JtPatternFrameworkProposal

Jt Proposal

Project Name: Jt Pattern Oriented Framework

Introduction

This proposal describes a Pattern Oriented Framework for the rapid implementation of Java applications. This integrated framework is based on a messaging architecture which provides strong encapsulation and loose coupling; framework components can be interchangeably plugged into complex framework applications using a “Lego" approach. This framework provides capabilities for the implementation of applications based on design patterns. The framework itself is conceived, from the ground up, based on design patterns. Jt is probably one of the first Pattern Oriented Frameworks (open source).

Goals

The framework addresses the following goals:

A) The pattern oriented framework implements and/or facilitates the implementation of well-known design patterns like Gang Of Four design patterns (GoF) and J2EE Design patterns. The framework itself is conceived and implemented based on design patterns (from the ground up). The framework also facilitates and accelerates the implementation of applications based on design patterns.

b) The framework architecture is based on a messaging design pattern: framework objects are able to interchange information and perform computations by sending, receiving and processing messages. A messaging API provides strong encapsulation and loose coupling; framework components can be interchangeably plugged into complex framework applications using a “lego/messaging” architecture. The framework takes full advantage of the power and simplicity of the messaging design pattern/API.

C) The framework lego/messaging architecture provides transparent access to remote components: remote framework objects are treated as local objects. Design patterns implemented by the framework (adapters, remote proxies and facades) make this possible by hiding the complexities associated with remote APIs.

D) The framework provides transparent integration with other technologies via framework adapters, proxies and the implementation of related design patterns. These technologies include BPM, Data Access Object implementations (DAO), Model View Controller implementations (MVC), EJBs, AJAX, JMS, XML and Web Services.

E) The framework is designed to be lightweight and fast (low overhead/small footprint).

F) The framework messaging/lego architecture should improve and simplify design/development efforts. There should be a tight correspondence between UML design diagrams and the framework messaging based applications and components needed for the implementation. The framework should provide wizards and automated capabilities for generating framework applications. Framework components should be easily added to BPM process diagrams. In future versions of the framework, it should be possible for application modules to be generated directly from the UML design diagrams. This goal is still work in progress. There is an early version of the Jt Wizard.

G) The framework messaging architecture facilitates testing and debugging efforts. It provides capabilities for testing components as independent units by sending messages to the component and verifying the expected reply messages.

H) In order to provide additional productivity, the framework has been integrated with open source IDEs.

Main Features

- Implemented J2EE design patterns include J2EE business delegate, J2EE Session Facade, J2EE Service Locator and J2EE Value Object.
- Web Services integration via the implementation of Web Services adapters and proxies. The Jt messaging API greatly simplifies the development and deployment of web services.
- Integration with the MVC (Model View Controller) design pattern and Ajax. Universal Jt components and adapters provide a transparent interface between the Jt framework API and these technologies. The business logic (controller piece) can be implemented using Jt framework components and/or BPM business processes.
- Integration with J2EE Enterprise Java Beans (EJBs) via Jt Adapters and proxies. EJB clients are able to gain transparent access to remote framework objects. No need to deal with the complexities of EJB application development. An implementation of the J2EE Service Locator pattern is also provided.
- Easy customization of framework applications. This is done via resource files: object attributes can be automatically loaded from a resource file.
- Integration with the XML APIs via XML adapters, helpers and built-in bean/XML mapping capabilities.
- Built-in logging/debugging capabilities. Messages between framework objects are automatically logged. This simplifies the debugging and testing tasks.
- Built-in testing capabilities.
- Efficient and lightweight in terms of memory utilization.
- Integration with the asynchronous Java Message Service (JMS). Jt messages can be sent and received via JMS adapters.

Current Status and Community

There are several people involved with the project. The software base is very stable. Several production quality applications has been built based on the Jt framework. One of the next objectives is to continue growing an active community of users and contributors. It is expected that community will continue to grow. So far the main focus has been the completion of the core framework functionality. We see Jt becoming an ASF incubator project as a next step in its evolution.

Meritocracy

Meritocracy will be fostered and encouraged within the project. We will follow the guidelines of the Apache Software Foundation. Community contributions and feedback are always welcome. We also plan to actively encourage individuals to get involved in the project.

Community

One of the next objectives is to continue growing an active community of users and contributors. It is expected that community will continue to grow. So far the main focus has been the completion of the core framework functionality. We see Jt becoming an ASF incubator project as a next step in its evolution. We plan to open the project to a wider audience in order to achieve a larger and more diverse community.

Alignment

Jt is aligned well with existing Apache projects and technologies. The framework has been integrated with several Apache technologies including Struts (MVC design pattern) and Axis (Web Services Adapter). The Jt automated Wizard capabilities has also been built based on Apache products.

Known Risks

Orphaned products: The projects has been growing as an open source project for several years.The team is committed to continue this work therefore the risk is fairly small. A preliminary Java Specification Request (JSR) is being worked on.

Inexperience with open source: Some of the current contributors have experience working with open sourceprojects and communities. The team expects to gain additional experience via the ASF incubator community.

Homogenous developers: The people involved with the project work for several organizations who are geographically distributed across the world.

Reliance on salaried developers: This project does not rely on salaried contributors. They work on a volunteer basis.

No ties to other Apache products: As described in the alignment section, this framework has been integrated with many Apache products. The project expects to adopt the Apache License 2.0

An excessive fascination with the Apache brand: This project started outside the Apache Software Foundation. It has been evolving for several years. Hopefully other Apache projects and initiatives will benefit from the capabilities provided by Pattern oriented framework.

Documentation and sources

Project information and complete documentation/sources can be downloaded from http://jt.dev.java.net/. java.net forums and mailings lists are not used. We plan to move to a new location which should serve as a central location for the project.

Apache sponsor

TBD.

ASF resources to be created

- Mailing lists
- A subversion repository
- A JIRA issue tracker

Contributors

- Dan Evans
- Cliff Campen
- Tuomas Kassila
- Pretti Sharma
- David Sheppard (Ed)

Getting involved

Please send a message to the list if you are interested in joining this project or using the framework for your particular application.