

CICS and WAS on z/OS: New Integration Capabilities

Leigh Compton
IBM Advanced Technical Skills
lcompton@us.ibm.com



SHARE in Boston

Abstract

- One advantage of running Java workloads in the z/OS distribution of WebSphere Application Server is the ability to provide connectivity to CICS applications with a very high capacity and high availability. New capabilities in WebSphere Application Server, CICS Transaction Gateway, and even CICS Transaction Server itself provide additional options for configuring the connections between the Java applications and CICS applications. The speaker will describe the use of CICS Transaction Gateway and Optimized Local Adapters in WAS, capabilities of each, and considerations for choosing when to use one versus the other.

Agenda

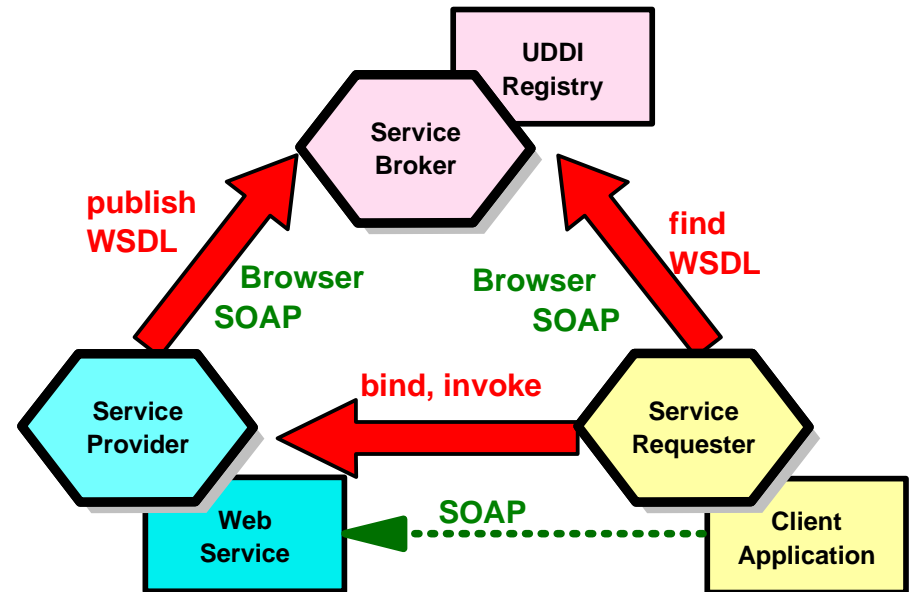
- WAS and CICS Integration
 - Transactional Interoperability
 - Transactionality, unit-of-work
 - Java calling CICS
 - CICS calling Java
- Web services
- CICS Transaction Gateway
- WebSphere Optimized Local Adapter (WOLA)

Java and CICS

- Goal is interoperability
 - With traditional qualities of service
 - Reliability
 - Availability
 - Transactional integrity
 - Security
- Calling CICS applications from WAS:
 - Web services
 - CICS TG
 - WebSphere Optimized Local Adapter
- Calling Java applications from CICS:
 - Web services
 - WebSphere Optimized Local Adapter

Web Services

- Architecture for
 - Application to application
 - Communication
 - Interoperation
- Definition:
 - Web Services are **software components described via WSDL** that are capable of being accessed via **standard** network protocols such as **SOAP** over **HTTP**
- WS-I.org (Web Services Interoperability Organization):
 - An organization to ensure interoperability



The entire industry is agreeing on one set of standards !!

Notes:

- Web services describe the role of a service provider which is some entity that has exposed an available business function.
- There is also the role of a service requester: an entity that would like to consume the business function made available by the service provider.
- The service provider and service requester communicate using SOAP (Simple Object Access Protocol) over HTTP (HyperText Transfer Protocol).
- WSDL (Web Service Description Language) is used to describe the interface to a business function. The WSDL includes a description of the available business function, what gets passed to the business function, what gets returned from the business function, the protocol (e.g. SOAP over HTTP), and the location of the Web service (also called the endpoint).
- CICS supports both SOAP over HTTP and SOAP over WMQ.
- Initially, most people thought the primary use of Web services would be to do business over the Internet. Today, the primary use of Web services is to provide interoperability between business entities within a company. In an Internet-based Web service environment, information about Web services are often placed in a business registry. This is similar to a telephone book. Potential users of the business function can do a 'lookup' in the business registry and 'find' someone that offer the business function they need. They can solicit the interface to the business function in question, so they can start doing business with that business using SOAP over HTTP.
- The WS-I (Web Service Interoperability Organization) was formed by over 170 companies to ensure everyone interprets the Web service standards the same way and that we really do achieve the interoperability that Web services promises.
- We, in the IT industry, have tried several techniques to interoperate in that past. This time around, everyone is agreeing on a single set of standards.

