CICS integration solutions
White paper

WebSphere® software

Integrating WebSphere Application Server and CICS using CICS Transaction Gateway.

By Phil Wakelin, IBM Software Group, and Nigel Williams, IBM Design Center

March 2008

Contents

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 (CICS TS). Delivering access to new and existing CICS applications using standards-based interfaces is becoming a necessity as organizations deploy new service-oriented solutions.

IBM CICS Transaction Gateway (CICS TG) software has been proven over many years to deliver high performing, security-rich and scalable access to CICS applications based on Java™ 2 Platform, Enterprise Edition (J2EE) standards. Implementing CICS TG 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 SOA. CICS TG is available on a range of IBM platforms and can be deployed in a number of different topologies. This paper reviews the different qualities of service that are provided by the most commonly used topologies, and makes deployment recommendations which will help the reader to chose the topology which best fits his specific set of requirements.

Smart SOA

Based on a large number of client engagements, IBM has defined a continuum of service oriented architecture (SOA) approaches, which we refer to as IBM Smart SOA™. In the most basic approach, known as foundational, new applications are built to be composite, made up of discrete business functions defined and invoked through service interfaces. IBM has also created a set of SOA entry points that provide the means, through best practice advice and cookbooks, to pursue the foundational approach in a proven, well-established way. The reuse entry point covers the creation of services based on existing assets, such as CICS applications. The connectivity entry point covers the connectivity between processes and information, including the information encapsulated by CICS applications. CICS TG is an enabler of the foundational approach to Smart SOA. In particular, it provides the connectivity from service-enabled J2EE applications to CICS applications, which are reused as part of the service implementation.

You can use CICS TG in integrated SOA solutions to combine the strengths of WebSphere Application Server and CICS TS.

Using the J2EE Connector architecture to connect to CICS

A J2EE application can connect to CICS TS 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 TS. 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. CICS TG supports the use of the JCA for connecting to CICS TS by providing a set of JCA resource adapaters.

The JCA can be used by a number of IBM Rational® and IBM WebSphere® products to assist application developers in modeling, assembling, deploying and managing composite CICS and WebSphere applications. 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.

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

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 TS, Version 3.1 and later provides the ability to produce and consume Web services natively in CICS TS. 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 TS 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 different business problems. Tightly coupled direct connections and loosely coupled Web services can be used together to enable you to integrate all of your CICS assets in an enterprise-class SOA.

Understanding CICS Transaction Gateway

CICS TG 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 TG, Version 7.1 is the most recent version and has many improvements, principally in the area of systems monitoring and extended integration with CICS TS.

The two main programming interfaces supported by CICS TG are the external call interface (ECI) and external presentation interface (EPI). The ECI provides a remote procedure call (RPC)-style interface to CICS programs, whereas the EPI provides a programming interface to invoke 3270-based programs.[1]

CICS TG, Version 7.1 enables remote Java clients to call CICS applications using an extended version of the ECI. Support is provided for interprogram data transfer using either traditional communication areas (COMMAREAs) or the new CICS TS, Version 3 channels and containers programming model. This new model enables applications to exchange large amounts of structured data, far greater than the traditional 32 KB COMMAREAs. The ability to use channels and containers is available for Java clients that use the new IP interconnectivity (IPIC) access protocol introduced with CICS TS, Version 3.2.[2]

These are the major components of CICS TG:

- *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 TG supports both native TCP/IP and Secure Sockets Layer (SSL) connections.

- *Client daemon*
An integral part of CICS TG on all distributed platforms. It provides the SNA and TCP/IP connectivity using the same technology as IBM CICS Universal Client software.

- *Resource adapter*
A component that provides the CICS TG Java classes used by the Java client applications to invoke the services in an attached CICS server.

- *IPIC protocol driver*
A new component of CICS TG, Version 7.1 that supports IP connectivity from the Gateway daemon or the resource adapter directly to CICS TS, Version 3.2.

CICS TG offers several different topologies to choose from. For example, when using WebSphere Application Server, connections can be *local* (CICS TG communication is invoked through the facilities of a locally installed gateway) or *remote* (connections can be made to a Gateway daemon in a three-tier configuration). The Gateway daemon can be running on the same system as WebSphere Application Server or on another system.

JCA

JCA 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.

