



WebSphere software

Integrating WebSphere Application Server and CICS using the J2EE Connector Architecture.

*By Phil Wakelin, IBM Software Group
and Nigel Williams, IBM Design Center
for On Demand Business*

Contents
2 Introduction
3 Understanding CICS Transaction Gateway
5 JCA
8 Application development
10 Using the JCA with different CICS Transaction Gateway topologies
11 Topology 1: WebSphere Application Server and CICS Transaction Gateway deployed on distributed platforms
14 Topology 2: Remote Gateway daemon on z/OS system
17 Topology 3: Deploying both WebSphere Application Server and CICS Transaction Gateway on zSeries servers
21 Summary
23 For more information

Introduction

Much of the world's data is processed on mainframes, using the qualities of service of proven transaction servers such as IBM CICS® Transaction Server for z/OS. Delivering access to new and existing CICS applications using standards-based interfaces is becoming a necessity as your organization moves toward becoming an On Demand Business. IBM CICS Transaction Gateway software has been proven over many years to deliver high-performing, security-rich and scalable Java™ 2 Platform, Enterprise Edition (J2EE) standards-based access to CICS applications. Implementing CICS Transaction Gateway software requires minimal changes to CICS systems and usually no changes to other existing CICS applications, enabling rapid exploitation of existing enterprise applications as part of an enterprise-class service oriented architecture (SOA).

Using the J2EE Connector Architecture to connect to CICS

A J2EE application can connect to CICS Transaction Server by using the J2EE Connector Architecture (JCA), which defines a standard architecture for connecting the J2EE platform to heterogeneous enterprise information systems (EISs), such as CICS Transaction Server. This white paper explains how using the JCA can provide benefits in terms of both quicker and easier application development and optimal qualities of service in the middleware runtime environment. The JCA can be used by a number of IBM Rational®, IBM Tivoli® and IBM WebSphere® products to assist application developers to model, assemble, deploy and manage composite CICS and WebSphere applications that communicate through the JCA. These capabilities enable application developers to rethink and reuse their core assets in SOAs instead of duplicating assets or adopting a rip-and-replace approach, both of which can be costly, risky and time-consuming. Another advantage for application developers is that the underlying infrastructure of the JCA automatically manages the connection pooling, transactional scope and security qualities of the composite applications, so that these capabilities do not have to be individually coded into each application – enabling application developers to concentrate on the development of business logic rather than on quality-of-service provisioning. The latest version of CICS Transaction Gateway provides support for the XA transaction protocol, enabling two-phase-commit transactional integrity between J2EE applications deployed in WebSphere Application Server on any supported platform and CICS applications running on the IBM z/OS® operating system.

You can use JCA in integrated On Demand Business solutions to combine the strengths of WebSphere Application Server and CICS Transaction Server.

JCA, along with other J2EE standard services such as Java Message Service (JMS) and Java Database Connectivity (JDBC), is a tightly coupled connectivity method. IBM CICS Transaction Server for z/OS, Version 3.1 provides the ability to produce and consume Web services natively in CICS Transaction Server. When deciding which connectivity style to use, you need to determine whether you want to use existing application interfaces, or create new application interfaces. You also need to decide whether you consider speed to market or flexibility as more important. And you need to choose whether your primary Web services deployment platform should be CICS Transaction Server or IBM WebSphere Application Server software. More often than not, these decisions are going to depend on your business requirements – and you will probably end up implementing a combination of both.

Because all of these connectivity methods are standards-based, and as a result, provide a high degree of flexibility, you should also take advantage of an appropriate set of complementary technologies to address a range of business problems. Tightly coupled direct connections and loosely coupled Web services can coexist to fully exploit the agility of an On Demand Business environment, and together they enable you to integrate all of your CICS assets in an enterprise-class SOA.¹

Understanding CICS Transaction Gateway

CICS Transaction Gateway is a set of client and server software components that enable a Java application to invoke services in an attached CICS server. The application can be a servlet, an enterprise bean or any other Java program. This paper focuses on the use of Java servlets and enterprise beans running in WebSphere Application Server.

CICS Transaction Gateway provides support for the JCA on a wide range of platforms and for a variety of deployment options.

Deployment options

CICS Transaction Gateway, Version 6.0 is the most recent version and is currently available on the following platforms: z/OS, Linux® on IBM @server® zSeries®, IBM AIX®, HP-UX, Microsoft® Windows®, Linux on Intel®, Linux on IBM POWER™ and Sun Solaris operating environments. Connectivity is provided on these platforms from all supported WebSphere Application Server environments to all supported CICS servers.² Also, CICS Transaction Gateway, Version 6.1, which is available only on the z/OS platform, provides unique transactional qualities of service through its support for the XA transaction protocol.

The two main programming interfaces supported by CICS Transaction Gateway are the external call interface (ECI) and external presentation interface (EPI). The ECI provides a call interface to communications area (COMMAREA)-based CICS programs, whereas the EPI provides a programming interface to invoke 3270-based programs.³

The major components that make up CICS Transaction Gateway are:

- Gateway daemon
A long-running process that functions as a server to network-attached Java client applications by listening on a specified TCP/IP port. CICS Transaction Gateway supports both native TCP/IP and Secure Sockets Layer (SSL) connections.
- Client daemon
An integral part of CICS Transaction Gateway on all distributed platforms. It provides CICS client-server connectivity using the same technology as IBM CICS Universal Client software.
- Java class library
Java classes used by the application to invoke the services in an attached CICS server.