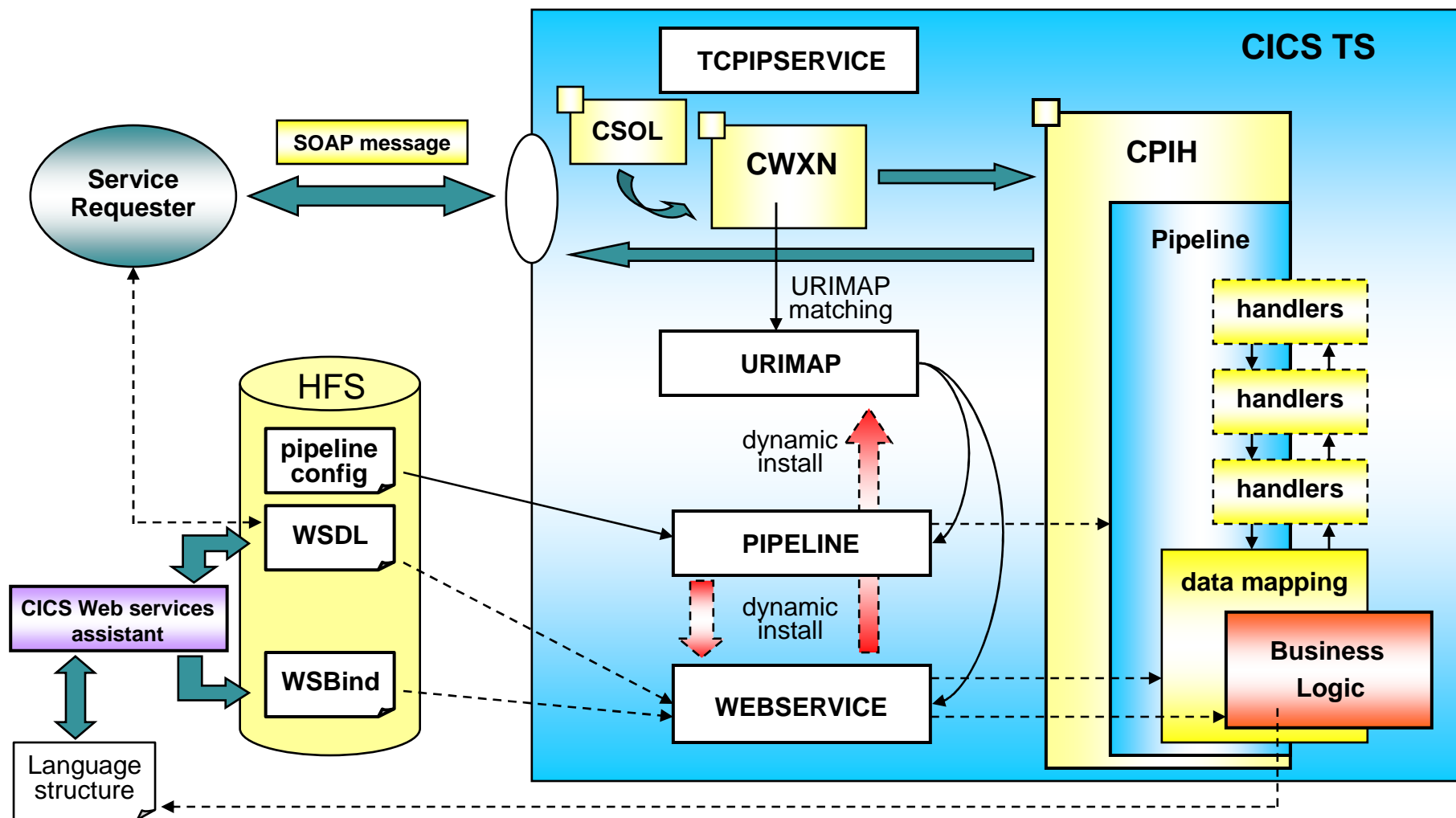
CICS Support of Web Service Standards

- XML Encryption Syntax and Processing
 - XML Signature Syntax and Processing
 - SOAP 1.1 and 1.2
 - WSDL 1.1 and 2.0
 - WSDL 1.1 Binding Extension for SOAP 1.2
 - WS-I Basic Profile 1.1
 - WS-I Simple SOAP Binding Profile 1.0
 - WS-AtomicTransaction
 - WS-Coordination
 - WS-Security
 - WS-Trust
 - MTOM / XOP
 - SOAP 1.1 Binding for MTOM 1.0
 - WS-Addressing
 - Both HTTP and WebSphere MQ network layers supported
 - HTTP 1.0 and 1.1 supported
 - CICS applications acting as providers or requesters are agnostic to transport mechanism used
- interoperability with entities using XML
 interoperability with entities using XML
 to send and receive Web services messages
 to describe Web service interfaces (WSDL 2.0 in TS 3.2)
 for interoperability with interfaces
 for interoperability between providers and requesters
 using SOAP
 for interoperability using SOAP
 for propagating transactional context
 coordinating transaction outcome
 authentication and encryption of all or part of a
 message, username token profile 1.0, X.509
 Certificate Token
 for establishing trust relationships (in TS 3.2)
 for efficient handling of large messages (in TS 3.2)
 to describe the use of MTOM (in TS 3.2)
 to indicate request and response routing (in TS 4.1)

Notes:

- This is a list of the Web service-related specifications that CICS supports.
- More details on these in the CICS Information Center.

Web Services: CICS as a Provider



Notes

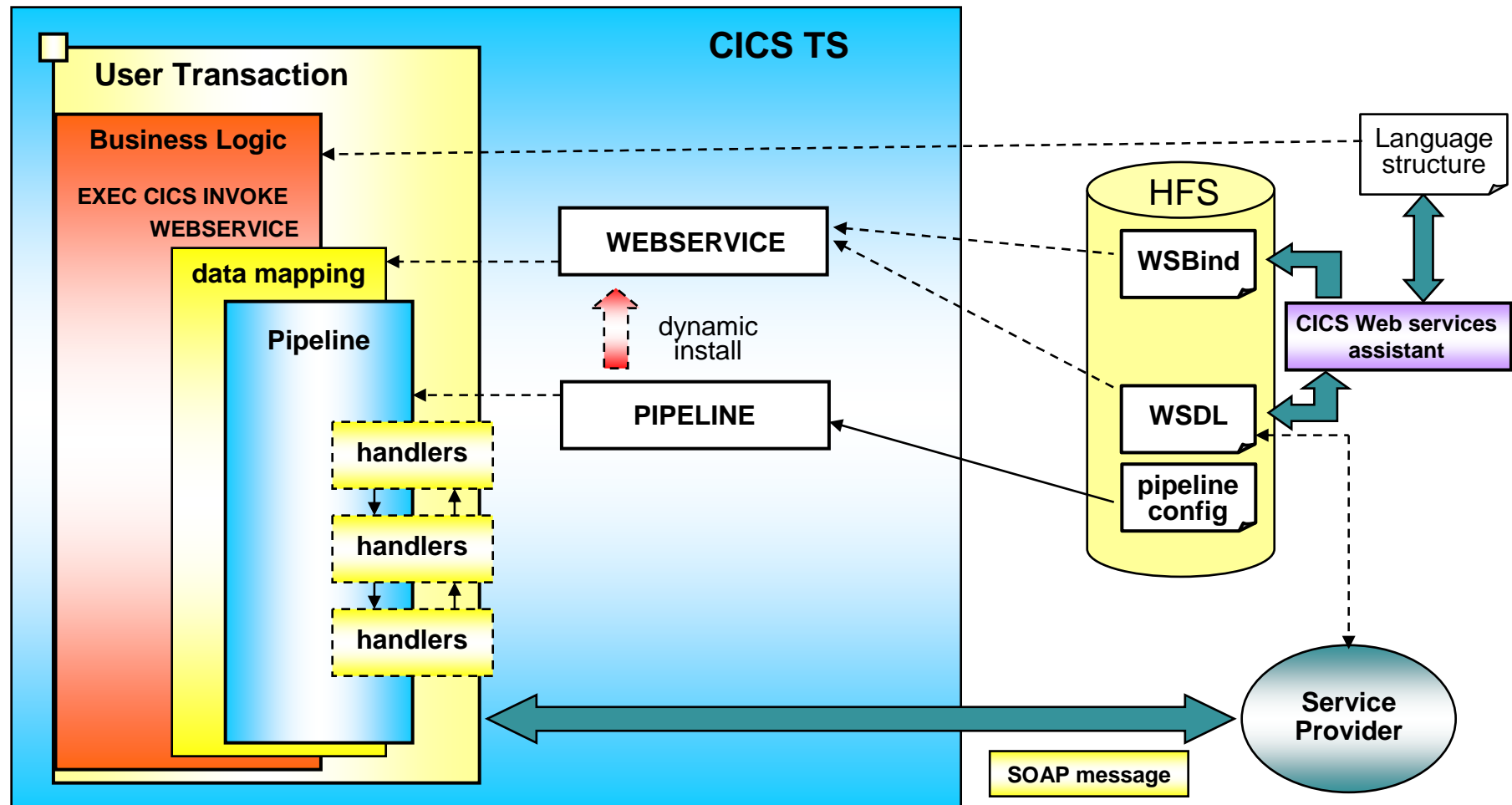
This diagram illustrates the entire flow and the CICS resources necessary to allow CICS to function as a service provider.

The Web services artifacts are built using the CICS Web Services Assistant. The CICS Web Services Assistant can be used in two scenarios. It can take the language structure of the business logic interface (a COMMAREA or container layout) and generate a WSDL document and a WSBind file. The WSDL document will be used to create a service requester. The CICS Web Services Assistant can also take a WSDL document as input and generate WSBind file and a language structure which will be used as the interface for new business logic or a wrapper program. The WSDL and WSBind files reside on HFS. The pipeline configuration file also resides on HFS. The PIPELINE resource definition points to the pipeline configuration file. From the PIPELINE resource, a WEBSERVICE resource and a URIMAP resource can be automatically installed.

When a SOAP message arrives through HTTP, the web attach transaction (CWYN) will scan the incoming HTTP request and find a matching URIMAP resource. When the URIMAP specifies USAGE(PIPELINE) it will attach a pipeline transaction called CPIH. CPIH will start the pipeline processing, invoke the handler programs according to the pipeline configuration file. It will use the information from the WEBSERVICE resource and convert the SOAP message into a language structure and LINK to the target application via COMMAREA or a CHANNEL. When the target application returns, the output language structure is converted back to a SOAP message and be passed back through the pipeline and will be sent back to the requester. The transport can also be WebSphere MQ.

The target application can be run under a different transaction id, and can be routed to a different CICS system connected via MRO.

