ChoosingTapestry

NOTE: This information is from an evaluation conducted in June 2004.

Introduction

We recently had to make a web application framework choice. We reviewed several, including:

- Cocoon
- Tapestry
- Maverick
- Struts

After a lot of reading, we chose Cocoon. After trying to use it, we went back to the drawing board with a well formed process. We applied that process to Cocoon and Tapestry. The following outlines our experience, but more importantly it outlines the process and evaluations that we made.

Process

Up and running out of the box

This included the following steps in an effort to follow existing documentation and make an initial evaluation.

- Download the software
- Find the install, build, and run documentation
- Build the software
- Launch the examples and poke around

After these steps were complete, we evaluated frustration, ease of installation, ease of getting the samples running.

- Cocoon passed
- Tapestry passed

Get a reasonable development environment rolling

This included finding the parts of the tool required to build out application, and getting them integrated into our application build environment. Our environment uses Eclipse for daily development, and Maven/Ant for nightly builds and testing. This environment is pretty standard and easy to integrate.

- Cocoon had problems here, but I got it working
- Tapestry, because of Spindle, passed with flying colors

Try to get a prototype application built and running

The title says it all. We had a small prototype that had to meet the following criteria inorder to pass.

- prompt for a user and password to login
- show a form with two fields after logged in
- show navigation when the user has logged in
- don't show navigation when the user is not logged in
- logout

After the prototype was built, we made several detailed evaluations.

- Gave up on Cocoon (see evaluations below for reasoning)
- Tapestry passed with flying colors

Evaluation

Robust and Helpful Community

Cocoon passed with flying colors Tapestry passed with flying colors

Robust and Available Documentation

Cocoon passed with flying colors Tapestry passed with flying colors

How much Non Application Plumbing=

This evaluation tried to determine the amount of extra stuff that adopting a particular framework imposed upon the application. This included items from the following list:

- Implementing interfaces for framework, not application purposes
- Configuration for framework, not application purposes

This is why I gave up on Cocoon, to much XML plumbing, to flexible, to many features.

Tapestry is very lean in this regard, I build a lot of application and very little plumbing.

Judge the maintenance aspects

Cocoon has some very good documentation about separation of concerns and how that helps, but in practice Cocoon simply does to much stuff and is easy to get off in the weeds for our staff.

Tapestry's strength interactive web applications. It also has excellent separation of concerns. This combined with the component model allows for a very modular application with modules that focus on a single task and can be integrated with ease. Needless to say, Tapestry passed with flying colors.

Judge the license for use in a commercial application

Both are ASF and Apache License, nuff said

Judge the learning curve

Cocoon is big, very big.

Tapestry has just enough to get our application built. The designer seems to have paid particular attention to having a small learnable API. Very good in this respect.

Conclusion

As you can tell, Tapestry was chosen for all the reasons stated here. An interesting point is that following were not high in priority during the evaluation.

- Has feature X, Y, Z
- Is an industry standard
- Has all sorts of features that might be nice in the future
- Application Performance

Fundamentally what was really important were the following:

- Can it be learned
- Can it be taught
- Can an application built using it be maintained
- Does it have enough features to build the application we need