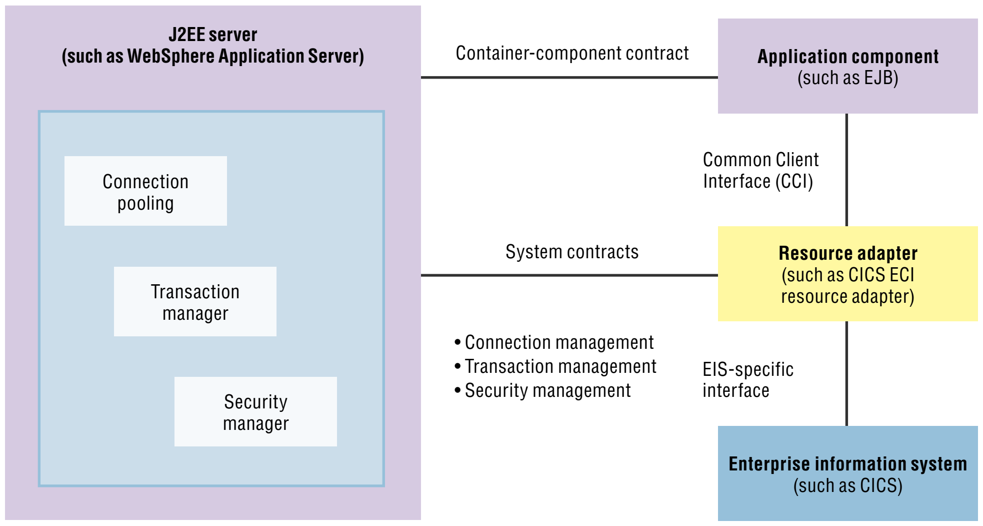

Figure 1. The JCA component structure

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 TS) 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 ECIConnectionSpec) are used to cater for 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, developer productivity can be increased when developing applications to communicate with multiple EISs.

Resource adapters

CICS TG, Version 7.1 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 or channel-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 TG on all platforms and provides one-phase-commit transactional coordination between WebSphere Application Server and CICS TS, and two-phase- commit support when used with WebSphere Application Server for z/OS.

- *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 either the use of CICS TG for z/OS, Version 6.1 or later, or CICS Transaction Gateway for Multiplatforms, Version 7.1 and CICS TS, Version 3.2.

- *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.

Each version of CICS TG provides these three resource adapters, all of which are licensed for deployment to J2EE application servers. CICS TG, Version 7.1 resource adapters are built using the Java5 software development kit (SDK) and can run only in a Java5 or later application server (for example, WebSphere Application Server, Version 6.1). To obtain copies of earlier versions of the CICS resource adapters that support WebSphere Application Server, Version 6.0 or Version 5.1, refer to CICS SupportPac™ CC03, available to licensed CICS TG users at:

ibm.com/support/docview.wss?uid=swg24008817

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 integrated solutions.

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 two-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 TS) 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.*
- 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 sign-on (in which the J2EE application server is responsible for flowing security context to the EIS) and component-managed sign-on (in which the application is responsible for flowing security context to the EIS) are supported.*

These system contracts are transparent to application developers, which means developers 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 the CICS ECI resource adapter:

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 instance of a Record or* ECIChannelRecord *representing the COMMAREA or channel interface to the CICS application.*
4. *Create an Interaction from the Connection. Specific interaction properties such as the transaction identifier can be passed using an* ECIInteractionSpec *object.*
5. *The call to the EIS is initiated by invoking the execute() method on the interaction, passing data as input and output records.*
6. *After the required interactions have been processed, the interaction and connection should be closed.*

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.

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 J2EE Connector (J2C) wizards.[3]

Generic classes	
ConnectionFactory	Creates a Connection 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 ConnectionFactory class.
Connection	A handle to a connection managed by the J2EE application server.
Interaction	A specific interaction with CICS TS 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 the user ID and password, that can be used to override values set in the ConnectionFactory class.
ECIInteractionSpec	CICS-specific interaction information, such as the mirror-transaction identifier and program name.
ECIChannelRecord	A wrapper for a JCA MappedRecord object, used to pass containers within the scope of a CICS channel.