Web services: CICS as Requester



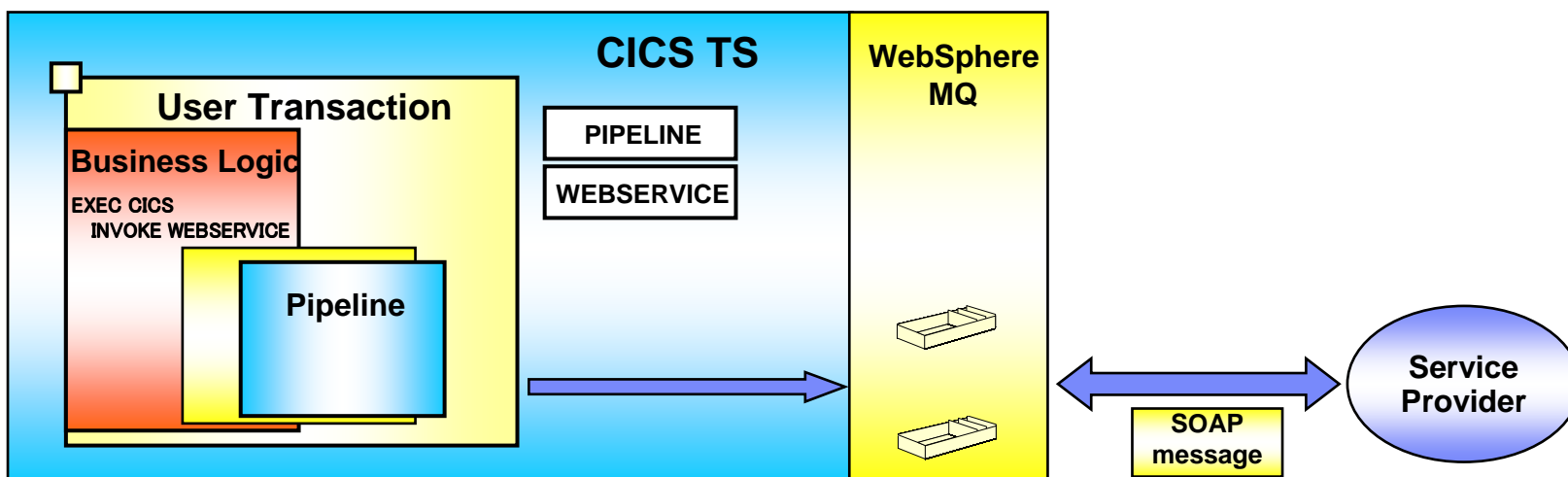
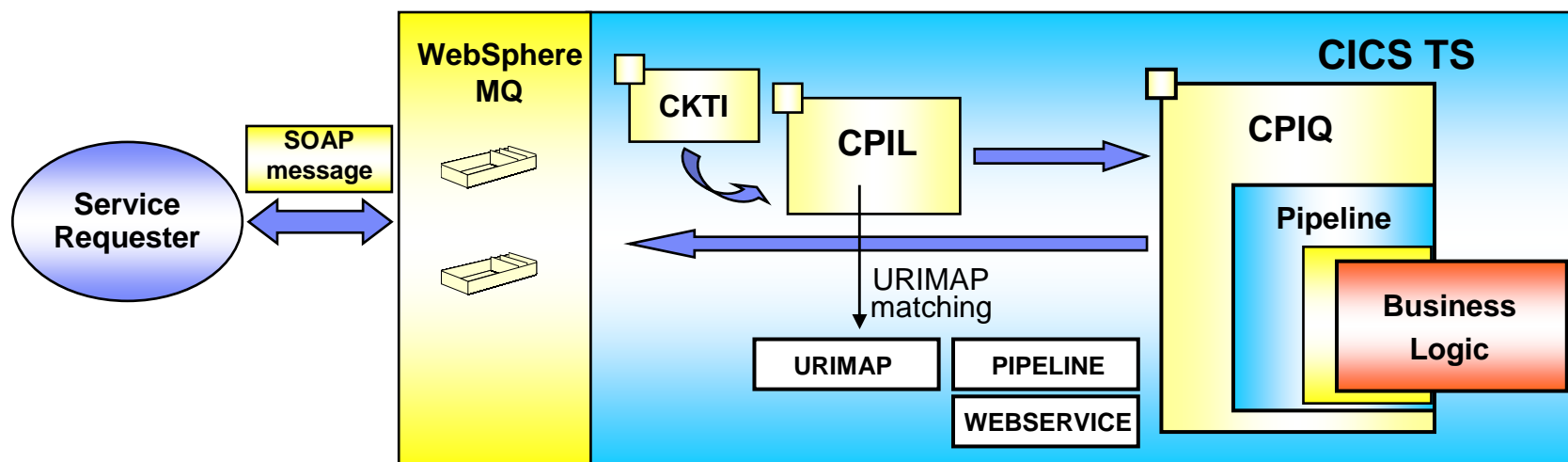
Notes

This diagram illustrates the flow and the CICS resources necessary to allow CICS to function as a service requester.

The WSDL document for the remote service provider will be processed through the Web services assistant. The utility will generate a language structure, which will be used as interface from the service requester program, and a WSBind file. There will be a PIPELINE resource pointing to a pipeline configuration file, and a WEBSERVICE resource.

The service requester program will issue an EXEC CICS INVOKE WEBSERVICE command, passing the request language structure via a CHANNEL interface. Information in the corresponding WEBSERVICE resource will be used to convert the language structure into a SOAP message. The SOAP message will be passed through the pipeline, invoking handler programs according to the pipeline configuration file. It will send the request SOAP message to the remote service provider either via HTTP or WebSphere MQ. When the response SOAP message is received, it will be passed back through the pipeline, converted back into a language structure, and passed back to the service requester program.

WebSphere MQ Transports



Notes

When using WMQ as a CICS Web service transport, the client places a SOAP message on a WMQ queue. In addition to the SOAP message, the client also places path information in the WMQ message header called an RFH2 header.

The CICS Systems Programmer would need to configure CICS to listen on a specific queue where the Web service requests will arrive. The work of listening on the queue is performed by the CKTI transaction. Once the item is obtained from the queue, the CPIL transaction compares the path in the RFH2 header with the paths specified in the URIMAP definitions. From then on, the flow is the same as with HTTP with the exception that the pipeline is processed under the CPIQ transaction instead of CPIH.

When CICS is the requester using WMQ as the transport, the application programmer uses the same commands as when using HTTP. They place the data to be passed to the Web service in a container, then invoke the INVOKE WEBSERVICE command. The WEBSERVICE specified after the WEBSERVICE keyword indicates to CICS as to whether to use HTTP or WMQ as a transport.

Web Service Invocation API

- EXEC CICS INVOKE WEBSERVICE() CHANNEL() URI()
OPERATION()
 - WEBSERVICE: name of the Web Service to be invoked
 - CHANNEL: name of the channel containing data to be passed to the Web Service (DFHWS-DATA container)
 - URI: Universal Resource Identifier of the Web Service (optional)
 - OPERATION: name of the operation to be invoked
 - V3.2 – timeout value (RESPWAIT) can be specified on the PIPELINE definition

Notes

The purpose of the command is to invoke the web service named and pass the channel into which the relevant containers have been put.

The container DFHWS-DATA must be created by the requesting application before the INVOKE WEBSERVICE command is issued.

The same container, DFHWS-DATA, will hold the response, if any, from the Web Service.

If a fault is returned from the Web service provider, CICS will provide the CICS application program with appropriate RESP and RESP2 values.

Web services: features and benefits

- Open, standards-based interoperability
 - Supported by many platforms
- WSDL clearly defines interfaces
- Rich set of QOS options
 - Security
 - Transactionality
 - Policies

About the CICS Transaction Gateway

- Offers high performance access to CICS TS
 - Secure
 - Scalable
- Provides J2EE™ standards-based connectivity
- Insures maximum transactional integrity
 - Two-phase-commit transactional integration between J2EE application servers and CICS
- CICS TG for z/OS
 - Supports local and remote Java clients
 - Provides remote client connectivity for C applications
- CICS TG for Multiplatforms
 - Supports local and remote Java clients
 - Supports local clients coded in C, C++, COBOL, and Visual Basic

Notes

IBM CICS Transaction Gateway provides highly flexible, security-rich and scalable access to CICS applications. It requires minimal changes to CICS systems and usually no changes to existing CICS applications.

