Outsource through the browser

Argun

The Web 2.0 delivery framework

Highlights

- In-browser development environment.
- Page flow engine with a visual flow editor.
- Visual process editor.
- In-browser diagram editor.
- WYSIWIG page authoring.
- Fine-grained security model.
- Built-in issue management.
- Easy integration with J2EE applications.
- File attachments.

Executive summary

Argun is a Web 2.0 development and execution environment. As other Web 2.0 systems it allows to harness network effects in the enterprise. But unlike other products Argun brings collaboration to the next level by allowing not only share information but collaboratively create executable systems in the in-browser development environment. Systems built in Argun are alive and executable from the first second.

All artifacts in Argun are organized in a tree. Tree structure provides clear escalation paths, and facilitates progressive realization and refinement. The system grows as a tree.

Argun runs in a servlet container and as such can be easily integrated with J2EE applications. In can be mixed-in to an existing Web/J2EE applications to incrementally provide Web 2.0 capabilities without re-toolings and big-bangs.

Argun makes Web development available for semi-technical people by providing in-browser easy to use development environment with WYSIWIG page editor, requirements capturing support, visual page flow editor, and context-sensitive help system.

In-browser development environment also makes outsourcing easy by significantly simplifying costly and time consuming on-boarding process.

In Argun Web pages and page flows can be developed using scripted Java and JSP. This approach obviates compile/deploy steps during development and allows quickly semi-automatically migrate to compiled Java/JSP once functionality is developed and tested.

Feature summary

- Development and execution through the Web browser.
- Rapid development with scripted Java and JSP, semi-automated conversion to regular Java and JSP.
- Visual page flow editor, page flow engine.
- WYSIWYG page authoring.
- Runs in a servlet container.
- Can be mixed-in to existing Web/J2EE applications.
- File attachments.
- Advanced help system.
- Fine-grained security model.
- Requirements capturing, generation of tables and web page prototypes from analysis terms.
- Wizard to generate pages for database tables.
- Database console with metadata browser.
- Java console.
- Issue management.
- Configuration registry

Overview

Every year software becomes more and more complex. Software methodologies become more and more exquisite. As a result, creating a "Hello world!" web application using enterprise technologies is quite a challenge. The software development industry seems to be forgetting the KISS principle. The industry is driven by vendors and high-paid consultants who have time and money to create intricate tools and techniques and write books about them. While these tools and techniques can be understood and effectively used by gurus, they leave a regular person completely lost. The problem here is that most of the people in the industry don't have time to catch up with latest and greatest advances of thought leaders. They need simple and robust tools and methodologies to deliver business value.

Outsourcing seems to be a very good approach to cut costs because of low cost of labor overseas. In practice, though, it is very difficult to establish an effective outsourcing model because of

- On-boarding costs. Offshore consultants have to be connected to the client network and to have development tools in accordance with client's requirements. Also, time and training is needed for offshore consultants to get familiar with client's environment and the problem at hand.
- Communication barriers. Time difference is the first of them. For example, there is no overlap in business hours between USA and India.

The goal of the Argun framework and the Hammurapi Group development model is to make simple things simple again. We do it through organizing our product and processes around principles easily understood by semi-technical people and by providing practices to break problems into small chucks, which can be implemented by offshore workers using in-browser development tools, which are part of the Argun framework.

Argun overarching principles:

- Fine grained, small well documented steps or tasks.
- Web delivery.
- Easy delegation.

- Small pieces of well documented functionality and information in well defined places.
- Incremental approach.

People live and think in hierarchies. We live in countries divided into states divided into counties divided into cities … We work in corporations divided into lines of businesses divided into departments … We break down projects into a tree structure (WBS). In software many application user interfaces have menus organized into trees. In programming languages, such as Java, classes form inheritance hierarchy and also package hierarchy. Hierarchies provide clear taxonomy, escalation path, and inheritance of features.

Because the paradigm of hierarchies is so ingrained into human nature, Argun framework leverages the concept of menu hierarchy as its most fundamental principle.

Menu is the skeleton of the application. Many system objects are associated with menu items. Examples include issues, interactions (page flows), help topics.

Each menu item can have owner(s). Tree structure of the application defines clear escalation paths for problems. Metrics such as number of visits or resource consumption (e.g. effort) can be aggregated along tree branches.

All significant components shall be mapped to a menu item including non-visual components. The truth is, an application component which is not visible for the end user of the application is "visible" for developers of components which depend on this component. A menu item for such a component becomes its home page with associated documentation, configuration screens, performance dashboards, etc.

