uPortal 3.0: Containers for Pluto, The Exchange of E-mail

In a series of e-mail messages, there was a discussion of appropriate component container for uPortal that would be used in conjunction with the use of Apache Foundation's Pluto implementation of the JSR 168 portlet specification. Developers expect to use the Apache Pluto portlet container for both uPortal version 2.3 and subsequent uPortal 3.0.

The exchange is documented here so others can follow the discussion. These are offered in the sequence of messages as they are referenced or included in subsequent messages. Apparently a combination of message times given in different time zones and perhaps even miss-set computer clocks give times that do not appear to be in the same chronology as the messages themselves.

Ken Weiner, Unicon, Inc. January 20, 2004 4:34 p.m.

Michael,

After seeing the dev mtg topics, Bill Thompson from Rutgers mentioned to me that choosing Avalon over another IoC framework like Spring or Pico might be a bad idea. I think you should start a discussion on the mailing list about IoC frameworks. Then Bill and others would have an opportunity to voice opinions.

-Ken

Mike Ivanov, instructional media + magic, inc January 20, 2004 8:29 a.m.

Well....

My arguments against Pico:

Pico is a very simple container for very simple components that is not enough for uPortal needs.
Besides I have heard lots of negative opinions about PicoContainer - it is not an enterprise solution.

I would rather discuss Spring Bean Factory vs Merlin, since we are going to use Avalon with Merlin container.

PicoContainer is not a competitor here at all

From Mike Ivanov, instructional media + magic, inc. January 20, 2004 8:55 a.m.

I guess we need to discuss Merlin vs Spring to compare their good/bad sides, what impact on performance and scalability they have, how good they are for implementing complex component dependencies and so on.

From Mike Ivanov, instructional media + magic, inc. January 20, 2004 10:23 a.m.

Merlin is a container that provides comprehensive support for the management of complex component based systems and it is built on the Avalon framework.

You can read about Merlin here:
http://avalon.apache.org/merlin/index.html

I think the idea of serviceable components managed by the ServletManager complies with the portlet container requirements better than java beans.
The portlet container (we are going to use Pluto) will be given the necessary services through the Avalon ServiceManager by the portal server such as FactoryManagerService, LogService, InformationProviderService, PropertyManagerService and so on. All these services and their dependencies can be easiliy described in the appropriate configuration files.

Services is a major goal of using the Avalon framework.

Besides the Merlin container provides a comprehensive meta model supporting component deployment. The meta model provides a set of directives that provide the information necessary for component deployment.

Merlin provides support for cascading containers. This model enables component assemblers to (among other things) associate jar files under a protected block scope where each block is associated with its own classloader. Each block manages a single container. A container manages multiple components.

Merlin will handle resolution of service dependencies for components contained in containers by looking for explicitly declared components commencing within the local container, and working progressively up the container hierarchy. If no explicit solutions are resolved, Merlin will attempt to build an implicit solution based on components declared in the respective container classpath.

Folks,

We have been using the Spring Framework for over a year now and have three apps in production; Undergraduate Online Admissions, an enterprise time tracking system and a course scheduling application. We have also used the Spring JDBC abstraction layer for two uPortal channels; Grades and Schedule and are working on a model to leverage more of the framework for channel (portlet) development.

Spring has a fantastic architecture, nice AOP support, and a great leader in Rod Johnson (J2EE Design and Development [1]). Using Spring we have been able to build properly layered architectures and testable components.

I would say it is definitely worthwhile to consider Spring for uP 3.0 "The Spring Framework - A Lightweight Container"[2] is a nice introduction and has a comparison to Avalon and Pico.

later.

Bill

[1] http://www.wrox.com/books/0764543857.shtml
[2] http://www.springframework.org/docs/lightweight_container.html

From Dmitriy Kopylenko, Rutgers University January 19, 2004 6:34 p.m.

Haven't heard about Merlin. Have you looked into Hivemind also?

What I heard about Avalon that it's on the much more complex and intrusive side than Spring and Pico. It uses "Interface based dependency injection" (former Type1 IoC), creating an unnecessary dependency of application components on the container...

Pico is much smaller then Spring - just offering IoC container, but as of 1.0 it's getting better and now supports "Setter dependency injection (former type2 IoC)...

My opinion (may be a little biased ;-) would be - go with Spring. But yeah, you need to evaluate the others and choose what best suites your needs.

Regards,
Dmitriy.

From Jim Farmer, JA-SIG Collaborative January 21, 2004, 9:03 a.m.

To Glenn Golden, University of Michigan

Extensive exchanges on the uPortal Developers List yesterday suggested four containers to be used with Pluto: Avalon/Merlin, Pico, Spring and Hivemind.

Are there any you would recommend rejecting immediately? Do you have some suggestions, preferences, or advice that would permit Mike to narrow the focus.

I believe Mike was planning is considering Avalon/Merlin with Pluto subject to verifying performance. In one message he wrote:

Merlin is a container that provides comprehensive support for the management of complex component based systems and it is built on the Avalon framework.

You can read about Merlin here:
http://avalon.apache.org/merlin/index.html

I think the idea of serviceable components managed by the ServletManager complies with the portlet container requirements better than java beans. The portlet container (we are going to use Pluto) will be given the necessary services through the Avalon ServiceManager by the portal server such as FactoryManagerService, LogService, InformationProviderService, PropertyManagerService and so on. All these services and their dependencies can be easiliy described in the appropriate configuration files.

