Other types of integration requiring standard Java APIs like JMS would not be advantageous for OFBiz, and integration tasks that benefit from JBoss' special features, e.g. jBPM, would put OFBiz at a disadvantage.

5) Flexibility: OFBiz is more flexible than Seam framework because it has many management and support tools for tasks such as user interface developement, web service exposure and database setup. With the OFBiz framework, we have more choices of scripting languages.

IV. Conclusion

In this paper, we compared two frameworks, OFBiz and JBoss Seam, using a typical enterprise application integration scenario. We find that for simple integration scenarios like the one presented here, OFBiz is better according to almost every criterion. It is therefore an excellent choice as a lightweight and easy-to-learn alternative to the full Java EE stack. However, we expect that Seam's complexity would eventually prove beneficial as the system size and complexity increases.

Acknowledgments

This research was supported by scholarships from Haadthip Public Company Limited and the Royal Thai Government to NR. We are grateful to Haadthip Public Company Limited for the case study scenario. We thank Vatcharaporn Esichaikul, Donyaprueth Krairit, and Chokchai Phatharamalai for helpful comments on this work.

References

[1] R. E. Johnson, "Components, frameworks, patterns," in *SSR '97: Proceedings of the 1997 symposium on Software reusability*. New York, NY, USA: ACM, 1997, pp. 10–17.

[2] M. E. Fayad, D. S. Hamu, and D. Brugali, "Enterprise frameworks characteristics, criteria, and challenges," *Commun. ACM*, vol. 43, no. 10, pp. 39–46, 2000.

[3] A. Gerdessen, "Framework comparison method: Comparing two frameworks based on techincal domains focussing on customisability and modifiability," Master's thesis, University of Amsterdam, August 2007. [Online]. Available: http://homepages.cwi.nl/~paulk/thesesMasterSoftwareEngineering/2007/AntonGerdessen.pdf

[4] M. J. Yuan and T. Heute, *JBoss Seam: Power and Flexibility Beyond Java EE 5.0*. Prentice Hall, 2007.

[5] Apache Software Foundation, "OFBiz, the Apache Open for Business Project — Open Source E-Business / E-Commerce, ERP, CRM, POS," 2008. [Online]. Available: http://apache.ofbiz.org

[6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[7] M. Fowler, *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.