CICS Transaction Gateway offers several different topologies to choose from. For example, when using WebSphere Application Server, connections can be local (CICS Transaction Gateway communication is invoked through the Java Native Interface [JNI]) or remote connections can be made to the Gateway daemon. The Gateway daemon can be running on the same machine as WebSphere Application Server or on another machine. The diverse range of supported environments and the different deployment options can be classified into three distinct topologies. This white paper outlines them and describes the principal differences among them.

JCA

JCA⁴ enables an EIS vendor to provide a standard resource adapter for its EIS. A resource adapter is a middle-tier connector that permits the Java application to connect to the EIS. JCA, Version 1.5 defines a number of components that make up this architecture, shown in Figure 1.

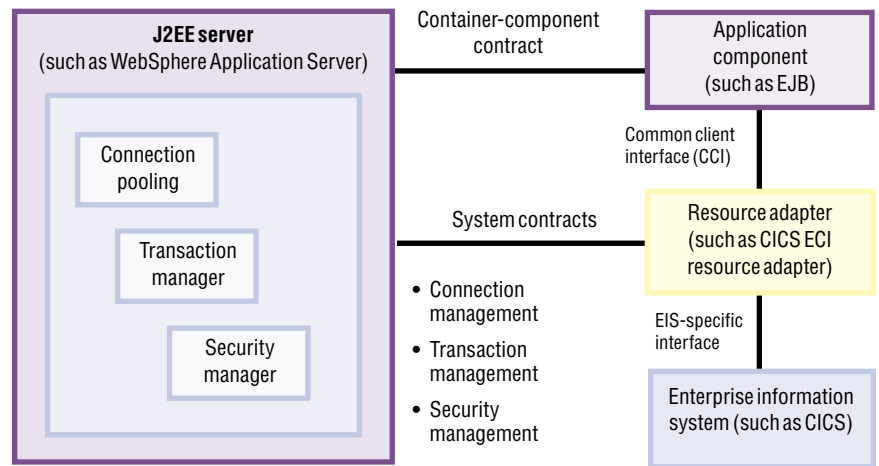


Figure 1. The JCA component structure

Common Client Interface

The Common Client Interface (CCI) defines a common application programming model for interacting with resource adapters and is independent of any specific EIS. Of course, this doesn't mean developers can write exactly the same code to access one EIS (for example, CICS Transaction Server) as they write to access another EIS (for example, IBM IMS™ systems). However, the generic CCI classes (such as connection or interaction) are the same in that they are independent of the EIS, whereas specific EIS classes (such as ECICConnectionSpec) are used to cater to the unique properties of each EIS. For example, the parameters used to call a CICS program are different compared with those used to invoke an IMS transaction, but the programming model is the same – independent of the EIS. As a result, you can help increase developer productivity when developing applications to communicate with multiple EISs. The CCI programming interface is similar to other J2EE interfaces, such as the JDBC interface.

Resource adapters

CICS Transaction Gateway, Version 6 provides three separate resource adapters, two for ECI and one for EPI.

- ECI resource adapter

This adapter is required to make calls to CICS COMMAREA-based programs. Because ECI is the most commonly used access mechanism, and because it is designed for optimal component reuse, the ECI resource adapter is the focus of this paper. It is supported with CICS Transaction Gateway on all platforms and provides one-phase-commit transactional coordination between WebSphere Application Server and CICS Transaction Server, and two-phase-commit support when used with WebSphere Application Server for z/OS. The JCA, Version 1.0 resource adapter is available for use with WebSphere Application Server, Version 5.0 and Version 5.1, and the JCA, Version 1.5 resource adapter is available for use with WebSphere Application Server, Version 6.0.

- ECI XA resource adapter

This adapter is required if the ECI calls need to be included as part of a global two-phase-commit transaction coordinated by WebSphere Application Server. It requires the use of CICS Transaction Gateway, Version 6.1 for z/OS. A JCA, Version 1.5 resource adapter is available for use with WebSphere Application Server, Version 6.0.

- EPI resource adapter

This adapter is required to make calls to CICS 3270-based programs. It provides a record-based interface to terminal-oriented CICS applications. The JCA, Version 1.0 resource adapter is available for use with WebSphere Application Server, Version 5.0 and Version 5.1, and the JCA, Version 1.5 resource adapter is available for use with WebSphere Application Server, Version 6.0. The EPI resource adapter is supported only in conjunction with CICS Transaction Gateway for Multiplatforms.

