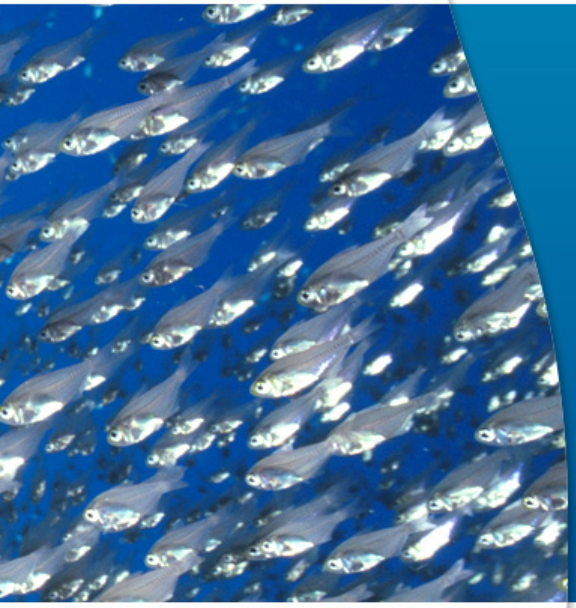
Java™ EE Connector Architecture 1.6 JSR 322

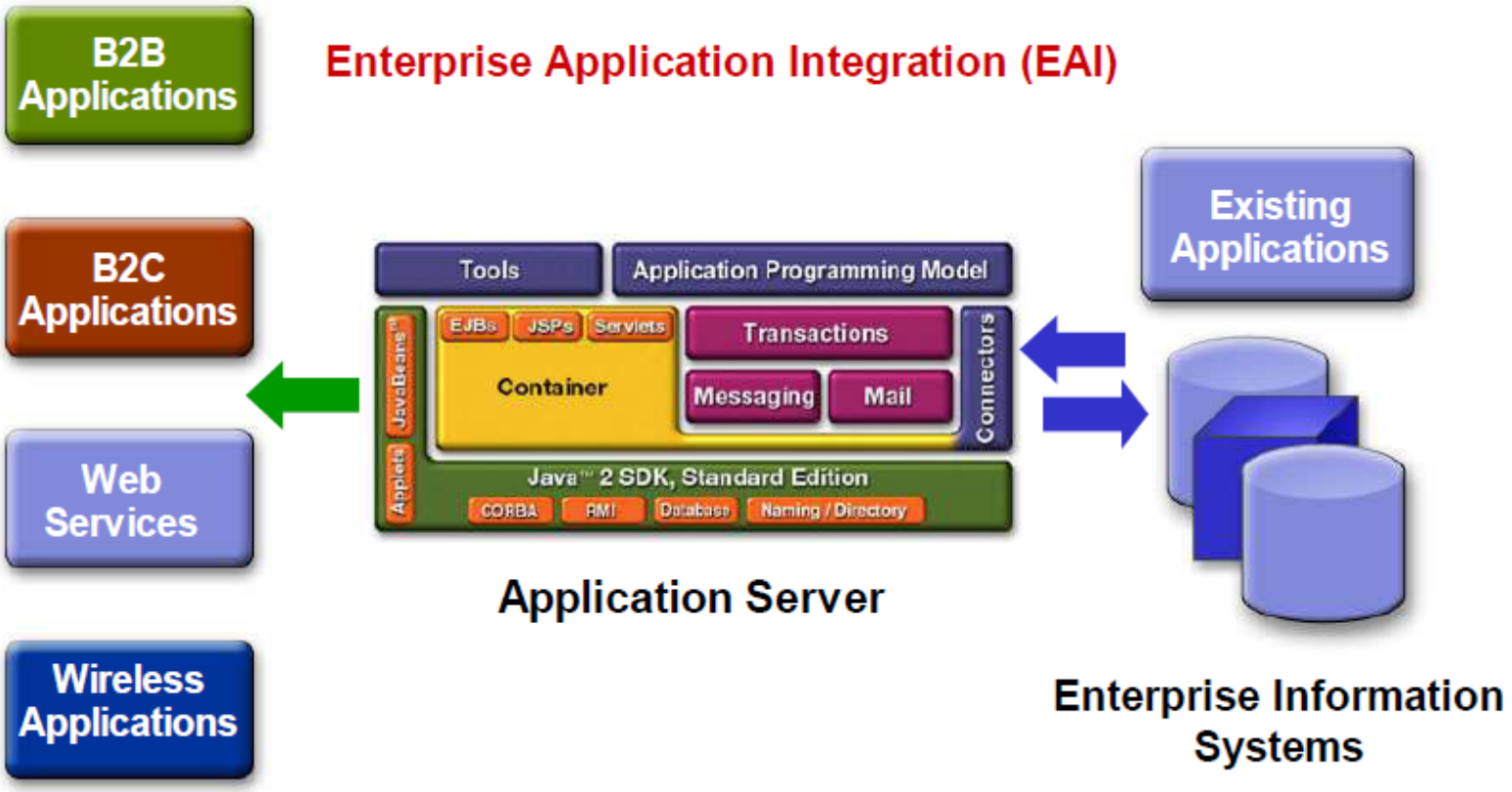
Sivakumar Thyagarajan

Sun Microsystems



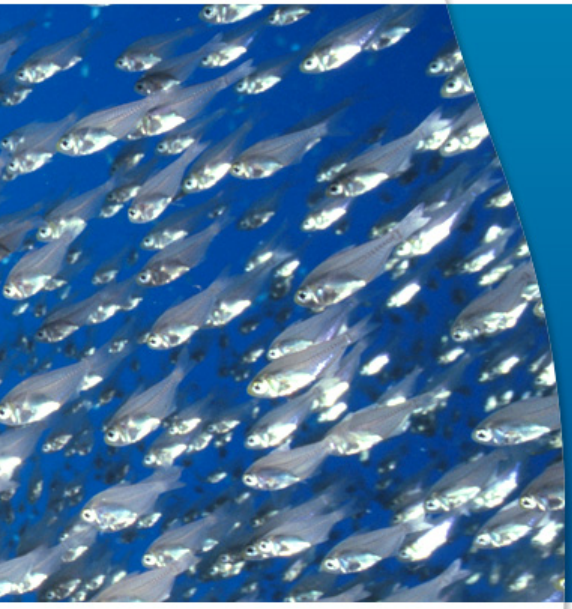
Architecture Overview

Java EE Connector Architecture
Overview



Connector Architecture
Capabilities so far in the EE platform

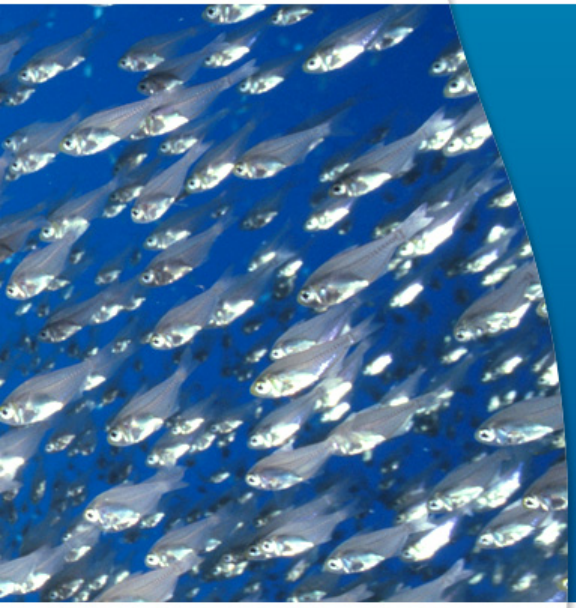
- Outbound communication
  - Mechanism for Java EE components to connect to external systems
  - Connection Management, Security Contract and Exporting Transaction and Security Context to EIS
- Inbound Communication
  - Mechanism for EIS to call Message Endpoints (MDBs), JMS provider pluggability
  - Transaction Inflow
- Lifecycle and Work Management



Connectors 1.6

Connector 1.6 Specification
Major Themes

- Driven by community and EG (JSR 322) feedback
- Themes
  - Ease of Development (EoD)
  - Generic Work Context mechanism
  - Security Context Inflow during Message Delivery and Work Submission
  - Misc. improvements to the specification



Ease of Development

Ease of Development (EoD)
Goal and approach

- Goal
  - to simplify the development of RAs for programmers who are either just starting with Connector technology or developing RAs of small to medium complexity
- Approach
  - Align with the EoD approach adopted by EJB/JPA
    - Make extensive use of Java language annotations, better defaults etc.