Table 1. Generic and ECI resource adapter classes

Tool-generated development

The CCI is targeted primarily at application-development tools and enterprise application-integration frameworks. J2C wizards, an optional feature in IBM Rational Application Developer and IBM WebSphere Developer for System z software, provide a development environment that contains a range of JCA resource adapters, including the CICS ECI and EPI resource adapters. The J2C wizards are 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.

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

- *The availability of Java programming skills.*
- *Time-to-market requirements. Tool-generated applications generally take less time to develop.*
- *The complexity of the COMMAREA structures 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 tooled solution can automate some of this process.*
- *The overall system-architecture requirements. The tooling capability of Rational Application Developer can be used to generate either servlet, Enterprise JavaBeans (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 depend on the topology in use. The three most common topologies, shown in Figure 2, are:

- *Topology 1. WebSphere Application Server and the CICS TG are both deployed on a distributed platform.*
- *Topology 2. WebSphere Application Server is deployed on a distributed platform and CICS TG is deployed on an IBM z/OS® system.*
- *Topology 3. Both WebSphere Application Server and CICS TG are deployed on an IBM System z™ platform.*

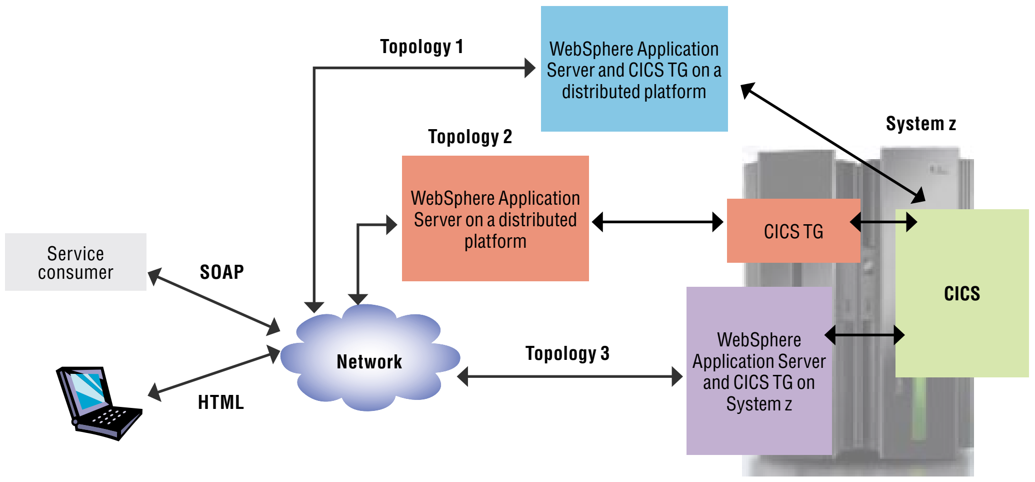
Topology 1
WebSphere Application Server and CICS TG on a distributed platform
System z
Topology 2
WebSphere Application Server on a distributed platform
CICS TG
CICS
Service consumer
SOAP
Network
Topology 3
WebSphere Application Server and CICS TG on System z
HTML

Figure 2. Common topologies for deployment of CICS TG 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 TG 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 a reusable EJB component that is exposed to different client types (a browser and a Web service requester). The EJB component uses a J2C bean, which is created using Rational Application Developer, to communicate with the CICS TG to access the CICS COBOL application.

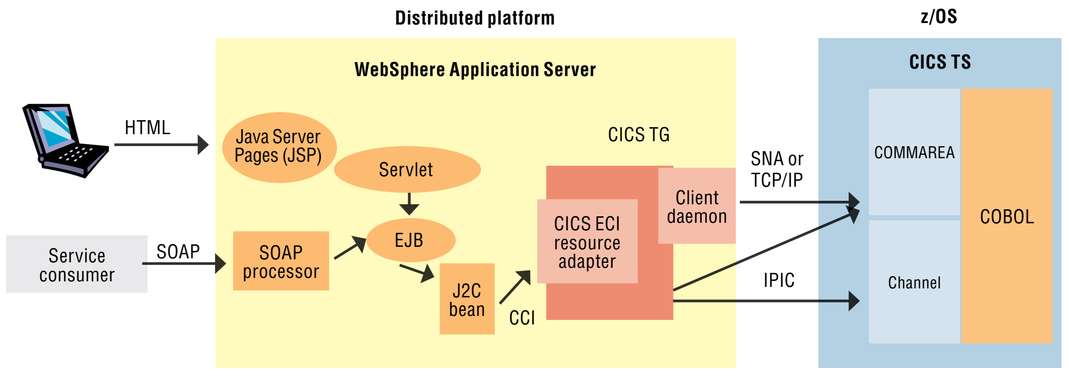

Figure 3. WebSphere Application Server and the CICS TG deployed on a distributed platform