Services is a major goal of using the Avalon framework.

Besides the Merlin container provides a comprehensive mata model supporting component deployment. The meta model provides a set of directives that that provide the information necessary for component deployment.

Merlin provides support for cascading containers. This model enables component assemblers to (among other things) associate jar files under a protected block scope where each block is associated with its own classloader. Each block manages a single container. A container manages multiple components.

Merlin will handle resolution of service dependencies for components contained in containers by looking for explicitly declared components commencing within the local container, and working progressively up the container hierarchy. If no explicit solutions are resolved, Merlin will attempt to build an implicit solution based on components declared in the respective container classpath.

Appreciate your comments.

jim farmer
uPortal Project Administrator

From Glenn Golden, University of Michigan, January 21, 11:12 a.m.

Jim,

Here's what I am doing now in the CHEF 2 framework. I'm not sure how much this is of value to the uPortal decision; uPortal's component framework is private to uPortal and need not have any relation with the CHEF component framework. CHEF does not have to provide Pluto with it's services, and there are probably requirements that Pluto brings to the table about how the services it requires are provided, so I'm not sure how much of this is relevant to the uPortal needs.

* * *

My primary candidate component framework for CHEF 2 is based on the component framework provided by Spring. I've looked at Pico, and have already built a system based on Avalon Framework (but have not looked into the Avalon containers such as Merlin).

Note, the CHEF 2 component framework is based on just the low level bean factory in Spring, and is not using any of the other aspects of Spring. Also note that this decision is localized and that there are no dependencies to anything "Spring" in any CHEF code other than our CHEF component framework. We may switch out the Spring for something else without anyone noticing.

I selected Spring because it can be used to support various models of component dependency resolution:

Service Locator - where a client uses a service (component) manager to ask for component instances at runtime,

Constructor Injection - where clients declare their component dependencies in their constructors, and the component configuration of the application provides the necessary meta data to let the component framework automatically satisfy these dependencies as it constructs the component,

Setter Injection - where the clients declare their component dependencies (as well as their configuration options) in bean setter methods, and, like Constructor Injection, the framework takes care of setting things up when the components are created.

Spring lets us deal with configuration as well as component dependencies in the same application configuration format. It also lets us "auto-wire", so we don't have to specify so much in the configuration, but also lets us indicate specific component relationships for special cases.

- Glenn

From Peter Kharchenko, Unicon, Inc. January 22, 2004 3:47 p.m.

All,
I'd like to thank Jim for putting together all of the correspondence relevant to the discussion on the container choice for the uPortal 3 development.
Michael and I have been looking at different component frameworks for some time now, and I wanted to outline the features that we think will be particularly important, and how different containers appear from that point of view.

In developing uPortal 3 we would like to create a flexible architecture that will be easier to customize and maintain then what we currently have in the uPortal 2. A number of component frameworks are available to facilitate such development, each providing a set of tools to describe a configurable set of components and manage their lifecycle. The uPortal requirements emphasize specific aspects of these frameworks, and I wanted to convey them now, so that further discussions on the choice of the component framework can address the details.

1. The most important aspect is the eloquence of readily available component configuration method. An expressive configuration language should allow us to describe a large arrangement of components, with complicated build-time dependencies, multiple concurrent configurations and flexible lifestyles (singletons, various pooling strategies). Because the number of components will be quite large, it is very desirable to be able to define hierarchical groups of components.

2. The component framework should not produce any noticeable performance impact. This should be possible because uPortal will, primarily, use static configurations with very little dynamic resolution required.

3. The component framework should be fairly mature (with the expectation of full maturity within the uPortal 3.0 development time frame), well maintained and available under the friendly licensing terms.

These are, also, a number of smaller requirements, such as susceptibility to unit testing, general ease of use/deployment and amount of framework-specific code that will have to be built.

Merlin:

Comparing Merlin, Spring and Pico we found that Merlin had, by far, most elaborate configuration facilities. The main disadvantage is in the number of framework-specific interfaces that have to be implemented. Merlin also requires an explicit implementation of the initialization procedure (i.e IOC type1) and a somewhat awkward web app deployment.

Spring:

Quite nice, but seems to lack in some important features:

- ability to specify multiple configuration alternatives for the same component (i.e. XmlBeanFactory appears to tie component to a particular implementation class, and furthermore, allows for only single configuration profile)
- ability to arrange components into groups and specify group hierarchies.
- configure component pools. (although I've seen some source code implementing a pooling aspect, we would like to avoid AOP in the uPortal 3.0 development)

At this point, Merlin appears to be the first choice. It's quite possible that we've missed some of the features in the Spring framework, or some of the downsides in Merlin implementation. Please let us know if this is the case - we're just starting the development and, at this point, the choice of the component framework can be quickly reconsidered.

thanks,
-peter.

Jim Farmer wrote:

>Enclosed is the document containing the eight messages from the discussion on component container choice.
>
>You were correct, none of these were on the uPortal Developers list. I >deleted the reference to the list. You can share this document with them if >you feel it relevant.
>
>Mike said he was going to talk with Peter last night to discuss Merlin and >Spring; I haven't yet received an e-mail about that discussion.
>
>Mike also corrected my description of the relationship between Merlin and Pluto. His language, used in the first paragraph, was more precise than mine.
>
>jim farmer
>
>