CICS Transaction Gateway: New Options for .NET & Java CICS Intercommunication

Leigh Compton, IBM Advanced Technical Support
lcompton@us.ibm.com

Thursday, 3 March 2011
Session 8275

Abstract

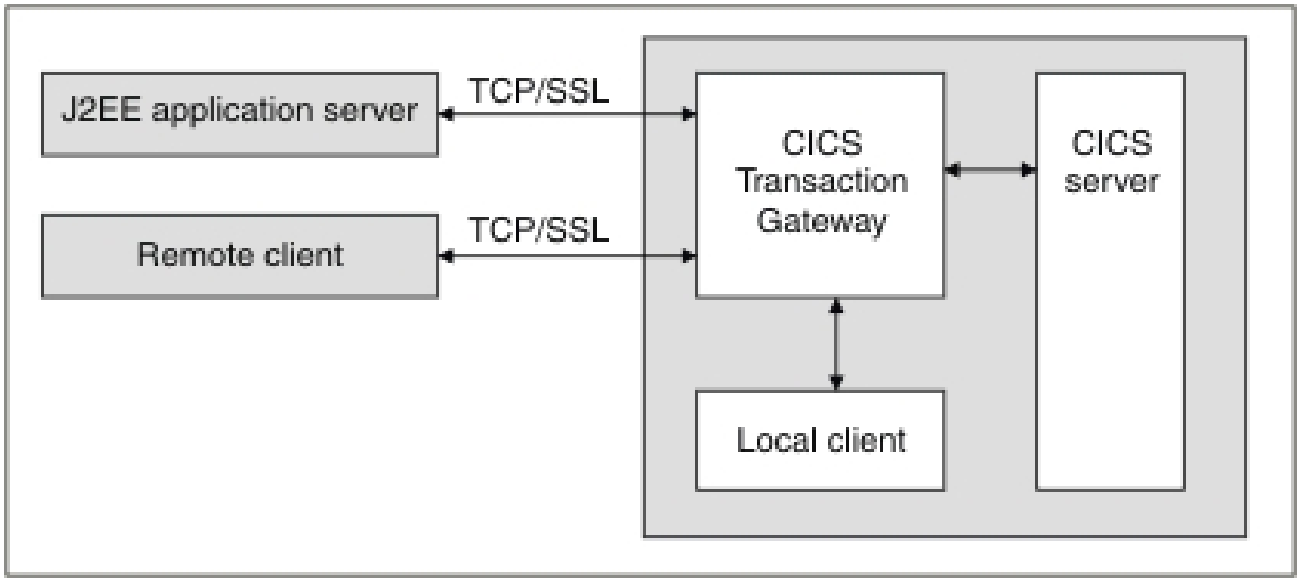
- Recent enhancements in CICS TG deliver improvements in connectivity with applications executing in Java and .NET application servers. These enhancements include:
  - Use of Channels and Containers - allowing the delivery of larger payloads
  - Remote client access from non-Java applications - supporting deployment in .NET environments.
  - Support for highly available XA configurations - improving scalability and availability
  - An API for systems monitoring - providing for integration with CICS Explorer
- The speaker will focus on how the applications executing in application server environments such as WebSphere Application Server and .NET can utilize APIs and interfaces provided by CICS TG to invoke applications which execute in CICS.

Agenda

- CICS Transaction Gateway
  - Deployment topologies
  - Application programming interfaces
- ECIv2
- Programming with .NET framework
- Programming in Java
- CICS Transaction Gateway v8.0
  - New and improved capabilities
  - High availability configurations

CICS Transaction Gateway v8.0

- High-performing, secure, and scalable connector
- Enables client applications to access CICS servers
- Flexible deployment options



Gateway - Notes

CICS Transaction Gateway is a high-performing, secure, and scalable connector that enables client applications in different runtime environments to access CICS servers.

Using standards-based interfaces, CICS Transaction Gateway delivers access to CICS applications.

CICS Transaction Gateway also provides flexible deployment options for different architectures.