In addition to accessing COMMAREA-based programs, CICS TG, Version 7.1 also supports interoperability with the CICS channels and containers programming model, allowing you to send and receive more than 32 KB of application data in a single ECI request. In order to call a channel-based program, you must use the IPIC protocol driver provided by CICS TG, Version 7.1, and you must connect to a CICS TS, Version 3.2 system or later.

In the Topology 1 configuration shown in Figure 3, the Gateway daemon is not required, because the CICS TG local protocol is used to invoke the Client daemon or the IPIC protocol driver directly from the J2C bean. The protocol is specified on the connectionURL parameter in the connection factory custom properties using the WebSphere Application Server administration console. The ECI request is flowed directly to the CICS server using either a TCP/IP, a Systems Network Architecture (SNA) connection or an IPIC connection.

The systems monitoring improvements provided with CICS TG, Version 7.1 for this topology include:

- *Advanced system metrics, providing enhanced support for problem determination and capacity planning.*
- *A transaction tracking infrastructure providing dynamic point-of-origin information when using IPIC connections to CICS.*
- *A transaction monitoring exit infrastructure for detailed analysis of transactions as they pass through the CICS TG components.*

This topology also applies to a completely distributed solution. In this case, instead of having CICS TS running on a mainframe, you can use CICS TG 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 system or on physically distributed systems. As with CICS TS for z/OS 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 topology 1 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 TG 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 co-resident CICS TG software. As a result, the connection pooling provides a benefit in terms of reduced memory and processor utilization. The network connections from the CICS TG into CICS TS are managed and pooled independently by the CICS TG, and are not subject to the JCA connection pooling system contract.

Transaction management: Topology 1

Two-phase-commit global transactions are supported, using the CICS ECI XA resource adapter and IPIC connectivity directly to CICS TS, Version 3.2. In addition, the CICS ECI resource adapter can be used with any version of CICS. The ECI resource adapter supports only one-phase-commit local transactions and, as result, the scope of the transaction is limited to a single CICS region. For more information, refer to the article *Exploiting the J2EE Connector Architecture, Integrating CICS and WebSphere Application Server using XA global transactions.*[4]

Security management: Topology 1

Security credentials (user ID and password) propagated through to CICS TS from WebSphere Application Server can be determined by the application (component managed) or by the Web or EJB container (container managed). Container-managed sign-on is recommended because it allows separation of application business logic from qualities of service, such as security and transactions. In this topology, 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, although it is also possible to use a custom login module to perform more advanced security mappings (see *Database identity propagation in WebSphere Application Server V6*.[5]) When an IPIC connection is used, asserted identities are also supported. In this case, a preauthenticated user ID can be flowed into CICS without a password, provided that an SSL client-authenticated connection is used.

Variants of topology 1

A variation of topology 1 is the use of a remote gateway, where WebSphere Application Server and CICS TG reside on different systems. 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 provides greater benefit because it can reduce the 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.

