Avicon
781.464.3000 • info@avicon.com • www.avicon.com

Avicon
Suite 140
404 Wyman Street
Waltham, MA
02451-1264

JCA – An Emerging Integration Architecture

Vijay Sundhar
Marc Linster
Steven Moy
Avicon, Inc.

Introduction

Integration inside the enterprise and integration with the extended enterprise are key components of every e-business strategy. Increasing pressure to cut cost, reduce cycle times, connect with outsourced service providers, manage heterogeneous environments resulting from acquisitions, increase the speed of information flow and provide decision makers with more accurate, more complete and more timely information is bringing the need for efficient and cost-effective system integration to the forefront for many CIOs and IT Directors.

Traditionally, integration with Enterprise Information Systems (EIS), such as ERP, CRM, MRP or Datawarehouses has been achieved through custom adapters built or bought specifically for the underlying EIS architecture, the EIS' data model, the processes it supports and the proprietary integration platform chosen for the enterprise. This resulted in very costly and inflexible integration architectures that locked IT organizations into proprietary integration platforms and limited the availability of EIS adapters to those provided by the integration platform provider or adapters custom-built for that integration platform.

"The maturity and acceptance of J2EE coupled with the release of JCA 1.5 with key integration features such as Inbound Communication/Message Inflow, EJB Invocation, Work Management, Adapter life cycle management and enhanced connection, security and transaction model provides enterprises a cost-effective standard based integration solution."

JCA (Java Connector Architecture) provides a standard framework for EIS integration that defines the interfaces and key building blocks for an adapter independently of the underlying EIS or the proprietary enterprise integration platform. JCA provides the foundation for reuse of EIS adapters across integration platforms, both for custom-built adapters and for commercially available adapters from specialized vendors.

The most recent release of JCA, Release 1.5, brings JCA to the enterprise level. With JCA 1.5 the new standard is getting sufficient support and commitment from major integration platform providers to make it an important ingredient for every future enterprise integration strategy. This article describes the JCA architecture and the key features of the latest release in an enterprise integration context.

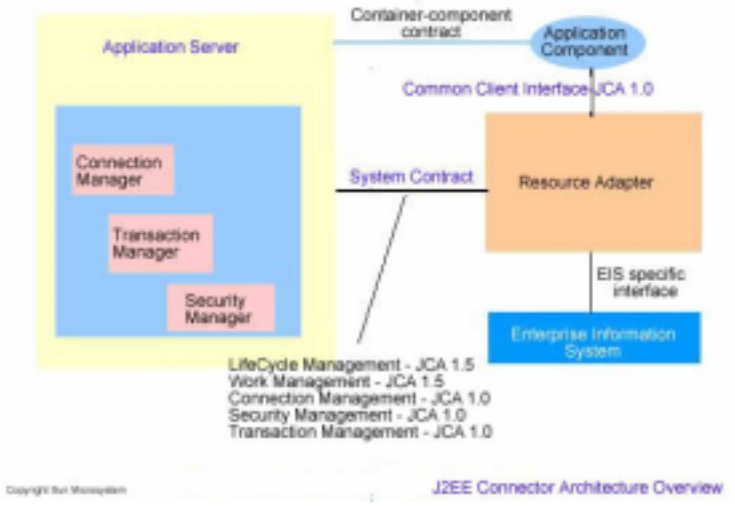
J2EE Integration Components

To provide an industry standard platform to build robust Java applications and integration solutions, Sun released the J2EE architecture. Some of the most widely used components of the J2EE standard for enterprise integration are:

- Java Server Pages & Java Servlets (JSP/Servlet) for presentation
- Enterprise Java Bean (EJB) for Business Logic
- Java Messaging Service (JMS) for messaging
- Java Transaction API (JTA) for transactions
- Java Authorization and Authentication Service (JAAS) for security
- Java Naming and Directory Service (JNDI) for service lookup
- Java Database Connectivity (JDBC) for accessing relational databases
- Java API for JAXB/JAXM/JAXP

Java Connector Architecture Overview

Although core application services like transaction management, security, and connection pooling are provided by Application Servers, no standard EIS integration mechanism exists. To fill this void, Sun has released the J2EE Connector Architecture Specification (JCA). JCA is a enterprise connectivity standard for creating and deploying Resource Adapters in any J2EE complaint application server such as Weblogic and JBoss. Currently, most application servers support JCA 1.0 release, as JCA 1.5 is not released yet. This enables java applications to connect and interact with any Enterprise Information System (EIS) such as SAP and Oracle in an industry standard way through a deployable unit called Resource Adapter. Resource Adapter, a Java Connector Architecture deployable unit, provides a plug between the enterprise Java environment and the EIS. A resource adapter is a set of classes that expose and implement:



- System Contract Interface
- Common Client Interface (CCI)
- Classes that integrate with the native interface(s) of the EIS (conceptually analogous to an EIS driver).