Ease of Development
Metadata annotations

- @Connector
  - @AuthenticationMechanism
  - @SecurityPermission
- @ConfigProperty
  - Automatic discovery of configuration properties
- @ConnectionDefinition, @ConnectionDefinitions
- @Activation
- @AdministeredObject

Ease of Development
@Connector sample usage

- Optionally provide a *ResourceAdapter* class
  - Required only if RA supports inbound, requires lifecycle callbacks, access to *BootstrapContext* etc

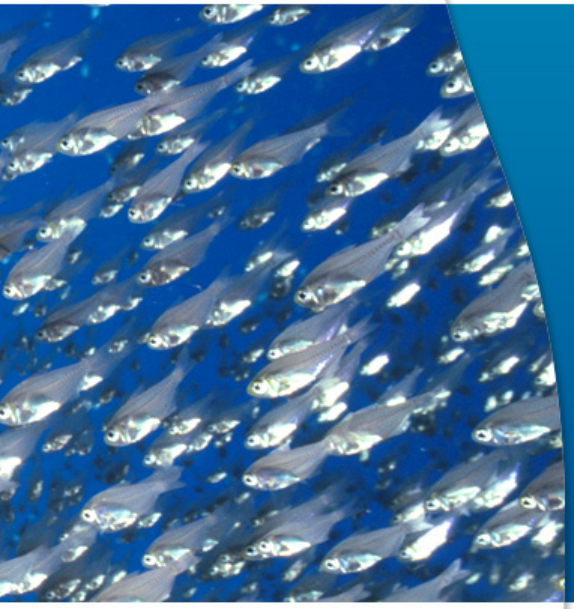
```
//A simple resource adapter that does not support
  transactions
@Connector()
public class MailResourceAdapter implements
  ResourceAdapter{
    // Define common configuration properties.
    //... other methods
}
```

Ease of Development
@ConnectionDefinition sample usage

```
//A simple ManagedConnectionFactory implementation
//that is part of a connection definition, could be
//defined as follows
@ConnectionDefinition(connectionFactory=CF.class,
    connectionFactoryImpl=CFImpl.class,
    connection=Conn.class,
    connectionImpl=ConnImpl.class)
public class ManagedConnectionFactoryImpl implements
  ManagedConnectionFactory {
    //...
}
```

Ease of Development
@Activation sample usage

```
@Activation(messageListeners={com.wombat.ra.MyMessage
  Listener.class})
public class MyActivationSpec {
    //Use of Bean Validation annotations to express
    //validation requirements
    @Size(min=5, max=5)
    private int length;
    //... other methods
}
```



Generic Work Context

Generic Work Context
Rationale

- A generic mechanism for the RA to propagate other contextual info from EIS during message delivery or Work submission
  - Examples: Security, Conversational Context, Availability/QoS etc
- Enables
  - an application server to support new message inflow and delivery schemes
  - provides a richer Contextual Work execution environment to the RA

Generic Work Context
Design Model

- RA submits a *Work* that implements *WorkContextProvider*
- Before calling *run* on the *Work* instance, *WorkManager* iterates through the *Collection* of *WorkContext*s provided by the *Work* instance
- Establishes Context based on the *WorkContext*