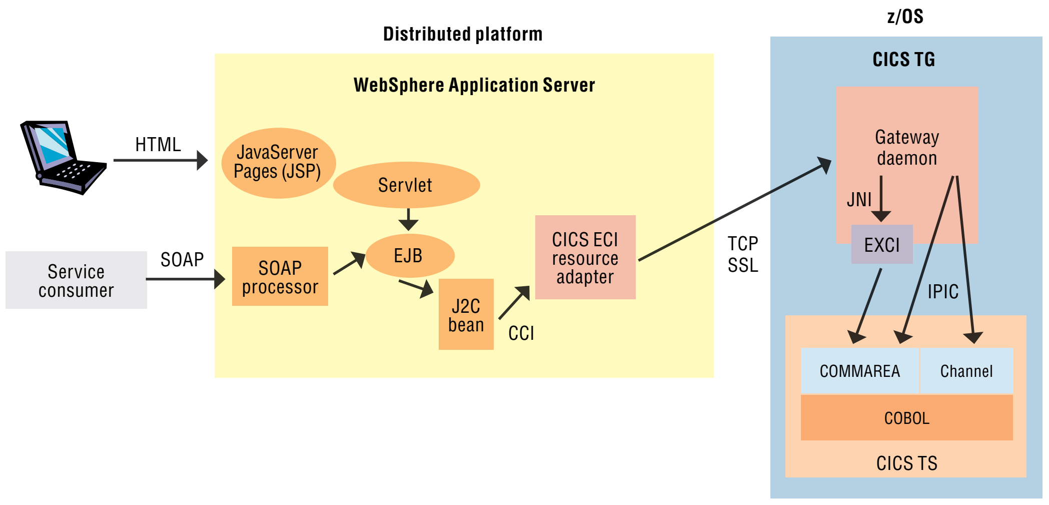

Figure 4. WebSphere Application Server deployed on a distributed platform and the CICS TG deployed on z/OS

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 Gateway daemon to the CICS server can use the External CICS Interface (EXCI) or – if the version of CICS TS is 3.2 – the IPIC protocol. The IPIC option allows you to send and receive more than 32 KB of application data in a single ECI request and provides additional transaction tracking capabilities.

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

This configuration is being widely used today because of the ease of support for TCP/IP connectivity into CICS TS and the high qualities of service provided, including load balancing and high-availability options. It also provides the ability to reduce the CPU cost of the Gateway daemon processing, which is written in the Java language, by exploiting zAAP processors. Topology 2 enables integration with z/OS IP workload-management functions, including IBM z/OS Workload Manager (WLM), 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). This topology also allows CICS TG to be deployed into the System z environment, which can leverage existing CICS systems-management skills within the enterprise.[6]

In addition to the systems monitoring improvements provided with CICS TG, Version 7.1 for topology 1, this topology provides support for interval-based statistics, recorded offline to the System Management Facilities (SMF). These statistics can be processed with CICS Performance Analyzer for z/OS, Version 2.1, providing the ability to integrate CICS TG systems monitoring with standard policies and procedures used on z/OS.

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 TS 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 cost of SSL connection handshaking is typically the most expensive component in SSL communications.

Transaction management: Topology 2
This topology supports two-phase-commit global transactions, using the CICS ECI XA resource adapter, and either EXCI or IPIC connections into CICS TS. Also, it supports one-phase-commit transactions using the CICS ECI resource adapter.

Security management: Topology 2
In this configuration, the Gateway daemon is the entry point to the System z platform 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, the Gateway daemon can be configured to accept a user ID without a password; however, you should then 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. Solutions such as SSL client authentication and virtual private networks (VPNs) can be used to establish such a trust relationship.

A trust relationship should also be configured between the Gateway daemon and the CICS server in one of the following ways:

- *When an EXCI connection is used, trust can be established using the multiregion operation (MRO) bind and link security mechanisms. Surrogate security checks can also be enabled to confirm that the user ID associated with the Gateway daemon has the appropriate authority to flow a specific user ID (or one of a generic set of user IDs) to CICS.*
- *When an IPIC connection is used, trust can be established by protecting the IP address in use by the CICS region, so that only a specific set of Gateway daemons can connect. This involves using the IBM RACF® SERVAUTH class profile and the ability of CICS to determine that the partner application is resident in the same sysplex.*

Topology 3: Deploying both WebSphere Application Server and CICS Transaction Gateway on System z

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

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

In this topology, only the CICS ECI resource adapters are supported. As in topology 1, the most common z/OS configuration involves a direct connection to CICS without passing through a Gateway daemon. On z/OS, this results either in a cross-memory EXCI connection between WebSphere Application Server and CICS, or an IPIC connection that uses optimized fast local sockets. The IPIC option allows you to send and receive more than 32 KB of application data in a single ECI request, provides additional transactional tracking capabilities and supports communication between WebSphere Application Server and CICS on different LPARs. Figure 5 shows the application deployed to WebSphere Application Server for z/OS.