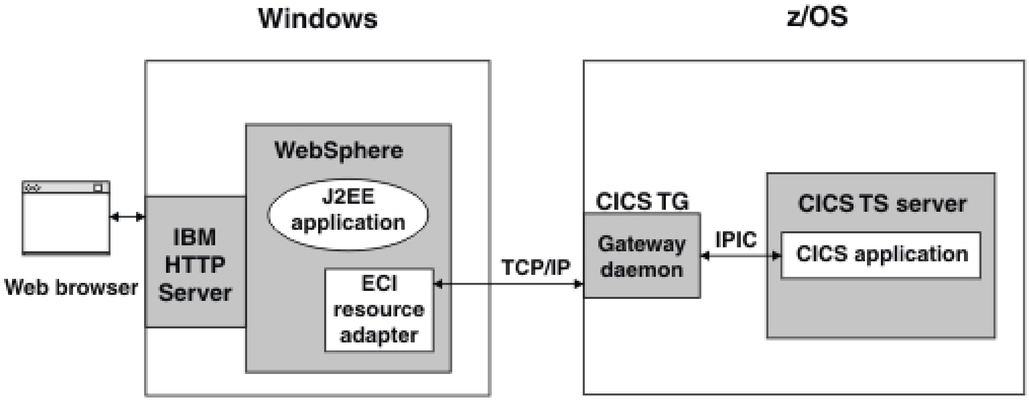
On all operating platforms, CICS Transaction Gateway provides a gateway to CICS for remote clients and also complements IBM WebSphere Application Server on a range of different platforms. In addition, CICS Transaction Gateway for z/OS is designed to exploit the qualities of service of the z/OS platform, including high availability and workload management.

CICS Transaction Gateway offers these features and benefits:

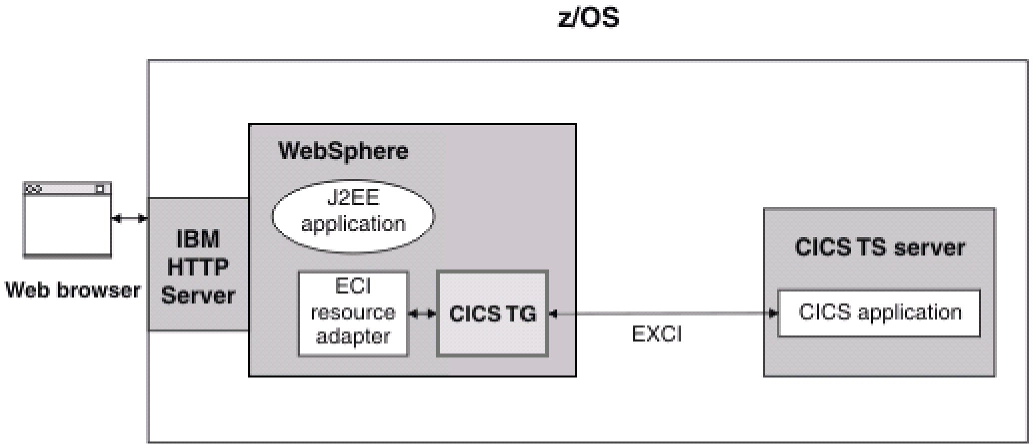
- A simple programming model with minimal change to CICS programs
- Access to COMMAREA, channel and 3270 applications
- A rich set of client APIs for different runtime environments
- Support for standard network protocols
- Support for different operating platforms
- Managed qualities of service and, on z/OS, high availability
- Access to statistics and request monitoring information
- Support for two-phase commit transactions from a J2EE application server

Deployment Topologies

- Remote mode



- Local mode



Topologies - Notes

Deployment topologies

CICS® Transaction Gateway can operate in either a two-tier configuration or a three-tier configuration. Each configuration provides different qualities of service.

Local and remote modes of operation

In a remote mode configuration, the client application and CICS Transaction Gateway are on different machines and the Gateway daemon listens on a specific port for incoming client requests. The Gateway daemon runs as a stand-alone process, handles the management of connections and threads, and forwards client requests to CICS.

In a local mode configuration, the CICS Transaction Gateway runs within the client application. Client applications send requests directly to CICS without using a Gateway daemon.

Prior to CICS TG v7.2, remote mode was only available to client applications coded in Java. ECIv2 extends remote mode topologies to clients coded in other languages through a set of C language bindings.

Features of remote mode

Remote mode is best suited to large-scale systems and has the following features:

- A common point of access to CICS for a variety of application types and operating systems
- A common point of configuration and administration for connections to CICS
- High availability with workload balancing across multiple CICS servers
- A lightweight client footprint
- Access to statistical information

Features of local mode

Local mode is best suited to environments where a small number of J2EE application servers are connected to CICS, and has the following features:

- Fewer components to manage than in remote mode
- Network topology is simplified

Client Applications

- Access one or more CICS servers
- Connect to several CICS servers at the same time
- Make concurrent program calls
- Remote mode clients
  - Connect to gateway daemon
  - Java, JCA
    - ECI and EPI
  - Remote C clients
    - ECI (ECIv2) only
- Local mode clients
  - Java, JCA
  - C, C++, COBOL, VB
    - ECI, EPI, and ESI

Clients - Notes

The CICS Transaction Gateway supports running Client applications in both local and remote modes. Client applications enable access to CICS server transactions and programs from the host machine.