CICS TG supports the following platforms:

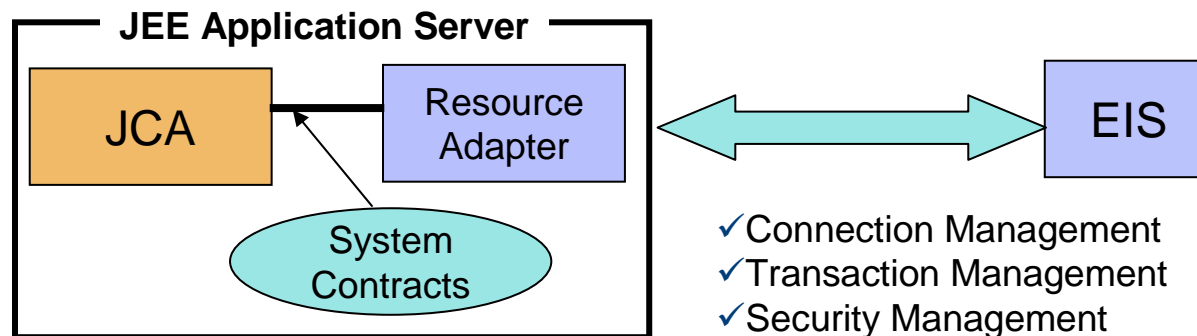
- IBM z/OS
- IBM AIX
- Linux on Intel, IBM POWER or IBM System z
- Microsoft Windows
- Sun Solaris on the SPARC platform
- HP-UX on RISC or Itanium platforms

Connectivity is provided on these platforms from all supported WebSphere Application Server environments to all supported CICS servers. The strategic SOA interface within the CICS Transaction Gateway is the J2EE Connector Architecture (JCA) interface. For maximum flexibility, programming interfaces are also provided in Java, C, C++, COM (for Microsoft Visual Basic) and COBOL.

CICS Transaction Gateway provides an XA-capable JCA ECI resource adapter. A J2EE application can invoke a CICS application using a two-phase commit transaction. This capability enables CICS Transaction Gateway to fully participate in a global transaction, where units of work can be coordinated across different resource managers,

About the J2EE Connector Architecture

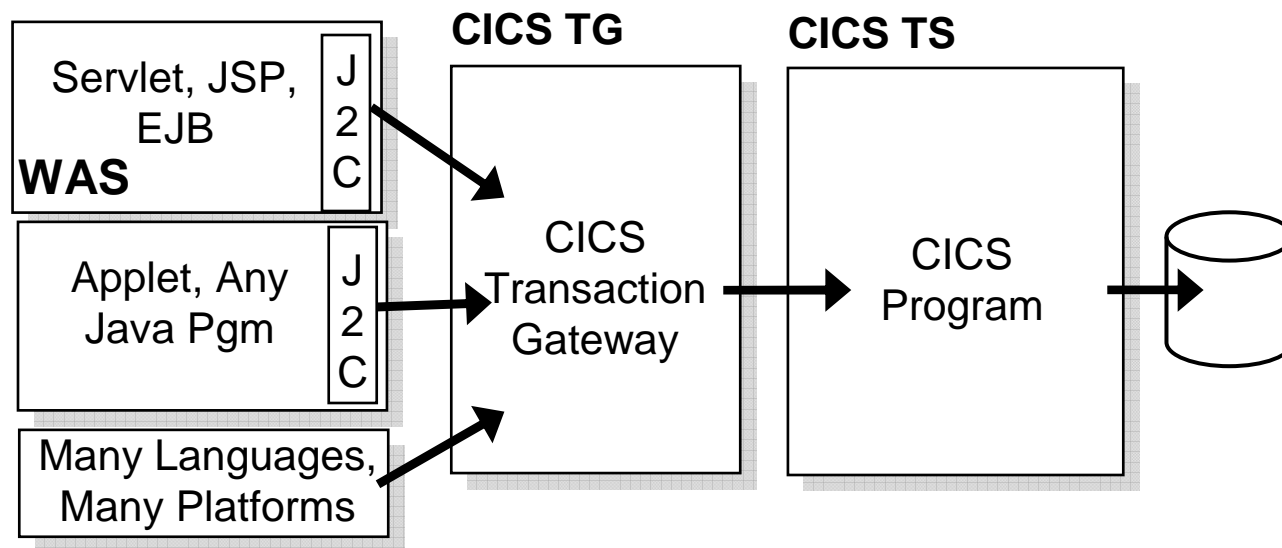
- Component of the Java 2 Platform Enterprise Edition specification
 - alongside other standard services, such as JMS, JDBC, and JNI
- Standard programming interface to all Enterprise Information Systems (EIS), such as CICS, IMS, and SAP
- Delegated management of Connections, Transactions, and Security for better, faster application development
- Widely supported in educational material and software tooling



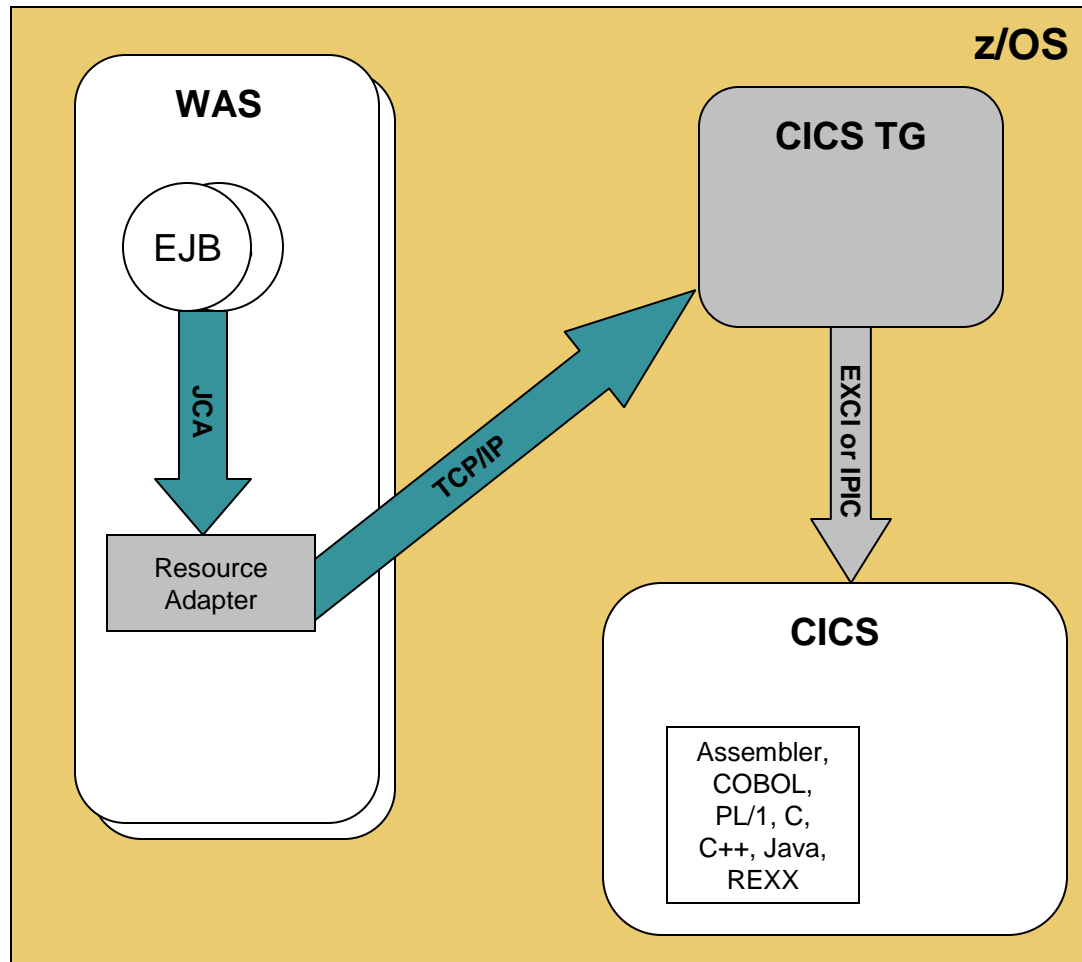
Notes

CICS Transaction Gateway supports the standard JCA, version 1.5 specification as its strategic interface. As a component of the J2EE specification, along with other standard services, the JCA provides a standard programming interface to all enterprise information systems (EISs). Using JCA offers two significant development advantages. First, it enables J2EE developers to program to a standard interface that is widely supported in educational materials and software tooling from IBM and other vendors. Second, JCA provides delegated management of connection pooling, transactional scope and security control so that J2EE developers don't have to develop these capabilities within the application. Together, these benefits mean better applications can be developed faster and more easily.