System Contract

The System Contract, as defined by the JCA specification, provides a set of standard interfaces that allow the Application Server to provide common system services to the Resource Adapters. This eliminates the need to implement low-level services at the Resource Adapter level and provides a common mechanism for the EIS to participate in the business transaction.

The following are the JCA 1.0 system level services provided by the application server to the resource adapter:

- Connection Management
- Security Management
- Transaction Management

The following are the JCA 1.5 system level services provided by the application server to the resource adapter:

- Life Cycle Management

  The life cycle management is a mechanism through which the application server manages the life cycle of a resource adapter. It is a contract defined as a set of interfaces that must be supported by both the application server and the resource adapter. This mechanism provides a facility to bootstrap a resource adapter during application server startup and exposes some of the application server services to the resource adapter. This also provides an application server to notify a resource adapter when it is undeployed or during application server shutdown. The key aspects of this feature is as follows:
  - Resource adapter initialization
  - Easy and uniform access to application server services for resource allocation and management
  - Resource adapter cleanup

- Work Management
  The Work Management contract provides a mechanism for a resource adapter to submit work to application server for execution. The reason for this is to prevent resource adapters from creating and managing threads directly in order to execute work. Also, some application servers do not allow threads to be created and managed due to security policy setting and for efficiency reasons. Some of the benefits of this model are as follows:
  - Isolates resources adapters from managing system resources
  - Provides a flexible work execution model for resource adapter to do work
  - Increases portability across many different application servers
  - Better management of system resources
  - Allows application servers better control over its system components

JCA release 1.0 incorporated Connection, Security and Transaction Management to support outbound communication to the EIS. JCA release 1.5 incorporates the other two system level services life cycle management and work management to support inbound communication from the EIS.

The system contract is a binding between the application server and the resource adapter to coordinate all of the above system level services. The application server and the resource adapter must implement their respective system contract interfaces as defined in the specification for proper deployment, execution and management of resource adapters.

Common Client Interface (CCI)

The Common Client Interface, as defined by the JCA specification, provides a common Application Programming Interface (API) for Java clients to communicate to heterogeneous EISs. The CCI allows the Java application developer to be insulated from the implementation specifics of the EIS integration, which are contained in the Resource Adapter, and thereby focus on the development of business logic. This mechanism provides a consistent programming model for any client to access any EIS and process results in an EIS independent format.

One of the key issues limiting data interoperability today is that of incompatible metadata. Metadata can be defined as information about data, or simply data about data. The common client interface at this point does not provide a generic mechanism for handling metadata, which is one of the key enterprise integration requirements. The primary focus of JCA 2.0 is to enhance CCI to provide generic metadata interface for handling both outbound and inbound data. This will leverage some of the work done in the Java Metadata Interface specification (JMI-JSR-000040).

Interaction Model Overview

The following are the two types of application interaction model supported by JCA:

- Outbound Communication
  The outbound communication provides a mechanism for an application component such as Session, Entity or Message Driven Bean (MDB) to send messages through a resource adapter to an EIS.
- Inbound Communication
  Inbound Communication allows resource adapters to handle events flowing in from the EIS to the application residing in an application server.

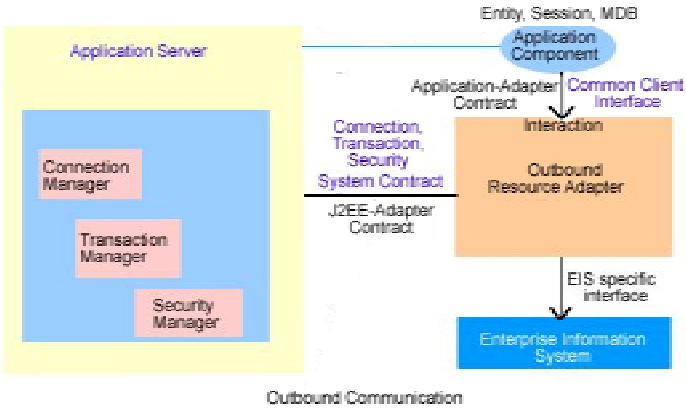
Outbound Communication

The outbound communication model provides a mechanism for an application component such as Session, Entity or MDB to send messages through a resource adapter to an EIS. This completely isolated the understanding from the application component of the message format, semantics and communication protocol required for dispatching the actual message.