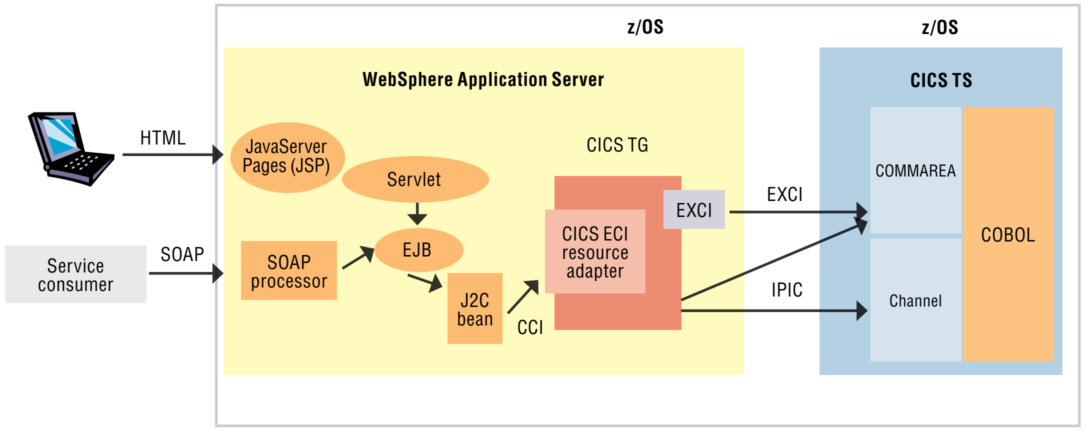

Figure 5. WebSphere Application Server and CICS TG deployed on z/OS

Remote connection support is also provided for this topology. However, the highest qualities of service can be achieved when a direct connection between WebSphere Application Server and CICS is used (as shown in Figure 5). 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 or IPIC sessions used for communication to the CICS server.

Transaction management: Topology 3a

A two-phase-commit capability is provided for either EXCI or IPIC connections. When using EXCI, Resource Recovery Services (RRS) – an integral part of z/OS – provides an optimized transaction coordination service, managing the transaction scope between WebSphere Application Server and CICS, if both WebSphere and CICS reside in the same LPAR. When using IPIC, XA requests are sent directly to CICS from the CICS ECI XA resource adapter with the benefit that WebSphere Application Server and CICS do not have to reside in the same LPAR.

Security management: Topology 3a

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

This function allows WebSphere Application Server to automatically pass the user ID of the thread (the caller's identity) to CICS TS when using an 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 TS. Thread identity support is provided for both EXCI and IPIC connections to CICS. Although, when using thread identity support, EXCI is only available if WebSphere Application Server and CICS reside in the same LPAR. To protect unauthorized access to CICS when using thread identity, a trust relationship should be configured between WebSphere Application Server and the CICS server in on of the following ways:

- *When an EXCI connection is used, trust can be established using the 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.*
- *When an IPIC connection is used, trust can be established by protecting the IP address in use by the CICS region so that only a specific set of WebSphere Application Server regions can connect. This involves using the SERVAUTH class profile and the ability of CICS to determine that the partner application resides in the same sysplex. Additionaly, an SSL connection can also be used to provide robust support for encryption and client authentication.*