CICS access via CICS TG



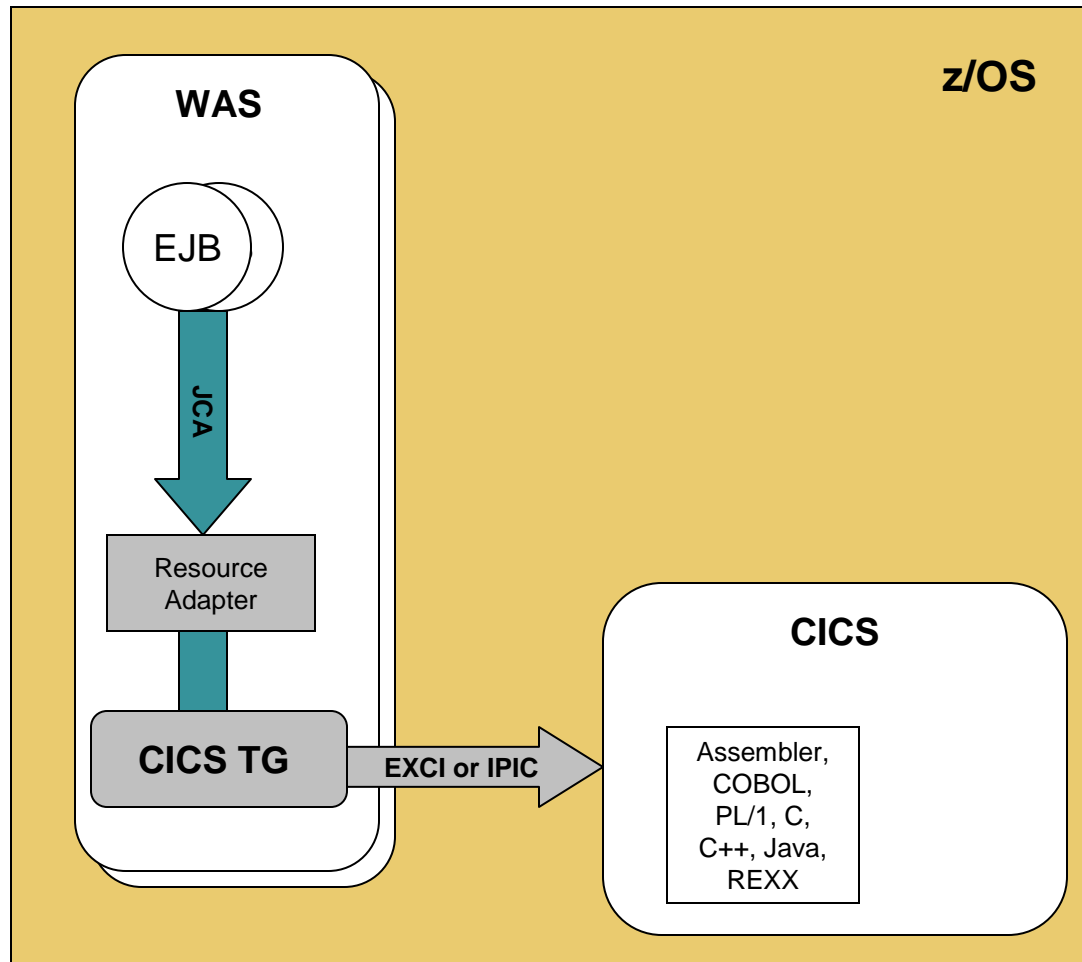
CICS TG Deployment with WAS on z/OS



Daemon Mode

- CICS TG deployed in its own address space.
- Network connection between resource adapter in WAS and CICS TG daemon
- EXCI (MRO) or IPIC connection from CICS TG daemon to CICS

CICS TG Deployment with WAS on z/OS



Local Mode

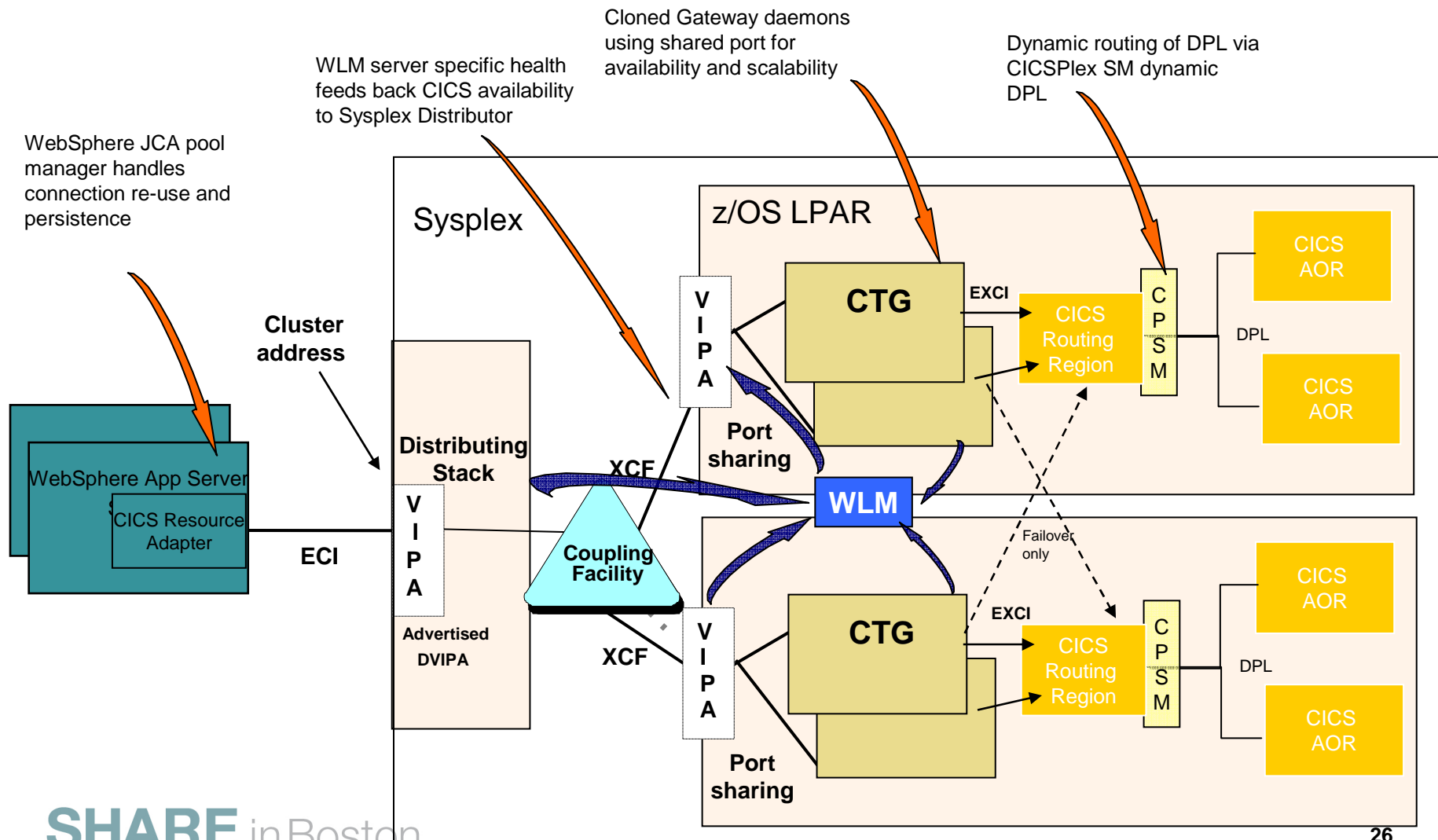
- CICS TG code deployed within WAS address spaces.
- EXCI or IPIC connection originating in WAS into CICS.