System contracts

JCA defines a standard set of system-level contracts between a J2EE application server and a resource adapter. These system-level contracts define the scope of the managed environment that the J2EE application server provides for JCA components. The standard contracts include:

- *A connection-management contract that provides a consistent application programming model for connection acquisition and enables a J2EE application server to pool connections to a back-end EIS. This leads to a scalable and efficient environment that can support a large number of components requiring access to an EIS.*
- *A transaction-management contract that defines the scope of transactional integration between the J2EE application server and an EIS that supports transactional access. This contract supports two interfaces, XAResource (for XA transactions) and LocalTransaction for one-phase-commit transactions. XA transactional support is used by JCA to support one-phase-commit coordination of distributed transactions across multiple resource managers in different EISs. LocalTransaction refers to transactions that are managed internally to a resource manager (such as CICS Transaction Server) without the involvement of an external transaction manager. Within WebSphere Application Server, these transactions are known as Resource Manager Local Transactions, and can be coordinated using a one-phase-commit protocol.*

The management of connections, transactions and security is controlled within WebSphere Application Server through a combination of both configuration parameters and application-deployment descriptors.

- *A security-management contract that enables security-rich access to an EIS. This contract provides support for a highly secure application environment, which helps reduce security threats to the EIS and helps protect valuable information resources managed by the EIS. Both container-managed signon (in which the J2EE application server is responsible for flowing security context to the EIS) and component-managed signon (in which the application is responsible for flowing security context to the EIS) are supported.*

These system contracts are transparent to the application developer, which means they do not have to implement these contracts themselves. In a managed environment such as WebSphere Application Server, system contracts are agreements between the J2EE application server and the resource adapter about how to manage connections, transactions and security. It is these system contracts that make the JCA such a robust solution for integrating CICS applications with J2EE applications running in WebSphere Application Server.

Application development

The following list shows the basic outline of the calls for using the CCI with CICS ECI resource adapters:

1. *Look up a `ConnectionFactory` for the ECI resource adapter.*
2. *Create a `Connection` object using this `ConnectionFactory`. A `Connection` is a handle to the underlying network connection to the EIS. Specific connection properties, such as a user name and password, can be passed using an `ECIConnectionSpec` object.*
3. *Create an `Interaction` from the `Connection`. Specific interaction properties such as the transaction identifier can be passed using an `ECIInteractionSpec` object. The call to the EIS is initiated by invoking the `execute()` method on the interaction, passing data as input and output records.*
4. *After the required interactions have been processed, the interaction and connection should be closed.*

When the CICS ECI resource adapter is used by a J2EE application, WebSphere Application Server can control the management of connections, transactions and security—enabling end-to-end On Demand Business solutions.

Table 1 provides a brief description of the main generic CCI classes (as they apply to CICS systems) and the specific ECI resource adapter classes.

Table 1. Generic classes

ConnectionFactory	Creates a <code>Connection</code> object from supplied connection settings. The connection settings are defined using the WebSphere Application Server administrative console, and are then looked up by the application to create the <code>ConnectionFactory</code> .
Connection	A handle to a connection managed by the J2EE application server.
Interaction	A specific interaction with CICS Transaction Server that occurs over an established connection.
Record	A generic wrapper for the data passed within the CICS COMMAREA.

Specific ECI resource adapter classes

ECIConnectionSpec	CICS-specific connection information, such as user ID and password, which can be used to override values set in the <code>ConnectionFactory</code> .
ECIInteractionSpec	CICS-specific interaction information, such as the mirror-transaction identifier and program name.

Tool-generated development

The CCI is targeted primarily at application-development tools and enterprise application-integration frameworks. J2EE Connector Tools, an optional feature in Rational Application Developer and IBM WebSphere Developer for zSeries software, provides a development environment that contains a range of JCA resource adapters, including the CICS ECI and EPI resource adapters. The J2EE Connector Tools feature is targeted at development projects requiring integration with back-end systems and provides support for a variety of client types, including Web services clients. It also allows existing software components, such as CICS programs, to be wrapped as services, which can communicate with other applications through a common interface. A CICS ECI service can be generated from an existing CICS COMMAREA-based program using a wizard, and the service can then be deployed in WebSphere Application Server.

Rational Application Developer provides J2EE developers with a next-generation, integrated development environment for building, testing and deploying J2EE applications and Web services. Existing CICS programs can be quickly and simply encapsulated as reusable Java beans through a sophisticated set of wizards, known as J2C⁵ wizards. These enable integration of existing components into an overall SOA. You can also use Rational Application Developer wizards to migrate existing components created with the WebSphere Studio Application Developer Integration Edition tooling, into more reusable Java bean components.

