

SOFIA

Salmon Open Framework for Internet Applications

by Salmon LLC

REVIEWED BY JOE GRIMES & BOB ROMITO

joe_grimes@hotmail.com bob_romito@hotmail.com

Salmon LLC

584 Ardsley Blvd.
Garden City South, NY 11530
Phone: 516 485-1118
Fax: 516 485-1122
Web: www.salmonllc.com
E-mail: safia@salmonllc.com

Specifications

Platforms: Windows NT 4 and 2000, JDK 1.3.1 or 1.4 depending on the IDE used
Pricing: Free of charge, open-source GPL license

Test Platform

Dell Latitude C610, 1GHz Intel Pentium III processor, 28GB disk, 512MB RAM, Windows 2000 with Service Pack 2

We work in the IT services department of a large insurance company and were asked to rewrite an old PC-based finance application using a Web-based Java solution. The project development team's background was based on mainframe technologies with some client/server and Web experience (HTML, ASP, basic Java, and JSP).

Our first attempt to build a different server-side Java application ended in frustration and an ungainly heap of code. Java scriptlets intermingled with HTML meant that organizing the JSP assets was cumbersome and Java debugging almost impossible. Furthermore, we couldn't believe that we were developing in a "modern development environment" without visual tools! There had to be a better way...

Salmon LLC, a New York-based consulting company, had recently completed a project for us using SOFIA (the Salmon Open Framework for Internet Applications). SOFIA, now available as open-source software, seemed to address some of our concerns regarding separation of code and gave us a visual environment to develop in.

What Is SOFIA?

SOFIA is a J2EE-based class and tag library for building database-driven Web applications. Conceptually it's similar to other open-source frameworks like Apache Struts. What makes SOFIA stand out from other frameworks is the built-in tools integration.

SOFIA integrates with Macromedia's Dreamweaver so that visual portions of an application can be "painted" instead of hand-coded. SOFIA also provides code generators for many nonvisual tasks such as database access and event handling. Finally, SOFIA integrates these tools, along with J2EE servers (Apache Tomcat, BEA WebLogic, IBM WebSphere), into best-of-breed integrated development environments (IntelliJ IDEA and Eclipse).

SOFIA's goal is to improve programmer productivity while avoiding vendor lock-in. It does a good job of meeting its stated goals. SOFIA integrates the products from several different vendors into a cohesive development environment. The various tools supported by SOFIA can be swapped out to suit individual

tastes and needs as well as any budget considerations. For this project we used Eclipse 2.0, Dreamweaver MX, and Tomcat 4.0.3 on MS Windows 2000 accessing a mainframe DB2 database.

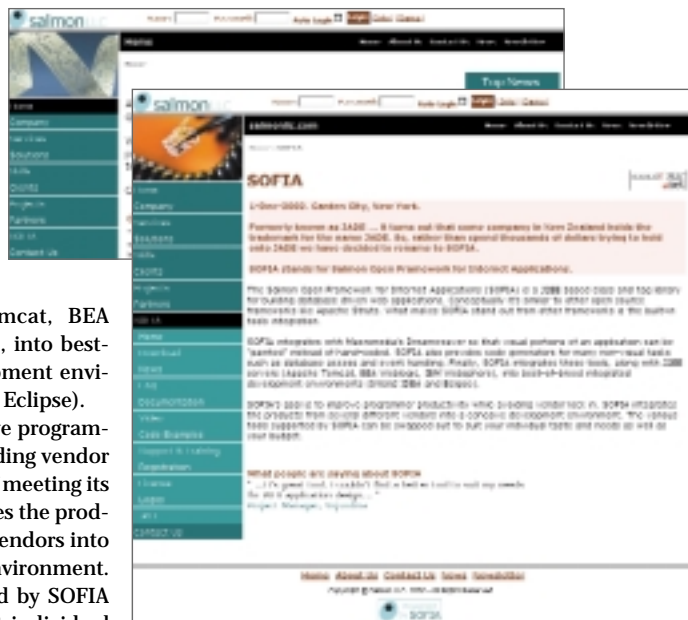
Dreamweaver has been extended to provide the ability to "paint" screens using SOFIA's drag-and-drop components so results can be seen in real time. The Dreamweaver integration provides a lot of power and enhances productivity. For developers using Unix or Linux, Dreamweaver is currently not available - a text editor would be needed to edit the JSPs. However, deploying finished applications to these operating systems should be fine.

Installing SOFIA

SOFIA is unusual in that it helps combine tools from a number of vendors. Some of the tools are open source, such as Eclipse, Tomcat, and the MySQL database engine, while others are commercial products, such as Macromedia's Dreamweaver and IntelliJ's IDEA. This can make the installation somewhat time-consuming since the commercial tools must be downloaded from the different vendors' Web sites and installed individually prior to launching the SOFIA installation program. However, once that's done, the menu-driven installation program ties the many pieces together automatically.

Features

SOFIA is architected around the MVC design pattern to cleanly separate the business logic from the user interface. This leads to easier code maintainability. Unlike some frameworks that require the developer to



write tedious code to integrate the various MVC components, SOFIA automates this process so that no application code is required to move the data from form fields on the browser page to server-side model components. SOFIA also provides a persistence framework so that getting the data from the models into the database can be accomplished with only a few lines of code. SOFIA handles all the underlying “plumbing” required to achieve these tasks transparently. We did not have to be concerned with form submission logic, embedding Java in-line in a JSP, or writing a lot of JavaScript. This, in turn, allowed us to focus on the business logic.