Notes

For the highest quality of service, users can run CICS Transaction Gateway on the IBM z/OS operating system. In the z/OS environment, CICS Transaction Gateway takes full advantage of the IBM Parallel Sysplex and workload management capabilities of the platform. It can support thousands of transactions per second by using multiple gateways and by exploiting memory-based external CICS interface (EXCI) pipes or TCP/IP socket connections to CICS systems that are colocated on the same logical partition (LPAR).

CICS Transaction Gateway for z/OS V7.2 provides additional XA support for global transactions from WebSphere Application Server through extending cloned gateway configurations across the Parallel Sysplex. This improved capability allows the building of a highly available connector solution—one capable of integration with the z/OS Sysplex Distributor and Resource Recovery Services. This removes the single point of failure at the LPAR level while at the same time providing for flexible naming of each CICS server supporting the highly available deployment.

CICS TG V7.2 - High availability





- Supported on all platforms
- Replacement for DFHJVSYSYSTEM_00 supporting IPIC and EXCI servers
- Supported for synconreturn and extended ECI requests (not XA)

- Mechanism to redirect ECI requests to a defined CICS server when using IP load balancing
- CICS server can be local to the LPAR or Gateway
- Supported for synconreturn and extended ECI requests (not XA)
- Two supported options:
 1. Logical server definitions
 2. CICS request exit

- Gateway cloning supported across multiple LPARs with XA transactions
- **Gateway group** defined using applid naming convention



CICS TG: features and benefits

- Simple programming model
- Access to COMMAREA and Channel applications
- Rich set of client APIs for different runtime environments
- Support for standard network protocols
- Support for different operating platforms
- Managed qualities of service and high availability
- Access to statistics and monitoring information
- Support for two-phase commit transactions from a J2EE application server

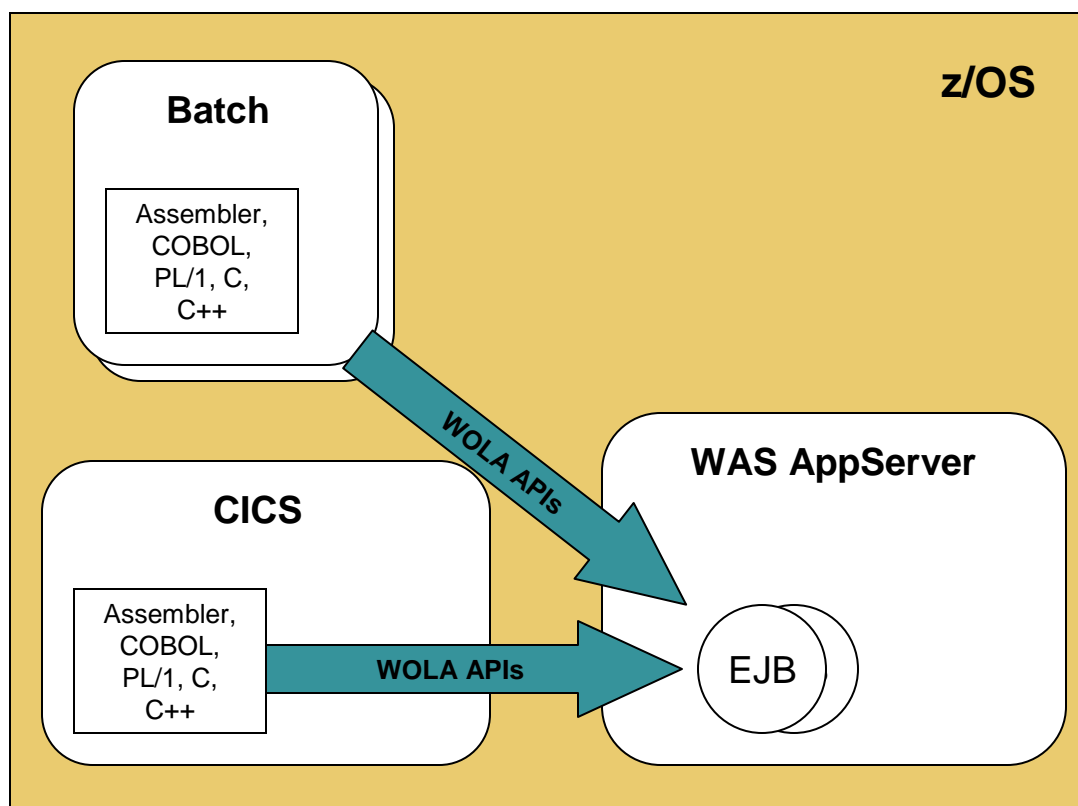
WOLA: what is it?

- New cross-memory communication structure for WAS v7
 - z/OS only
 - New API for non-Java applications
 - Inbound invocation of Java applications
 - *From batch*
 - *From USS*
 - *From CICS*
 - And a new Java Connector Architecture adapter
 - Outbound invocation of CICS applications

Notes

- The WebSphere Application Server for z/OS optimized local adapters (WOLA) is a relatively low-level communication mechanism that allows cross-address space communications from WebSphere outbound, and from external address spaces inbound to WebSphere.
- WebSphere Application Server for z/OS already has a cross-memory communication structure used internally. It makes use of the Daemon server's shared memory that allows address spaces within that cell on that LPAR to communicate cross-memory.
- Think of WOLA as an extension to that capability. The extension is the ability of external address spaces to participate just as address spaces inside the cell can.