Several factors should be considered when choosing between a hand-coded development process or a toolled solution like IBM Rational Application Developer. These include:

- *The availability of Java programming skills.*
- *Time-to-market requirements. Tool-generated applications might take less time to develop.*
- *The complexity of the COMMAREA or 3270 screens. Hand-coded solutions require the programmer to manage the conversion of COMMAREA structures or 3270 screens to JCA records, which can be complicated and time-consuming. A toolled solution can automate this process.*
- *The performance and throughput requirements. The performance of hand-coded CCI applications might be slightly better than tool-generated applications because of the additional abstraction and runtime requirements of the tool-generated applications.*
- *The overall system-architecture requirements. The tooling capability of Rational Application Developer can be used to generate either servlet, EJB or Web services-based components.*

Using the JCA with different CICS Transaction Gateway topologies

The JCA system contracts are the key to the qualities of service provided by WebSphere Application Server and CICS ECI resource adapters. They provide the mechanisms by which the management of connections, security and transactions are performed. However, the qualities of service vary depending on the topology in use. The three most common topologies, shown in Figure 2, are:

- *Topology 1. WebSphere Application Server and the CICS Transaction Gateway are both deployed on a distributed (non-zSeries) platform.*
- *Topology 2. WebSphere Application Server is deployed on a distributed platform and CICS Transaction Gateway is deployed on a z/OS system.*
- *Topology 3. Both WebSphere Application Server and CICS Transaction Gateway are deployed on zSeries servers.*

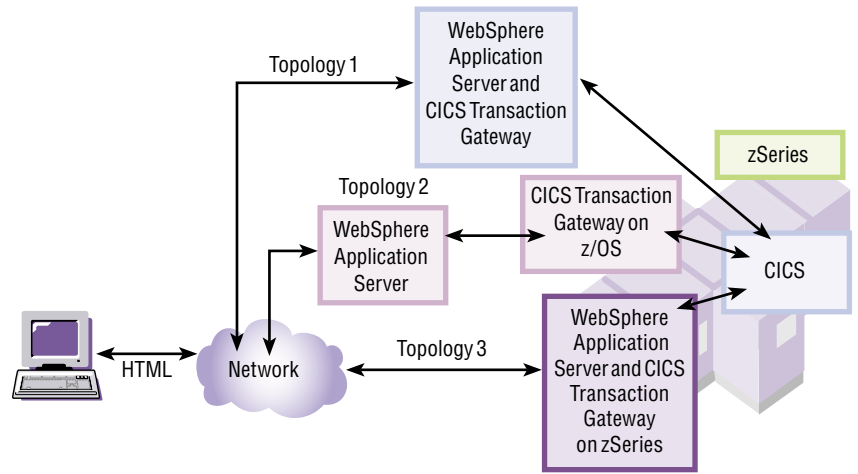


Figure 2. Common topologies for deployment of CICS Transaction Gateway and WebSphere Application Server

Topology 1: WebSphere Application Server and CICS Transaction Gateway deployed on distributed platforms

In this topology, both WebSphere Application Server and CICS Transaction Gateway are deployed on one of the distributed platforms, such as a Microsoft Windows or UNIX® platform. Both ECI and EPI resource adapters can be used in this configuration. Figure 3 shows an Enterprise JavaBeans (EJB) application using the ECI resource adapter to access a COMMAREA-based CICS COBOL application.

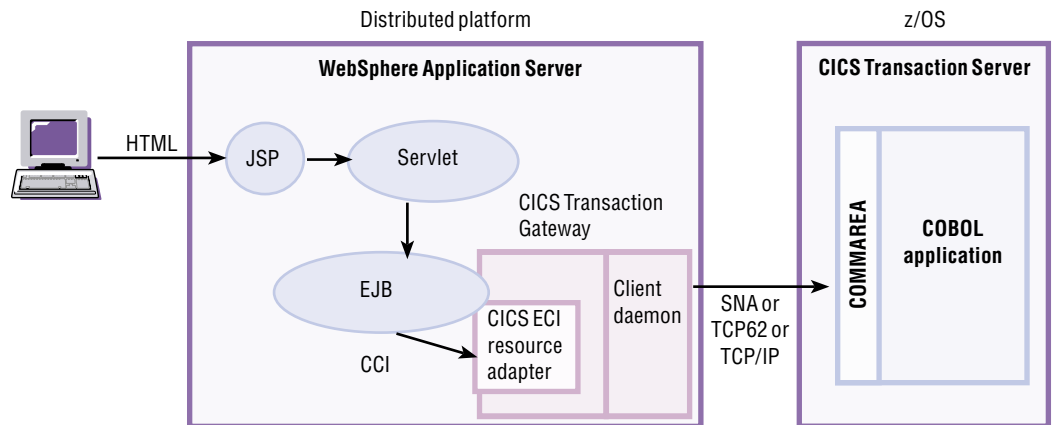


Figure 3. WebSphere Application Server and the CICS Transaction Gateway deployed on a distributed platform

In this configuration, the Gateway daemon is not required because the CICS Transaction Gateway local protocol is used to invoke the Client daemon native libraries directly from the enterprise bean. The protocol is specified on the gatewayURL parameter in the connection factory custom properties using the WebSphere Application Server administration console. The Client daemon flows the ECI request to the CICS server using a Systems Network Architecture (SNA) connection. If CICS Transaction Server is at Version 2.2 or later, the request from the Client daemon can be passed to CICS Transaction Server using TCP/IP directly.

This topology also applies to a completely distributed solution. In this case, instead of having CICS Transaction Server running on a mainframe, you can use CICS Transaction Gateway to connect WebSphere Application Server to CICS applications deployed in the IBM TXSeries® for Multiplatforms environment. The requests are flowed over TCP/IP, enabling WebSphere Application Server and TXSeries to be either on the same machine or on physically distributed machines. As with CICS Transaction Server on the mainframe, both EPI and ECI interfaces are supported.