Architecture

The outbound communication is a generic contract between the Application Server, Resource Adapter and the Application Component for handling outbound messages. The following well-defined contracts ties the application server, resource adapter and the application component:

- System Contract
  This interface is for the application server to communicate with the resource adapter to coordinate Connection, Security and Transaction Management
- Common Client Interface (CCI)
  This interface is for the application component to communicate with the resource adapter for interacting with the EIS



When an application component needs to send an outbound message to the EIS, it first obtains a connection reference through the JNDI interface. This connection handle is defined part of the common client interface. Using the connection handle, the application component can now create many interactions. The interaction reference is used to send outbound message to the EIS. Customization can be done at the interaction level to further process the outbound message before sending it to the EIS. During EIS interaction, the application server and the resource adapter communicate to coordinate security and transaction integrity.

Inbound Communication

Inbound Communication allows resource adapters to handle events flowing in from the EIS to the application residing in an application server. Prior to JCA 1.5, a resource adapter can only handle outbound events to the EIS. With this feature, the resource adapters can handle bi-directional flow of business events. The inclusion of this feature as part of JCA 1.5 is a key milestone that positions JCA as viable enterprise architecture for solving complex integration problems. Prior to this model, the J2EE components could send and receive messages but through special contract custom built specific to a particular message provider. This contract has to be built for every different message type.

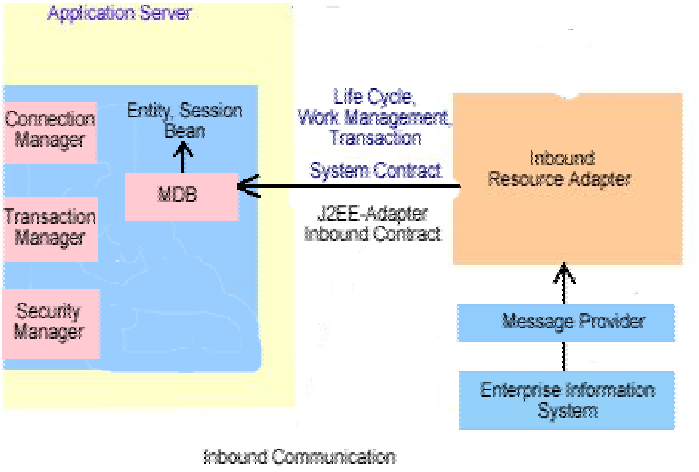
The following are the key goals of this model:

- Provide a standard mechanism to dispatch incoming messages from EIS to the application server components such as Session, Entity bean or Message Driven Bean (Message Endpoint or Consumer)
- Provide a standard mechanism to plug in wide variety of Message Providers to deliver messages to the application server
- Provide a standard mechanism to propagate incoming transaction and security context to the application server

Architecture

The inbound communication is a generic contract between a resource adapter and an application server for handling inbound messages from various message providers. A message provider is one that is a source for incoming message from EIS. In a B2Bi environment, a message provider can be JMS, HTTP, FTP, SMTP, CORBA and File System etc. This generic contract enables different message providers to be plugged into an application server independent of the message style,

message semantics and infrastructure. This allows resource adapters to synchronously or asynchronously delivers incoming messages to the message endpoints residing in the application server. The Inbound resource adapter uses the work management facility to create work instances to read, process and dispatch incoming message from EIS to the appropriate message endpoint (application component) in the application server.



The architecture supports both synchronous and asynchronous delivery of inbound messages. The synchronous delivery takes place through EJB invocation. In this case, the resource adapter uses the MDB as a generic dispatcher for session and entity bean. The invocation is achieved through the use of bean client view via a message driven bean. The bean client view and to access the session and entity bean is defined in the EJB specification. In the case of asynchronous delivery, the message is delivered directly through the invocation of the onMessage method of the Message Driven Bean.

The key aspects of this generic inbound contract is as follows:

- It provides a mechanism to ties the system level features such as connection, security and transaction into a standard loosely couple model.
- The loosely coupled model enables J2EE components to consume messages without the knowledge the message provider.
- Enables different message providers to be plugged into an application server through a resource adapter.