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

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

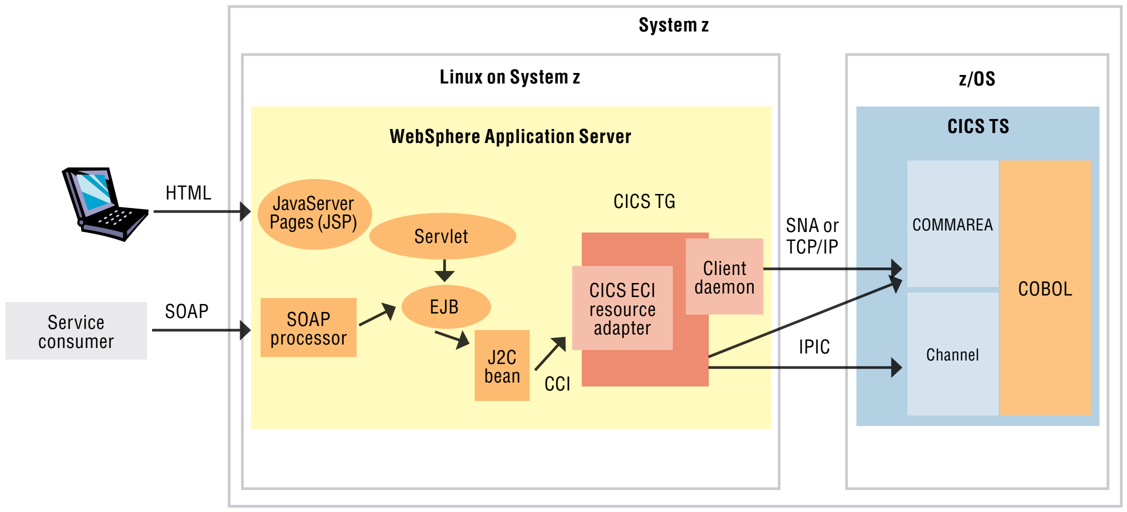

Figure 6. WebSphere Application Server and the CICS TG deployed on Linux on System z

The JCA qualities of service for this configuration are almost identical to those described for topology 1 because Linux on System z (within a JCA and CICS TG 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. Alternatively, an Advanced Program-to-Program Communication (APPC) connection to CICS TS or CICS Transaction Server for VSE/ESA™ can be provided by IBM Communications Server for Linux.

The same choice of where to deploy CICS TG exists for implementations of WebSphere Application Server on Linux on System z as it does for implementations of WebSphere Application Server on distributed platforms. CICS TG 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 deployment of the Gateway daemon on z/OS.

Summary

Using the support for JCA provided by CICS TG enables you to easily integrate J2EE applications running in WebSphere Application Server with your existing (or new) CICS applications. Then by service-enabling the J2EE applications, you can quickly expose your CICS applications as services.

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 what JCA qualities of service are enabled for the most commonly used topologies with WebSphere Application Server and CICS TG.

Topology	Connection pooling	Transactions	Security
1. CICS TG and WebSphere Application Server on distributed platforms	Connection pooling of local in memory connection objects	Global transactions with two-phase commit using the XA protocol (when IPIC connectivity is used to CICS TS, Version 3.2 or later)	Asserted identity with IPIC and SSL client - authenticated connection to CICS TS, Version 3.2 or later
2. Remote Gateway daemon on z/OS	Connection pooling of TCP/IP network connections from the J2EE application server to the Gateway daemon	Global transactions with two-phase commit using the XA protocol to any supported release of CICS TS	Asserted identity to any supported release of CICS TS (SSL or non-SSL)
3a. CICS TG and WebSphere Application Server on z/OS	Connection pooling of local in memory connection objects	Optimized global transactions with two-phase commit using RRS	Asserted identity and unique thread identity support
3b. CICS TG and WebSphere Application Server on Linux on System z	Connection pooling of local in memory connection objects	Global transactions with two-phase commit using the XA protocol (when IPIC connectivity is used to CICS TS, Version 3.2 or later)	Asserted identity with IPIC and SSL client-authenticated connection to CICS TS, Version 3.2 or later

Table 2. Summary of CICS TG and WebSphere Application Server topologies

For more information

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

For CICS TS

ibm.com/cics

For WebSphere Application Server

ibm.com/software/webservers/appserv/was/

For Rational Application Developer

ibm.com/software/awdtools/developer/application/index.html

[1] To learn why using the ECI to interface to CICS programs is the best reuse option, read the white paper *Options for Integrating CICS applications in an SOA*, which can be downloaded from:

ibm.com/software/htp/cics/tserver/v32/library/index.html

[2] To learn more about the new IPIC support delivered by CICS TS, Version 3.2, refer to the white paper *CICS delivers IP interconnectivity* which can be downloaded from:

ibm.com/software/htp/cics/tserver/v32/library/index.html

[3] The J2C function in WebSphere Application Server and Rational Application Developer refers to IBM's implementation of JCA in these products.

[4] Exploiting the J2EE Connector Architecture, **ibm.com**/developerworks/websphere/techjournal/0607_wakelin/0607_wakelin.html

[5] Database identity propagation in WebSphere Application Server V6, **ibm.com**/developerworks/websphere/techjournal/0506_barghouthi/0506_barghouthi.html

[6] If the XA two-phase-commit function of CICS Transaction Gateway V7.1 for z/OS (or earlier) is used, then all the Gateway daemons used for load balancing must be located in the same z/OS LPAR, because they all require access to the same IBM MVS RRS.

WSW14013-USEN-00