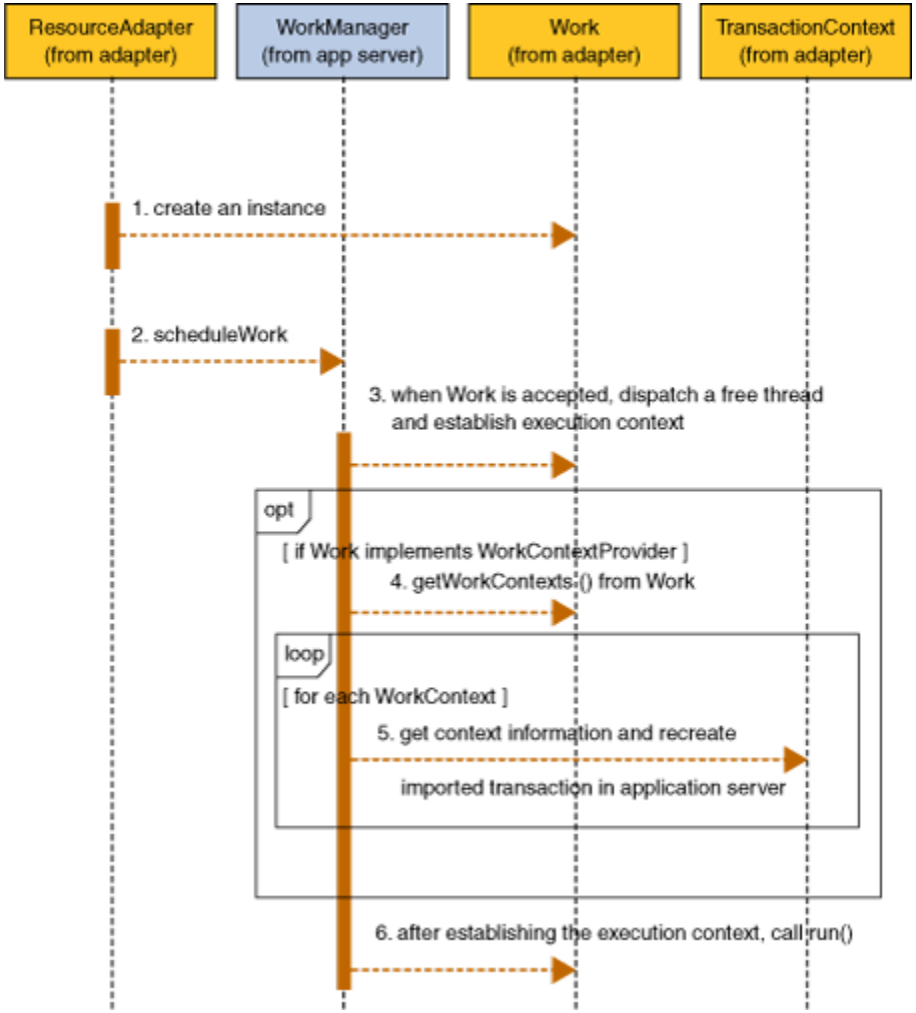
Generic Work Context

WorkContextProvider and WorkContext interface

```
package javax.resource.spi.work;
public interface WorkContextProvider {
    List<WorkContext> getWorkContexts();
}

package javax.resource.spi.work;
public interface WorkContext {
    String getName();
    String getDescription();
}
```

Generic Work Context

Context assignment sequence diagram



Generic Work Context
Features

- Compatible with the Connector 1.5 Work submission and context assignment model
- Standardized *TransactionContext*, *SecurityContext* and *HintsContext*

Generic Work Context

Standardized *WorkContexts*

```
public class TransactionContext extends ExecutionContext
  implements WorkContext {
    public TransactionContext(Xid xid) { ... }

    public TransactionContext( Xid xid, long timeout){ ... }

    public String getName(){

        return "TransactionContext";

    }

    ... other methods

}

public abstract class SecurityContext implements WorkContext {
    public String getName(){

        return "SecurityContext";

    }

    .... other SecurityContext related methods

}
```

Generic Work Context
Sample usage

- Scenario
  - A business use-case requires that the work done by the application components, during a message inflow, be automatically enlisted as part of the imported transaction .
- Solution
  - Use the new *TransactionContext* to propagate transaction information from the EIS to the *MessageEndpoint* as part of the *Work* instance performing the message delivery

Generic Work Context

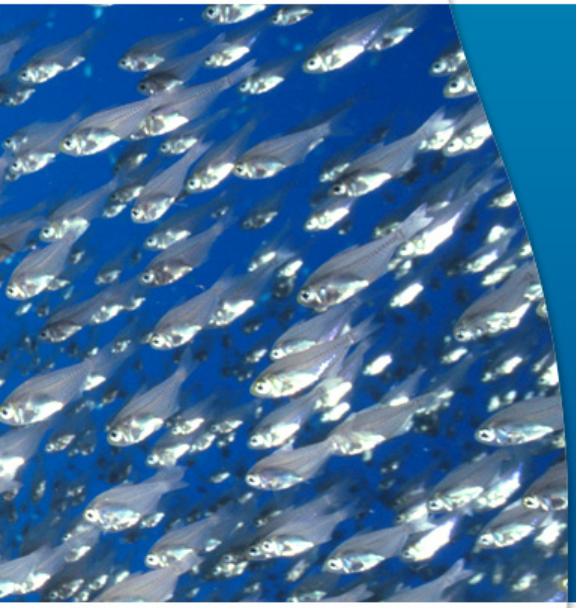
Sample usage – ResourceAdapter implementation

```
public class MyResourceAdapterImpl
          implements ResourceAdapter {
    ...
    public void start(BootstrapContext ctx) {
        bootstrapCtx = ctx;
    }
    ...
    {
        WorkManager workManager =
          myRA.bootstrapCtx.getWorkManager();
        workManager.submitWork(new MyWork());
    ...
    }
}
```