Requirements for Inbound Communication

In order to handle inbound events from the EIS, the following are the key requirements:

- Setup Message Provider for message delivery from EIS
- Resource adapter provided either by the Message Provider or the third party must list all the endpoint message listener types such as Session, Entity or MDB and configuration properties for each message listener for endpoint activation.
- Support for inbound contract by both the application server and resource adapter runtime environment
- Endpoint deployment to deploy endpoint application on an application server

Application Interaction Summary

The table below gives a brief summary of the following:

- List of all system contract supported in both JCA 1.0 or JCA 1.5
- Shows the key actors involved in both outbound and inbound message flow scenarios
- Shows the system contract requirements for both outbound and inbound interaction model

System Contract	Version Supported	Outbound Communication			Inbound Communication	
		Application Component (Client)	J2EE Server	Adapter	J2EE Server	Adapter
Connection Management	1.0		X	X	-	-
Security Management	1.0		-	X	-	X
Transaction Management	1.0		-	X	-	X
Lifecycle Management	1.5		X	X	X	X
Work management	1.5		X	X	X	X
Common Client Interface	1.0	X	-	X	-	-

X – Must implement System Contract interfaces as defined in the specification for supporting the specific feature listed

Deployment Model

The runtime resource adapter files are packaged in a Resource Adapter Archive (RAR), which is similar to the JAR file format except with a RAR extension. JCA provides two types of deployment models for deploying this file: one for deploying in a J2EE environment (Managed) and the other for deploying in a non-J2EE environment (Non-Managed).

- Managed scenario
  In this scenario, the adapter is deployed within a J2EE complaint Application server. This scenario enables the integration of EISs with any enterprise Java application through a set of secure, scalable and transaction mechanism as defined by the JCA architecture.
- Non-Managed scenario
  In this scenario, the adapter is deployed in a non-J2EE environment. In this scenario the application directly uses the resource adapter to access the EIS.

JCA Adoption

Today, the primary adoption of JCA is by the application server vendors such as BEA, IBM, IPlanet and others due to the following reasons:

- Sun mandated all J2EE vendors to support JCA for J2EE 1.2 certification
- J2EE will be the preferred JCA runtime platform for enterprise integration due to the encapsulation of all related services in a standard way
- Most application server vendors will support some form of Adapter Development Kit for adapter development

The next in the adoption list are adapter vendors such as Merant and IWay Software who provide out of the box adapters for all major enterprise systems. These out of the box adapters can be deployed on any of the application servers that support JCA standard.

The last in the list will be the major integration vendors such as TIBCO, Vitria, Webmethods, SeeBeyond and others. Most of these vendors have not supported JCA 1.0 due to the following reasons:

- Lack of support for handling inbound messages from EIS and metadata in JCA 1.0
- Proprietary integration platform requiring migration to J2EE for hosting JCA adapters

The release of JCA 1.5 is expected to be around during the second quarter of 2003 as part of J2EE 1.4. J2EE 1.4 along with JCA 1.5, WebServices and enhancements to EJB, JSP and Servlet container will provide a robust industry standard platform for deploying and managing application services. Most application server vendors will immediately support JCA 1.5 to move into the integration space. On the other hand JCA 1.5 will force existing integration players such as TIBCO, Vitria and Webmethods to use J2EE as

their integration platform. JCA tools and solutions will emerge during the later part of the year providing support for rapid application development of adapters.

JCA is evolving and with continued contribution from key application server, integration broker vendors and from the J2EE community, it is poised to emerge as a key integration standard to solve major integration issues. It is also the only J2EE standard that completely encapsulates key services such as connection, security, transaction and client interaction to communicate within and across the extended enterprise.

JCA – The Bottom Line

In short, JCA 1.5 provides a comprehensive integration development framework to create an extended enterprise in a cost-effective manner. It is very important to note that the connector architecture is not limited to connectivity to legacy systems but, as an architecture to invoke both inter and intra-enterprise services. Most importantly, JCA is designed to deliver the following business values:

- Ease of Integration
  The JCA open architecture connects different enterprise information systems across internal and external heterogeneous environments. JCA enables consistency, plug-compatibility, and repetition.
- Industry Standardization
  As part of the J2EE standard, JCA framework provides maximum interoperability with other major industry vendors' frameworks, such as WebLogic, TIBCO, Vitria, MQSeries, SAP, hence protecting and leveraging existing IT investments.
- Enterprise architecture flexibility
  By using JCA as the connectors, IT decision makers can phase-out legacy systems or adapt new technology with minimal impact to both the enterprise architecture and business
- Cost reduction
  With strong industry adoption, JCA will provide more predictable, maintainable and consistent results, eliminating costly custom integration efforts.

IT decision makers should adapt JCA capabilities as part of their IT enterprise tool sets by taking small, strategic steps to establish "quick wins". More importantly, IT decision makers can leverage key learning's from these smaller quick wins to establish instant credibility, and then gradually roll out larger initiatives to the rest of the organization.

Reference

Java Connector Architecture 1.0 & 1.5 specification Proposed Final Draft – Sun Microsystems
J2EE Connector Architecture and Enterprise Application Integration – Rahul Sharma, Beth Stearns & Tong Ng