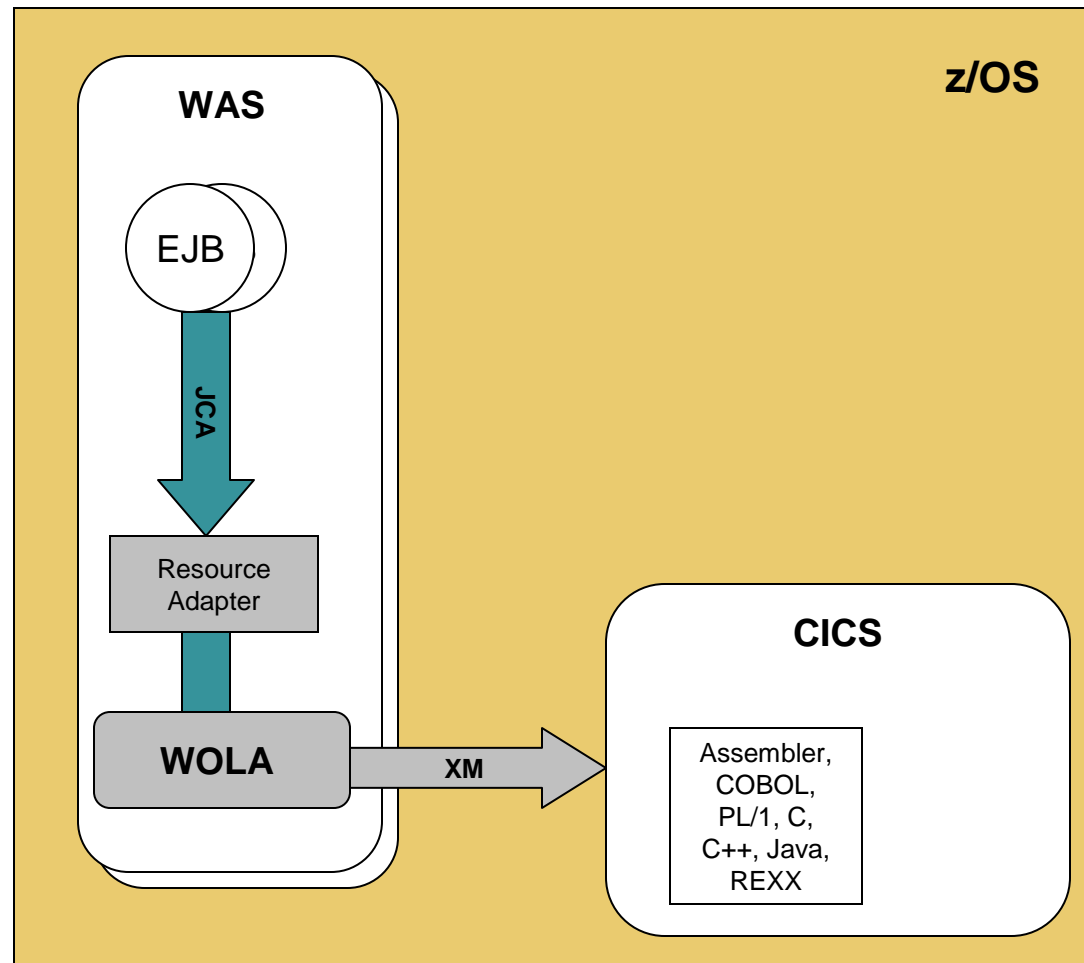
WOLA: Reuse of Java applications



Notes

- Generally speaking, when people think of WAS they think of the applications inside of WAS being users of data and services outside of WAS. But that's changing. More and more we find that people have a desire to leverage EJB assets inside of WAS as part of a broader service architecture.
- It makes sense -- WAS is a powerful Java EE runtime environment; WAS provides a wide range of key services such as security, transaction and container services. EJBs written to leverage the power of the WAS platform can in turn be leveraged by things outside of WAS.
- The issue is access to the EJBs inside of WAS.
- WOLA was created to address the specific need to access WAS z/OS EJBs in a highly efficient and high throughput manner.
- So inbound to WAS z/OS the original intent. But the developers did not wish to create a partial solution, so they made WOLA bi-directional ... both inbound to WAS z/OS and outbound from WAS z/OS.
- WOLA is not a brand new technology. It's actually an extension of an existing WAS z/OS communication mechanism called "Local Comm".
- WOLA supplies a set of modules that provide the APIs and facilities to connect to the extension of the Local Comm architecture. An external address space (batch program, CICS region, USS process) needs to have access to those modules (example, STEPLIB for batch) and it's able to make the connection.

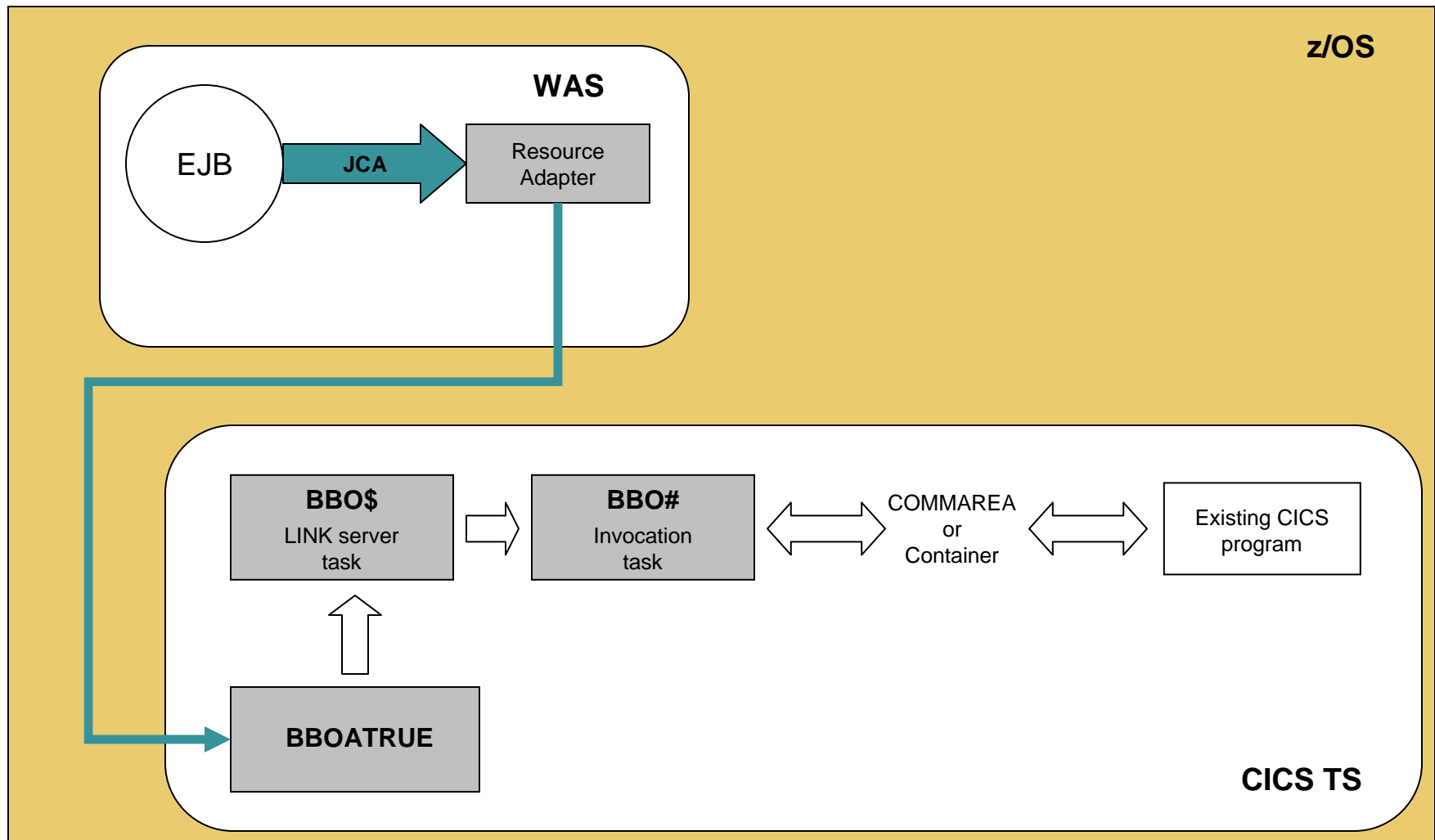
WOLA: Re-use of CICS applications



Notes

- The developers of WOLA did not wish to create a partial solution, so they made WOLA bi-directional ... both inbound to WAS z/OS and outbound from WAS z/OS.
- On the WAS side there's a new Java Connector Architecture (JCA) resource adapter supplied that provides the way for EJB applications to access WOLA. That JCA adapter installs like any other JCA adapter and can be used to invoke programs running in CICS.