Argun is non-intrusive. Menu items are matched against URL's. Therefore existing web pages can be wrapped into a menu without any modification.

Argun features a number of productivity boosters. These include generation of tables and web pages from analysis documents using term tagging, development using scripted Java and JSP, Java -> XML -> HTML pipe, dispatching of URL's to Java methods using naming conventions and thus reducing number of configuration files, multi-staged page instantiation, smart links, WYSIWIG HTML editor, in-browser development without the deployment step, database access framework, dynamic generation of data access and data transfer classes, database backed property sets, …

File attachments

The Argun framework features a virtual file system. Files are stored in the database. Any system object can have an associated file system.

The framework provides screens to browse the file system, view, download, edit and upload files.

The file system features extensible list of file types and associated actions. File types are organized into a hierarchy by their MIME type.

In particular, out-of-the box file actions include

- In-place editor for text files.
- In-place WYSIWIG editor for HTML files.
- Text and HTML file viewers.
- Glossary and analysis documents for requirements capturing.

Page flow (interaction) management

The distinguishing features of Argun page flow engine are:

- Visual in-browser flow editor
- Simple graphical notation
- Loose coupling of interaction steps with pages and sub-flows
- Progressive realization
- Flows are persisted in the database and can be suspended, resumes and transferred between users
- Form fields can stored in the interaction

Argun uses the term **interaction** to refer to a page flow primarily because the flow can include not only pages but other interactions and automated steps.

The visual interaction editor uses a simple graphical notation, as people draw it on dashboards.

The problem with notations like UML or BPMN is that people who know how pages shall flow typically don't know UML or BPMN. As such they draw boxes and arrows and somebody translates their drawings to the notation used by a flow authoring tool. The translation step consumes time and resources. What is worse is that after the translation the person who created the original flow diagram with boxes and arrows cannot validate that the translation was done correctly.

Argun interactions are drawn as boxes and arrows with supporting descriptions. Then, logic is added to the flow by providing guard and action code to interaction steps and transitions.

The interaction (flow) diagram doesn't specify pages to be used on each step. Pages or interactions are "bound" to steps at a later time.

Not bound steps are executed without user interaction (automated step). Automated steps help to implement complex decision logic. They also allow adding functionality to the interaction progressively, page by page.

Interaction steps can be bound to a menu item in Argun, to another interaction or to a URL. Binding to URL allows building interactions which span multiple applications. An external page shall implement a very simple contract in order to be part of interaction – it has to redirect to the interaction controller passing interaction ID and step ID. Interaction controller URL, interaction ID and step ID are passed to the external page as request parameters.

Loose coupling between pages and interactions allows developing and testing them independently. Also, multiple interactions can share the same page or sub-interaction.

Each interaction, step and transition has an associated database-backed property set. HTML forms can be posted to the interaction controller which will save form fields in the step or interaction property set.

Security

Argun security model consists of Users, Groups, Roles, Permissions, and Menu items. A user can be a member of a group or be assigned a role. Groups can be assigned roles as well. These security objects can be granted or denied permissions or access to menu items.

Information about Users, Groups and Roles can be stored in the Argun database or obtained from external sources, e.g. J2EE security or a Single Sign On system.

Permissions are used in programmatic security. Association of users, groups, and roles with menu items represents declarative security.

Issue management

In Argun issues, as the most of objects, are associated with menu items. This gives advantage of clear ownership of issues and clear escalation procedures.

For users with appropriate permissions (e.g. testers or users participating in UAT) an icon to report issues is displayed on each page. Argun captures issue context, user needs to enter only few issue attributes such as description, severity.

Help system

Argun help system provides contextual knowledge in an active environment. Users get help which they need when they need it.

Help topics in Argun are associated with menu items. Each menu item can have its own tree of help topics.

Because help topics are associated with menu items, users can see help only for functions of the system they have access to. E.g. an accountant using a financial system will not see help topics related to administering the system. This approach eliminates "noise" in the help system and allows users to quickly find answers.

The motto of Argun help system is "Write once, display anywhere". Help can be content can be delivered in three ways:

- A tooltip associated with a page element, e.g. form control. Tooltips can display up to 1 kb (configurable) of HTML text. Delivering help in tooltips increases user productivity by obviating context switches between the application window and the help window.
- Traditional help window with a help topic tree view and search function. Terms referenced from a page can also be delivered as tooltips sparing users from jumping between help pages.
- As a single HTML or PDF document. This presentation type is useful for new system users.