The specific qualities of service (in terms of the JCA system contracts) that apply to this topology are as follows:

Connection pooling—Topology 1

Pooling of connections is provided seamlessly by the pool-manager component of WebSphere Application Server. This capability enables you to reuse connections to the resource adapter between multiple J2EE components, such as enterprise beans or servlets. The pool parameters are configured through the connection-pool settings for the connection factory. Because CICS Transaction Gateway and WebSphere Application Server both reside on the same host in this configuration, these pooled connections are actually a set of connection objects, each representing a local connection between WebSphere Application Server and the coresident CICS Transaction Gateway software. As a result, the connection pooling provides a benefit in terms of reduced memory and processor usage. The network connections from the CICS Transaction Gateway into CICS Transaction Server are managed and pooled independently by the CICS Transaction Gateway Client daemon.

Transaction management—Topology 1

The ECI resource adapter supports only the LocalTransaction interface. As a result, the scope of the transaction is limited to the Resource Manager (that is, the associated connection factory and the specified CICS server). Such Resource Manager Local Transactions support only one-phase-commit processing. So, if the enterprise bean shown in Figure 3 attempts to access another recoverable resource (for example, a JDBC call to update an IBM DB2® table), an exception occurs when the transaction manager tries to commit both updates. However, if you use a J2EE application server that supports the JCA option of last-resource commit optimization, an interaction using an ECIFactory class can participate in a global transaction, provided that it is the only one-phase-commit resource in the global transaction. This function is provided by the last-participant support in WebSphere Application Server, Version 6.0, and WebSphere Application Server Enterprise Edition, Version 5.0.

Security management—Topology 1

Security credentials (user ID and password) propagated through to CICS Transaction Server from WebSphere Application Server can be determined by the application (component managed) or by the Web or EJB container (container managed). Container-managed signon is recommended because it is good practice to separate the business logic of an application from qualities of service, such as security and transactions. In this topology, however, the principal means of enabling container-managed authentication is by specifying the user ID and password in a JCA authentication entry (also known as an *alias*) and associating the alias with the resource reference when the J2EE application is deployed.

A variation of Topology 1 is where WebSphere Application Server and CICS Transaction Gateway are installed on one of the distributed platforms but reside on different machines. The main difference in this variation is that the connection pool represents physical network connections between WebSphere Application Server and the Gateway daemon. As a result, the connection pool might provide greater benefit because it can reduce network overhead, as well as memory and processor consumption.

Topology 2: Remote Gateway daemon on z/OS

When WebSphere Application Server is deployed on one of the distributed platforms, it is possible to access CICS through a Gateway daemon running on z/OS, as shown in Figure 4.

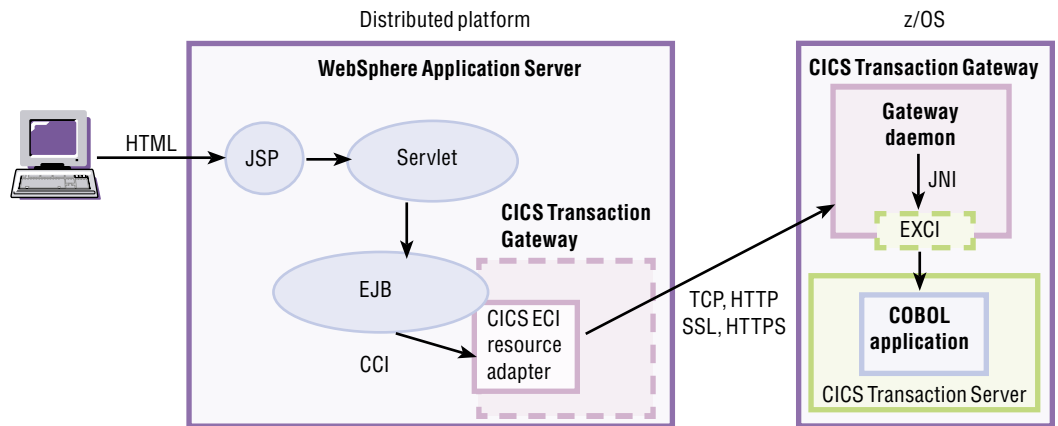


Figure 4. WebSphere Application Server deployed on a distributed platform and the CICS Transaction Gateway deployed on z/OS

In this configuration, the protocol specified in the connection settings of the connection factory is one of the remote protocols (TCP or SSL). The communication from the CICS Transaction Gateway on z/OS to the CICS server uses the External CICS Interface (EXCI). The EXCI is a function of CICS Transaction Server for z/OS that allows non-CICS address spaces on z/OS systems to link to programs in a CICS Transaction Server region.

This configuration is being widely used today because of the ease of support for TCP/IP connectivity into CICS Transaction Server, and the high qualities of service provided including load balancing and high-availability options.

When CICS Transaction Server security is enabled, Topology 1 requires that a user ID and password are flowed with each ECI request. In some situations, for example, when authentication is being undertaken using a different mechanism such as with client certificates, user ID authentication by CICS Transaction Server does not easily fit within the security design of a project. Using Topology 2 can help to avoid this problem because CICS Transaction Gateway for z/OS supports the concept of asserted identity, whereby a preauthenticated user ID is flowed into CICS Transaction Server without a password.