SOFIA provides a palette of over 50 GUI components implemented as a JSP custom tag library. There are simple components that provide support for form fields and more complex components such as trees, navigation bars, and data grids. This in and of itself is a very powerful feature, but it is the Dreamweaver integration that really breathes life into SOFIA (see Figure 1). There are SOFIA extensions for Dreamweaver to

support the SOFIA tag library in much the same way it handles standard HTML tags. This, in effect, creates a GUI development environment for JSPs similar to Visual Basic or PowerBuilder.

The SOFIA custom tag library enables a developer to create a static rendition of the page, which can be made dynamic through the use of controllers. With SOFIA, Java code is placed in the controller and a separate class is edited in the IDE, rather than in the GUI tool. SOFIA allows GUI runtime dynamic manipulation through the Java code in the controller and so requires no in-line code in the JSP. This separation allows for easy debugging, so we didn’t have to step through the machine-generated servlets to locate a problem in our page.

To further simplify some of the coding tasks, SOFIA provides functionality directly from the IDE toolbars to generate the code for models, controllers, and forms. For instance, a model can be automatically generated by simply choosing the database tables and columns required (see Figure 2).

When it’s time to test the application, the IDE integration will automatically redeploy code changes to the Tomcat application server and launch the browser so that results may be verified with a minimum number of clicks.

Conclusion

The Salmon Open Framework for Internet Applications enabled us to build and deploy a server-side Java application in a short amount of time. We found the framework easy to use and understand. Salmon LLC provided excellent documentation, examples, and support to help us get our project started. With integration and support for tools such as Dreamweaver to create the view and Eclipse to develop the model and controller, SOFIA makes for a productive visual development environment. To us the most valuable feature of SOFIA resides in its handling of all the tedious plumbing associated with developing business applications, thus enabling us to concentrate our efforts on satisfying the business requirements. ☘

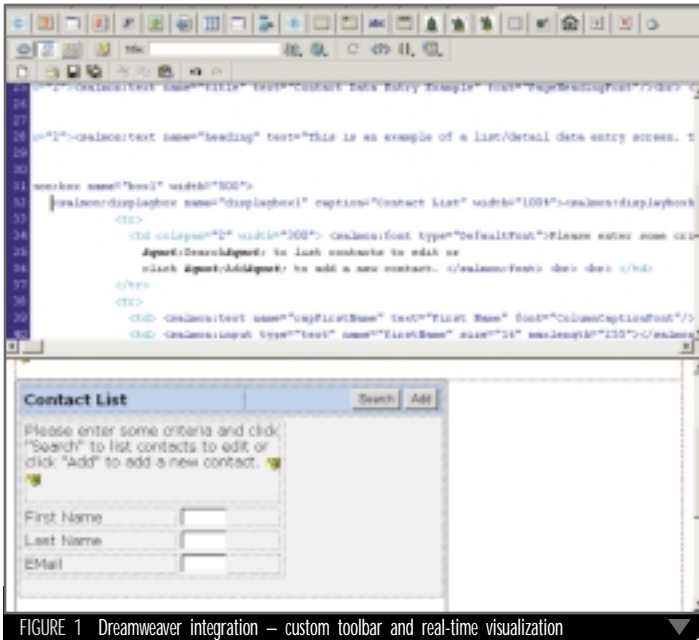


FIGURE 1 Dreamweaver integration – custom toolbar and real-time visualization

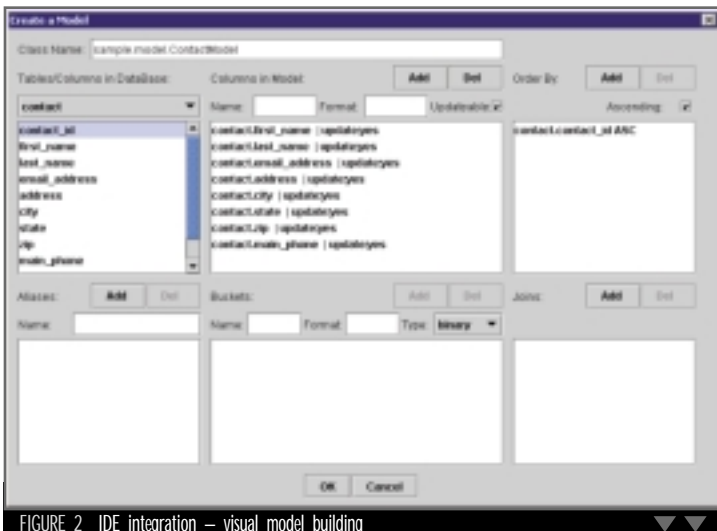


FIGURE 2 IDE integration – visual model building

Product Snapshot

Target Audience: Java programmers, Web designers

Level: Intermediate to advanced

Pros:

- Extensive custom tag library
- IDE integration (Eclipse 2.0 and IntelliJ's IDEA 2.5.2, 2.6, 3.0)
- Dreamweaver integration

Cons:

- Complex install process