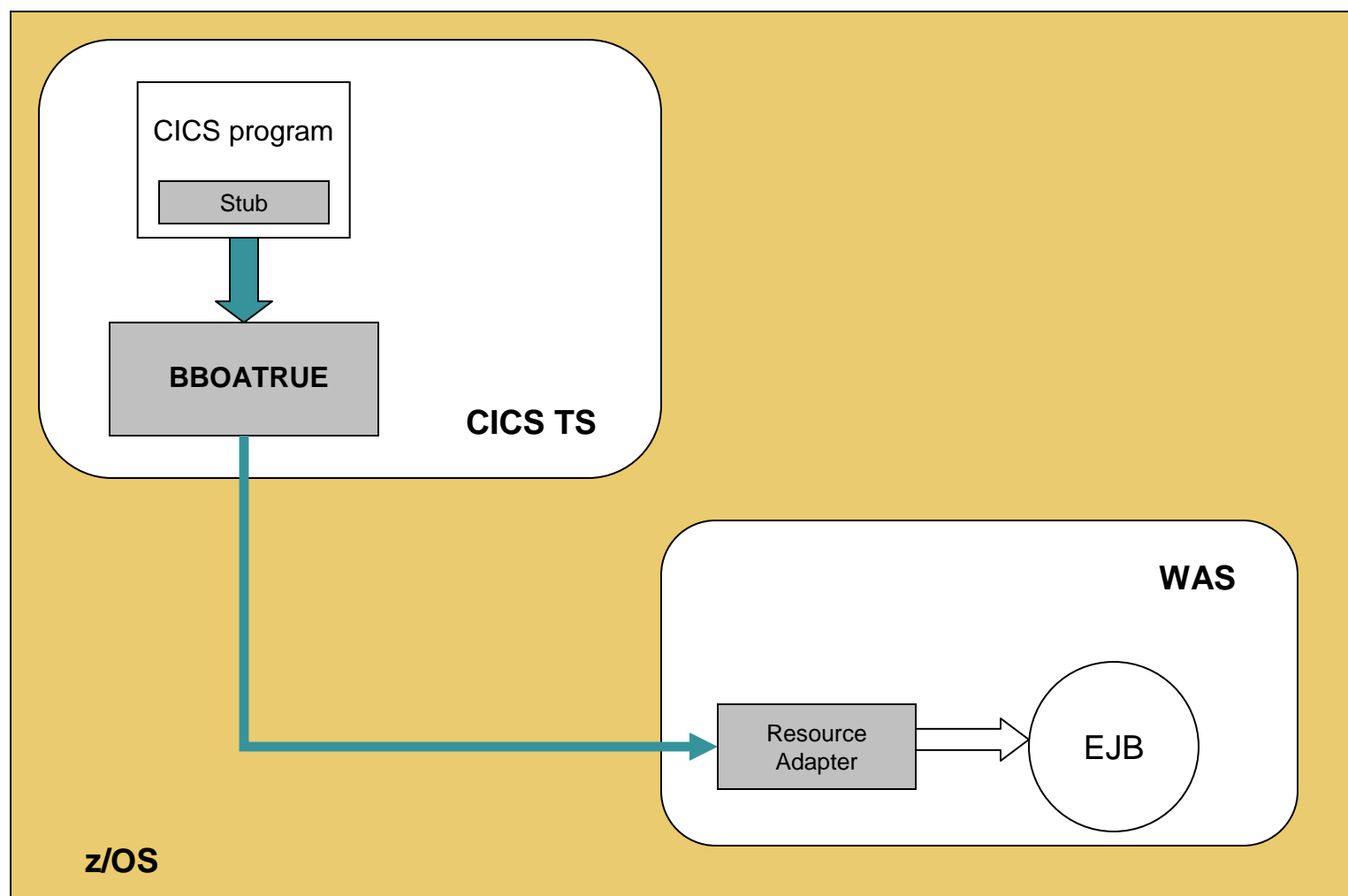
WOLA: WAS to CICS



Notes

- To enable CICS to support WOLA involves a few fairly straight forward CICS system programmer tasks, such as making the WOLA module library available to CICS via the DFHRPL DD statement, and installing the BBOACSD group into the CICS CSD.
- When you install the group, you make the WOLA programs, tasks and transactions available to CICS.
- WOLA uses a Task Related User Exit (TRUE) to implement its support in CICS. This is what provides the lower-level switching in and out of the CICS region and utilizes the Local Comm function to communicate with the WAS server.
- A 3270 control transaction called BBOC is supplied ... this allows you to manually start the TRUE.
- A LINK Server task -- BBO\$ -- receives requests inbound to CICS and maps the request to the desired program. The BBO\$ link server task is registers with the Daemon/Server in WAS and provides a pool of connections. Before any requests can be sent to CICS, the link server task must be started.
- Upon arrival of a request, BBO\$ will start an instance of transaction BBO#, which then invokes the named CICS program via LINK passing either a Commarea or a Channel.

WOLA: CICS to WAS



Notes

- From the CICS side you can call an EJB with the Invoke API call. Parameters on the call include a “service name”, which is the JNDI home interface name for the EJB.
- The WOLA invocation stub is link-edited into your CICS application.
- At runtime, the stub will utilize the TRUE to invoke the business method on the EJB.
- The enterprise bean that you want to invoke from CICS must include a method called **execute** that accepts a byte array as input and returns a byte array as output. This is the method that will receive control when an application in CICS uses one of the adapter API calls such as Invoke or Send Request.

WOLA: features and benefits











- Efficient – cross-memory communications from WAS to the external address space, or from the external address space into WAS
- Bi-directional capability – you can leverage WAS EJB assets as local services from external address spaces such as CICS or batch programs.
- Security propagation – When operating from WAS you can flow the user ID, the servant ID, or the EJB role ID into the external address space; or from the external address space you can flow the client ID or, in the case of CICS, the CICS region ID or the CICS task userid.
- Transaction propagation – When operating from CICS into WAS, WAS can participate in a CICS unit of work for two-phase commit processing; from WAS you can flow last participant into CICS.

Notes

- WOLA is bi-directional. CICS TG only supports Java applications invoking CICS applications.
- WOLA requires WAS and CICS to execute on the same LPAR, while CICS TG provides facility for Java applications running on other LPARs or even other operating systems to invoke CICS applications.
- The Java Connector Architecture adapter provided with CICS TG supports XAResource (two-phase commit) interoperability between the Java and CICS applications.
- The WOLA connector inbound to WAS provides full two-phase commit when invoked from CICS applications. The WOLA API is able to propagate a CICS transaction into WAS utilizing RRS for syncpoint coordination.
- The Java Connector Architecture interface provided with CICS TG supports an unlimited number of containers and user-defined channel names when invoking a CICS application. The WOLA interface restricts container usage to a single container within a named channel.

Functional Comparison: CICS TG and WOLA

- Java program invoking CICS application
- CICS program invoking Java application
- 2pc WAS invoking CICS
- 2pc CICS invoking WAS
- Use of channels and containers

WOLA	CICS TG
	
	
	
	
	

Learn more about WOLA

- Thursday 8am – WAS z/OS WOLA Application Designs
- White paper - *A Brief Introduction to WebSphere for z/OS Optimized Local Adapters*
 - <ftp://ftp.software.ibm.com/itsolutions/SOA/WP101490BriefIntroductionOptimizedAdapterWhitePaper.pdf>
- Redpaper REDP-4550 - *WebSphere on z/OS - Optimized Local Adapters*
 - <http://www.redbooks.ibm.com/redpapers/pdfs/redp4550.pdf>
- WOLA TechDoc Page
 - ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP101490
- Information Center - WAS v7 for z/OS
 - <http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp>

Learn more about Web services

- Redbooks
 - Application Development for CICS Web Services, SG24-7126-01
 - Implementing CICS Web Services, SG24-7657-00
 - Securing CICS Web Services, SG24-7658-00
 - Considerations for CICS Web Services Performance, SG24-7687-00
 - Integrating Back-end Systems with WebSphere Application Server on z/OS through Web Services, SG24-7548-00
 - CICS Web Services Workload Management and Availability, SG24-7144-01

Learn more about CICS TG

- Redbooks
 - Developing Connector Applications for CICS, SG24-7714-00
 - Exploring Systems Monitoring for CICS Transaction Gateway V7.1 for z/OS, SG24-7562-00
 - J2C Security on z/OS, REDP-4202-00
- Information Center – CICS TG v8 for z/OS
 - <http://publib.boulder.ibm.com/infocenter/cicstgzo/v8r0/index.jsp>
- White Papers
 - Exploiting the J2EE Connector Architecture - Integrating CICS and WebSphere Application Server using XA global Transactions
 - http://www.ibm.com/developerworks/websphere/techjournal/0607_wakelin/0607_wakelin.html
 - Integrating WebSphere Application Server and CICS using CICS Transaction Gateway
 - <ftp://ftp.software.ibm.com/software/http/cics/pdf/WSW14013-USEN-00.pdf>

Summary

- More than one option for integrating Java and CICS applications
 - Web services
 - CICS TG
 - WebSphere Optimized Local Adapters
- CICS TG and WOLA offer highest throughput when using WAS on z/OS
 - WOLA is *not* designed to replace CTG, but to *complement* it.