Topology 2 enables integration with z/OS Internet Protocol (IP) workload-management functions, including IBM Sysplex Distributor and TCP/IP port sharing. Use of these technologies allows an individual Gateway daemon to be removed as a single point of failure and enables incoming work to be efficiently balanced across multiple Gateway daemons in multiple z/OS logical partitions (LPARs). Topology 2 also allows CICS Transaction Gateway to be deployed into the zSeries environment, which can leverage the existing CICS systems-management skills within the enterprise.⁶

The WebSphere Application Server JCA connection-pooling mechanism can significantly improve performance when using network connections between WebSphere Application Server and a remote CICS Transaction Gateway daemon.

The specific JCA qualities of service that apply to this topology are as follows:

Connection pooling—Topology 2

The connection pool represents physical network connections between WebSphere Application Server and the Gateway daemon on z/OS. In such a configuration, it is essential to have an efficient connection-pooling mechanism because otherwise, a significant proportion of the time from making the connection to receiving the result from CICS Transaction Server and closing the connection can be in the creation and destruction of the connection itself. The JCA connection-pooling mechanism mitigates this overhead by allowing connections to be pooled by the WebSphere Application Server pool manager. This is particularly important when encrypted SSL connections are required, because the processing consumption of SSL connection handshaking typically uses the most resources in SSL communications.

Transaction management—Topology 2

In this topology, two-phase-commit global transactions are now supported, using the CICS ECI XA resource adapter along with CICS Transaction Gateway for z/OS, Version 6.1. Also, one-phase-commit transactions using the CICS ECI resource adapter are still supported. Note that in either case, if the enterprise bean is deployed with a transactional deployment descriptor (for example, a value of REQUIRED), the resulting ECI request to CICS Transaction Server uses the transactional EXCI. In this case, both CICS Transaction Gateway and CICS Transaction Server need to be configured to register with Resource Recovery Services (RRS).

Security management—Topology 2

In this configuration, the Gateway daemon is the entry point to the zSeries system in which your CICS system is running, so it is normal for the Gateway daemon to perform an authentication check for incoming ECI requests from clients. However, after the user has authenticated to WebSphere Application Server, a password might not be available to send to the Gateway daemon. In this case, therefore, you need to devise a way to establish a trust relationship between WebSphere Application Server and the Gateway daemon so that WebSphere Application Server can be trusted to flow only the user ID on the request through to CICS Transaction Gateway. Solutions such as SSL client authentication and virtual private networks (VPNs) can be used to establish such a trust relationship.

Topology 3: Deploying both WebSphere Application Server and CICS Transaction Gateway on zSeries servers

In a zSeries topology, WebSphere Application Server can be deployed on either a z/OS system or on a Linux operating system. The qualities of service differ between these two topologies and are therefore discussed separately.

Topology 3a: WebSphere Application Server and CICS Transaction Gateway on z/OS

In this topology, only CICS ECI resource adapters are supported. As in Topology 1, the most common z/OS configuration uses a local CICS Transaction Gateway. On z/OS, this results in a direct cross-memory EXCI connection between WebSphere Application Server and CICS. Figure 5 shows an application deployed to WebSphere Application Server for z/OS.

CICS Transaction Gateway, Version 5.01 introduced remote connection support for this topology, which specifically targets z/OS.e users.⁷ However, the highest qualities of service can be achieved when a local connection between WebSphere Application Server and CICS Transaction Gateway is used.

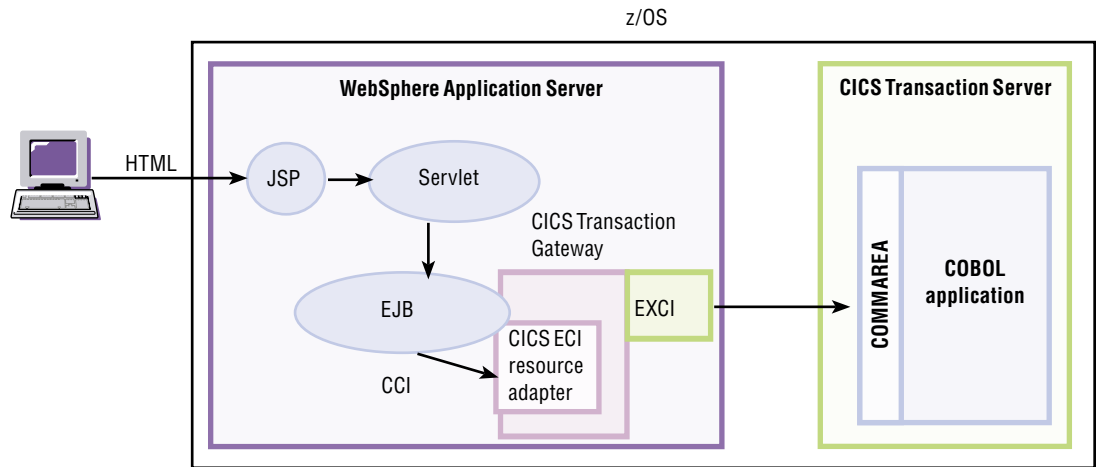


Figure 5. WebSphere Application Server and the CICS Transaction Gateway deployed on z/OS

These services relate to the three JCA system contracts as follows:

Connection pooling—Topology 3a

The connection pool is a set of connection objects managed by WebSphere Application Server, which is not directly associated with the EXCI pipes used for communication to CICS Transaction Server.