Generic Work Context

Sample usage – Work implementation

```
public class MyWork implements Work, WorkContextProvider {
    void release(){ ..}

    List<WorkContext> getWorkContexts() {

        TransactionContext txIn = new
          TransactionContext(xid);

        List<WorkContext> icList = new
          ArrayList<WorkContext>();

        icList.add(txIn);

        // Add additional WorkContexts

        return icList;

    }

    void run(){

        // Deliver message to MessageEndpoint;

    }

}
```



Security Inflow

Security Inflow
Goals

- Enable an end-to-end security model for Java EE application-EIS integration
- Support the execution of a *Work* instance in the context of an established identity.
- Support the propagation of *Principal* information from an EIS to a *MessageEndpoint* during message inflow

Security Inflow
Design Model

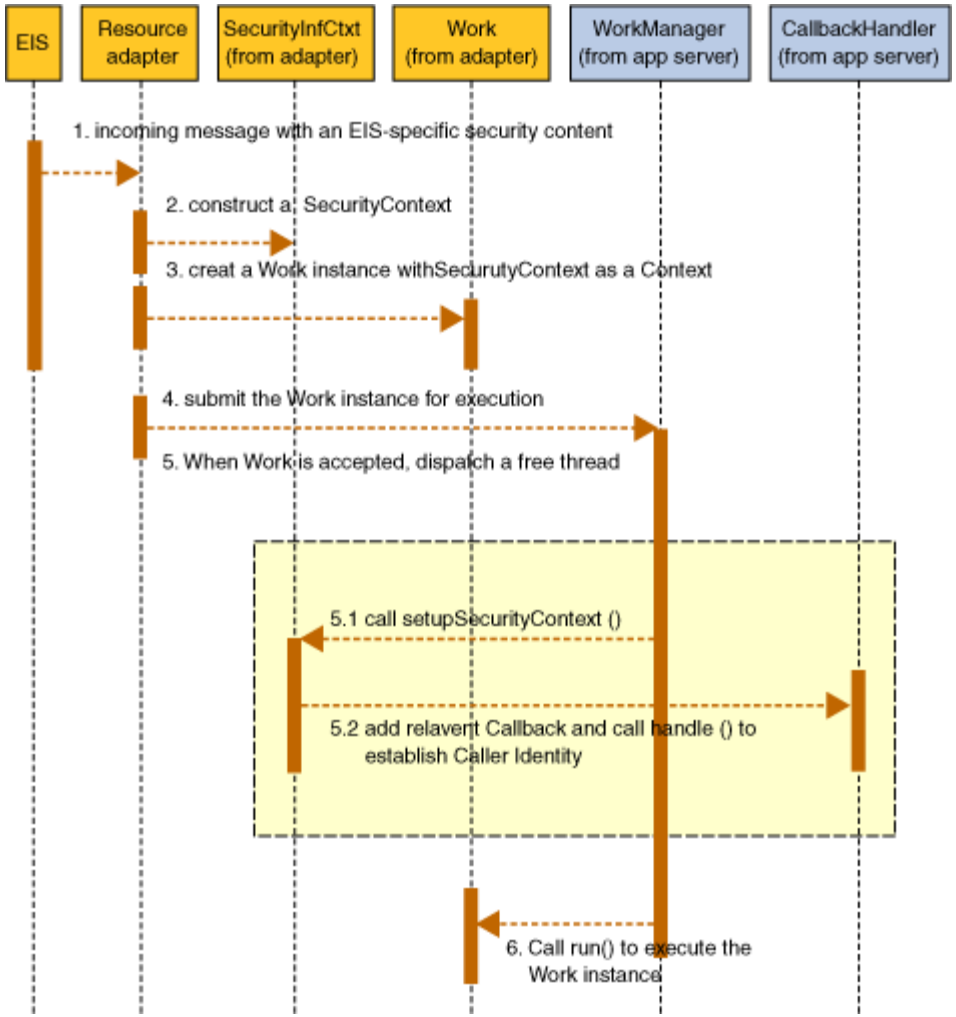
- A standard *WorkContext*, *SecurityContext* that may be provided by the RA while submitting a *Work* for execution
  - AS establishes security context for the *MessageEndpoint*/*Work* during context assignment
- Leverage the work done in JSR 196: Java Authentication Service Provider Interface for Containers
  - RA uses the various JSR-196 Callbacks to establish caller identity