External HTML documents (e.g. JavaDoc, online tutorials) can be mounted the help system tree.

Transclusions allow defining a piece of information in one place and then including (in-lining) it to multiple help topics. As such information can be conveniently delivered and there is no need to update the same information in multiple places.

Process modeling

Argun's visual process modeler uses simple visual notation (box and arrow, similar to PERT) to document processes. Process modeling supports definition of tasks, roles, artifacts (task inputs and outputs).

Future versions will include a process execution engine and events bus. It will be possible to associate processes with events to start them automatically.

State machines

Argun features visual in-browser state machine modeler to model lifecycles of system objects. Future versions will include state machine execution engine.

Analysis

There are two special file types in Argun: Glossary and Analysis document. These files support tagging of document text to form a glossary of terms. Terms then can be used to generate database tables, web pages and other artifacts.

These file types are used by business analysts to capture requirements and use cases.

Use

Argun can be used as an independent application, be a front-end for an EJB application, or can be mix into an existing web application.

Development and delivery

Argun is the key enabler of Hammurapi group offshore development model.

Our US team works with customers to build identify requirements and build the skeleton of the application. Then our offshore team adds flesh to the bones. In this model the customer personnel defines the WHAT because they know what they need. The Hammurapi Group developers define the HOW because they know how to use Argun framework. Speaking in other words, the customer with the help of Hammurapi Group US staff ensures that the right system is being built; Hammurapi Group developers and architects ensure that the system is being build right.

Hammurapi Group development facility is located in the city of Khabarovsk (GMT+10).

Development is done on hosts accessible by both the customer and the offshore team. This approach allows to start development very quickly by providing ready-to-use environment. Development can be hosted in the customer environment if the customer chooses so.

When solution is developed the customer can continue to host it in the Hammurapi Group environment or the solution can be migrated to the customer data center.

Ossification

In nature many organisms use different tissues to grow their body and to support it. For example, in human embryo the skeletal pattern is formed in cartilage which is then

converted into bones. This process is called ossification. Tree branches are supple when they grow and become thick later to support increased weight of branches, twigs and leaves growing off them.

In software development products are often prototyped using scripting languages and then developed using strongly-typed languages. The problem with this approach as compared to the nature's one is that prototype is typically thrown away.

Some of you probably frown reading about scripted Java and JXP. Scripted languages are weakly typed and interpreted. Java is more robust because it is strongly typed and faster because it is compiled. This is all true.

The goal of the Argun framework is to achieve functionality first. As an old maxim goes, "It is easier to make a functional application fast than a fast application functional".

In the Argun framework scripted Java and scripted JSP's (JXP) is used for rapid development. It gives the advantage of obviating the deployment step – changes in the code are immediately available for execution and testing.

These scripted pages can be used in production environment as well; many production systems use scripted languages. But if there is a need, scripted pages can be semi-automatically "ossified" into regular Java classes and JSP pages. In this case there is no throw-away prototype code. The framework automatically generates classes, JSP's and updated bindings. The manual part includes replacement of untyped declarations with typed ones and extracting of embedded SQL statements into SQLC tasks. This is a mechanical activity which can be accomplished by a junior-level developer.

Process definitions and interactions can also be ossified into Java classes.

Inoculation

Continuing analogy with live nature, there is a concept of inoculation in Argun.

First of all, Argun itself can be inoculated to an existing Web application and wrap existing web pages and/or provide services like fine-grained security or metrics collection.

Argun menu item location and link instantiation mechanism allows to inoculate branches of functionality to Argun applications and incrementally add functionality.

Argun provides a way to build one application with a growing set of capabilities instead building multiple applications with pieces of overall functionality needed by the end user.

For example, a company builds a financial application and buys an accounting module from some vendor. That module can have links to the financial analysis module menu items. If the financial analysis module is not installed these links will not be instantiated, they will be either invisible or rendered as a plain text, not as a link. When the financial module is installed accounting module pages will display links to the financial module.

Model driven development

Argun has an extensible library of code templates. These templates are used for generation of web pages and code from models.

Argun features the following out-of-the box generators:

- Database tables to web pages.
- Analysis terms to database tables.
- Analysis terms to web pages.

Developers can define custom models and leverage Argun's code generation infrastructure. They can also leverage the database model to generate custom data access and data transfer classes.

Coming soon

- Process engine
- State machine engine
- Event bus
- Inter-user messaging with e-mail gateway.

References

- Hammurapi Group home page
  http://www.hammurapi.biz