Transaction management—Topology 3a

A two-phase-commit capability is provided through RRS, an integral part of z/OS. RRS acts as the external transaction coordinator for z/OS in managing the transaction scope between WebSphere Application Server, CICS Transaction Server and other z/OS subsystems, including IBM IMS Transaction Manager (IMS/TM), IMS/DB, DB2 and IBM WebSphere MQ systems.

Security management—Topology 3a

Both container- and component-managed signon are supported. When using container-managed signon, WebSphere Application Server for z/OS provides a z/OS system-specific functionality known as *thread-identity support*.

You can configure WebSphere Application Server for z/OS to use a local operating-system registry (such as IBM RACF®) as its user registry.

Doing this helps ensure that the security identity established after authentication in WebSphere Application Server will be an RACF user ID if you use basic authentication or form-based login. If an SSL client certificate is used to authenticate, you can configure RACF to map that certificate to an RACF user ID. This means the Java thread in WebSphere Application Server on z/OS will have a security identity that is an RACF user ID. The unique thread-identity support in WebSphere Application Server for z/OS allows WebSphere Application Server to automatically pass the user ID of the thread (the caller's identity) to CICS Transaction Server when using the ECI resource adapter. This satisfies a common end-to-end security requirement of automatically propagating the authenticated caller's user ID from WebSphere Application Server to CICS Transaction Server.

In this topology, the ECI resource adapter flows only the user ID to CICS Transaction Server with the ECI request – it is assumed that the user is already authenticated by WebSphere Application Server. A trust relationship can be configured between WebSphere Application Server and the CICS server using the Multi-Region Operation (MRO) bind-and-link security mechanisms. Surrogate security checks can also be enabled to confirm that the user ID associated with the WebSphere Application Server region has the appropriate authority to flow a specific user ID (or one of a generic set of user IDs) to CICS Transaction Server.

Topology 3b: WebSphere Application Server and CICS Transaction Gateway on Linux on zSeries

When WebSphere Application Server is deployed within Linux on zSeries (shown in Figure 6), this configuration provides a flexible and scalable environment based on the virtualization capabilities of IBM z/VM® and Linux systems.

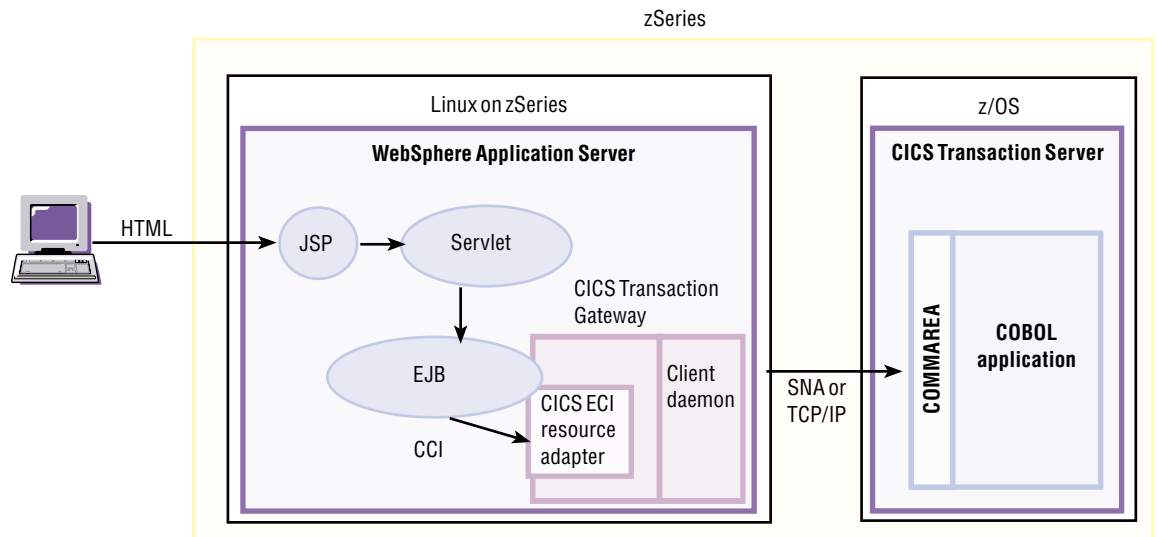


Figure 6. WebSphere Application Server and the CICS Transaction Gateway deployed on Linux on zSeries

The JCA qualities of service for this configuration are almost identical to those described for Topology 1 because Linux on zSeries (within a JCA and CICS Transaction Gateway scenario) can be treated as a distributed platform. A significant exception to this generalization is that the IBM HiperSockets™ hardware feature can be used to provide a highly efficient cross-memory transport for TCP/IP-based communication into CICS Transaction Server (using the ECI over TCP/IP function of CICS Transaction Server, Version 2.2).

