What about the Future?

What New Java Language and Platform Technologies Might be Useful to Jini Developers?

09/13/2006

Gregg Wonderly
Cyte Technologies Inc.

Generics

- Problems
  - Generics don’t travel in serialization streams without factory generators.
  - Service Lookup won’t work with Generic types as parameters.
- Benefits
  - Generics get rid of “ugly” casting.
  - Generics can help find bugs.

Annotations

- Everyone keeps talking about using annotations for services, some people have started.
- Can we/should we use the same annotations as JEE?
- Are annotations a service startup enabler?
- Can we create a deployment mechanism through the use of annotations which might allow deployment tools to interact with a container?

java.util.concurrent

- The JTSK has threads pools of its own design.
  - It might make sense to move to the java.util.concurrent queues which have better scalability for certain load patterns.
  - Allowing Configuration of thread pool implementations seems to make sense.
- Futures might be a good thing to use for a new ServiceDiscoveryManager which could include rediscovery, which a lot of people seem to reimplement for firewall scenarios.

java.util.concurrent (cont)

- Asynchronous calls could use Futures.
  - Canceling would require a new invocation layer with explicit support for canceling.
  - The RMI programming model doesn’t have this today, but a new invocation mechanism based on futures could support it through the use of a Future on the service side too.

JMX Connectors

- We should define a JMX connector which uses service lookup and provides for managing all the issues about partial failure.
  – We should standardize a factory interface for generating a JMX connector.
  – This would let JMX enabled services provide a way to get a connector without having to know the wire connection details.

JMX MBeans

- We’ve talked about adding Mbeans to the standard JTSK services to allow insight into the operations of the services.
- We could also use MBeans to provide dynamic reconfiguration of parameters such as all Configuration values.

JMX MBeans (cont)

- Using JMX should be considered an extension of the Administrable concepts.
- Random interfaces don’t have to be created for service administration of simple attributes.
- With the use of StandardMBean and its two arg constructor, the provided Class can be a Jini remote interface that would allow a full ServiceUI to be plugged in at anytime.

JDK 1.6 Scripting

- Can we exploit the scripting in JDK1.6 to create generic, type-less algorithms for some of the service control and administration mechanisms?
- Is there anything to be gained from using scripting in a different Configuration implementation?
- What about other JDK1.6 features?

IDE Tools

- The IDEs are getting all kinds of power for supporting J2EE/JEE service creation and deployment.
- An IDE module for Jini service configuration would make it possible to direct the user through more explicit steps. X.509 by default for example.
- An IDE module for –dl.jar creation and/or PREFERRED.LIST creation might make some of the issues more tractable by enumerating the choices.

A Service Deployment Standard

- com.sun.jini.start
  - well known and used widely by default.
- Cheiron
  - Seven – finally it's here!
- Harvester
  - Provides some useful tools

A Service Deployment Standard (cont)

- Startnow
  - A collection of many things that I've found useful, including the PersistentJiniService class.
- WhatsItDo
  - An investigation of how to deploy Jini services which are examples, into a container. This is a container of containers for varied types of service deployment.

Discussion

Gregg Wonderly

Cyte Technologies Inc.