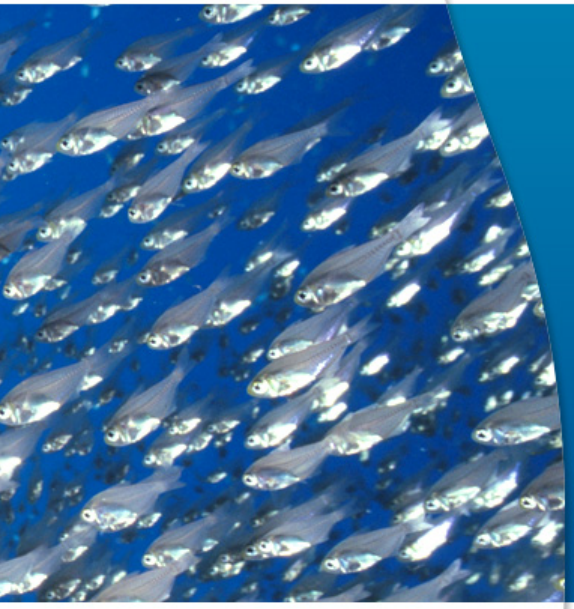
Security Inflow
SecurityContext abstract class

```
package javax.resource.spi.work;

import javax.security.auth.Subject;

import javax.security.auth.callback.CallbackHandler;


public abstract class SecurityContext
                    implements WorkContext {

    //Establish caller identity through the use of JSR196
      Callbacks

    public abstract void setupSecurityContext(

        CallbackHandler handler,

        Subject executionSubject,

        Subject serviceSubject);

}
```

Security Inflow

Security Context establishment – sequence diagram





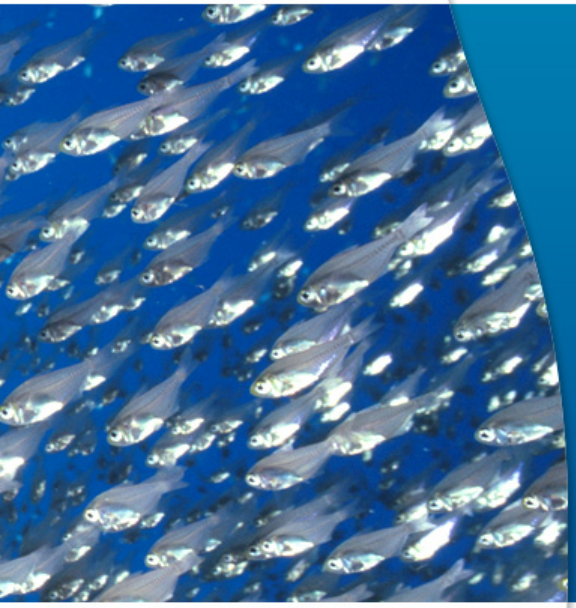
Miscellaneous Improvements

Miscellaneous improvements

- Transaction Management
  - Specification of Transaction Support Level at Runtime
- Work Management
  - Distributed Work Processing
  - HintsContext
- Message Inflow
  - createMessageEndpoint timeouts
  - Retryable exceptions

Miscellaneous improvements

- Bean Validation integration (JSR 303)
- Standalone Connector Environment
- Configuration Property processing
  - range specification through JSR 303
  - dynamic reconfiguration and confidential params support
- Classloading requirements
  - clarified scenarios where deployed standalone RARs must be made visible to applications



Resources & Summary

Resources – learn more & get started

- JSR 322 page http://jcp.org/en/jsr/detail?id=322
- Reference implementation at http://GlassFish.org
  - Download **GlassFish v3** – Java EE 6 release
  - Try Connector 1.6 samples and build 1.6 RAs
- Blogs
  - Siva: http://blogs.sun.com/sivakumart
  - Binod: http://weblogs.java.net/blog/binod/
  - Jagadish: http://blogs.sun.com/JagadishPrasath/

Resources

Java EE 6 and GlassFish v3

- Java EE 6 Home
  java.sun.com/javaee
- Java EE 6 Downloads
  java.sun.com/javaee/downloads
- Upcoming Training
  java.sun.com/javaee/support/training

Java EE 6

- Sun GlassFish Enterprise Server v3 Home
  www.sun.com/glassfishv3
- Community Page
  glassfish.org
- The Aquarium Blog
  blogs.sun.com/theaquarium
- White Papers/Webinars
  sun.com/glassfish/resources

GlassFish

Summary

- Part of Java EE 6 and reference implementation available through GlassFish v3
- Simplifies resource adapter development
- Adds powerful new capabilities to assign contexts during Work execution and Message delivery
- Enhances enterprise application integration capabilities in Java EE 6

Q&A and Thank You!

Sivakumar Thyagarajan

sivakumart@sun.com

Connector Specification comments

javaee-connector-comments@sun.com