The same choice of where to deploy CICS Transaction Gateway exists for implementations of WebSphere Application Server on Linux on zSeries as it does for implementations of WebSphere Application Server on distributed platforms. CICS Transaction Gateway can be deployed within the same Linux partition as WebSphere Application Server (in which case a local connection from WebSphere Application Server is used) or on z/OS as in Topology 2 (in which case a remote connection is used). The decision factors are the same as those described in Topology 2, which discusses the use of the Gateway daemon on z/OS.

Summary

Using the support for JCA provided by CICS Transaction Gateway enables you to easily integrate J2EE applications running in WebSphere Application Server with your existing (or new) CICS applications. The JCA system contracts define three qualities of service – for the management of transactions, security and connections. The operational benefits that you obtain from using the JCA depend on the chosen deployment topology. Table 2 summarizes how JCA qualities of service are implemented for the most commonly used topologies with WebSphere Application Server and CICS Transaction Gateway.

Table 2. Summary of JCA system contract support for CICS ECI resource adapters

Topology	Transactions	Security	Connection pooling
1. CICS Transaction Gateway and WebSphere Application Server on distributed platforms	Global transactions with last participant support (LPS)*, or resource manager local transactions	Component-managed or container-managed	Connection pooling of local, in-memory Connection objects
2. Remote Gateway daemon on z/OS	Global transactions with two-phase commit using the XA protocol, or resource manager local transactions	Component-managed or container-managed (including asserted identity)	Connection pooling of TCP/IP network connections from the WebSphere Application Server to the Gateway daemon
3a. CICS Transaction Gateway and WebSphere Application Server on z/OS	Global transactions with two-phase commit using MVS RRS	Component-managed or container-managed (including thread-identity support)	Connection pooling of local, in-memory Connection objects
3b. CICS Transaction Gateway and WebSphere Application Server on Linux on zSeries	Global transactions with two-phase commit using the XA protocol, or resource manager local transactions	Component-managed or container-managed	Connection pooling of local, in-memory Connection objects

*LPS is a function of IBM WebSphere Application Server Enterprise, Version 5 and WebSphere Application Server, Version 6.

For more information

To learn more about using JCA to integrate WebSphere Application Server with CICS Transaction Server, contact your IBM representative or IBM Business Partner, or visit:

For CICS Transaction Gateway

ibm.com/cics/ctg

For CICS Transaction Server and TXSeries

ibm.com/cics/tserver/

For the latest IBM documentation on CICS Transaction Gateway and the JCA

ibm.com/cics/ctg/library



© Copyright IBM Corporation 2006

IBM United Kingdom Limited

Hursley Park
Winchester
Hampshire
SO21 2JN
United Kingdom

Printed in the United States of America
01-06
All Rights Reserved

AIX, CICS, DB2, @server, HiperSockets, IBM, the IBM logo, IMS, the On Demand Business logo, OS/390, POWER, RACF, Rational, Tivoli, TXSeries, WebSphere, z/OS, zSeries and z/VM are trademarks of International Business Machines Corporation in the United States, other countries or both.

Intel is a trademark of Intel Corporation in the United States, other countries or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries or both.

UNIX is a trademark of The Open Group in the United States, other countries or both.

Linux is a trademark of Linus Torvalds in the United States, other countries or both.

Other company, product and service names may be trademarks or service marks of others.

- ¹ The comparison of the different CICS connector options are discussed in the "Delivering quick access to CICS systems using strategic integration options" white paper (G224-7557-00).
- ² If CICS Transaction Gateway is deployed on z/OS (including in WebSphere Application Server for z/OS), the only supported CICS server is CICS Transaction Server on z/OS or OS/390.[®]
- ³ To learn why COMMAREA-based programs are the best reuse option, read the "Delivering quick access to CICS systems using strategic integration options" white paper.
- ⁴ Before the existence of JCA, IBM recognized a need for a common way to connect to EIS systems and introduced the Common Connector Framework (CCF). JCA provides similar function to the CCF, but it is an open specification and provides stronger emphasis on system contracts and qualities of service.
- ⁵ The J2C functionality in WebSphere Application Server and Rational Application Developer refers to IBM's implementation of JCA in these products.
- ⁶ If the XA two-phase-commit functionality of CICS Transaction Gateway, Version 6.1 is used, then all the Gateway daemons used for load balancing must be located in the same z/OS logical partition (LPAR), because they all require access to the same MVS RRS.
- ⁷ z/OS.e is a specially priced offering of z/OS that provides select z/OS function for the zSeries 800 system. Traditional workloads, such as CICS Transaction Server, are not licensed for z/OS.e. Because the CICS Transaction Gateway relies on the EXCI provided by CICS Transaction Server for z/OS, CICS Transaction Gateway must be installed in a full-function z/OS LPAR. This means a remote CICS Transaction Gateway connection must be used from WebSphere Application Server into CICS Transaction Gateway.