In local mode, the Client application is installed and run on the CICS Transaction Gateway host machine. In remote mode, the Client application is installed and run on a machine remote to the CICS Transaction Gateway host machine.

Client applications have the following capabilities in common:

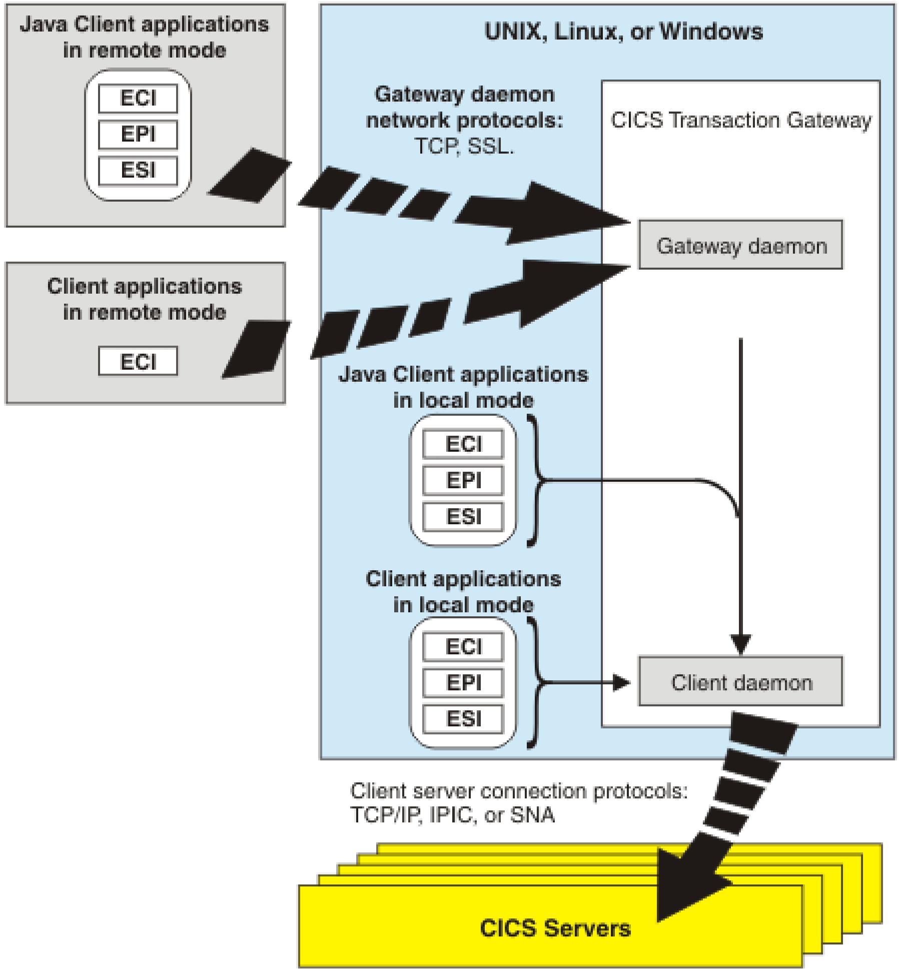
- They can be written to access one or more CICS servers.
- They can connect to several CICS servers at the same time.
- They can have several program calls running concurrently.

Java Client applications use the Gateway classes to communicate with CICS servers. JCA applications use the J2EE CICS resource adapters to communicate with CICS servers.

Non-Java Client applications running in local mode use the Client API to communicate with CICS servers. Non-Java Client applications running in remote mode use the ECI version 2 C language bindings to communicate with CICS servers.

CICS TG for Multiplatforms

- Java and ECIv2 remote clients
- Java and non-Java applications in local mode
    - ECI
    - EPI
    - ESI


Java Client applications in remote mode
ECI
EPI
ESI
Client applications in remote mode
ECI
UNIX, Linux, or Windows
Gateway daemon network protocols: TCP, SSL.
CICS Transaction Gateway
Gateway daemon
Java Client applications in local mode
ECI
EPI
ESI
Client applications in local mode
ECI
EPI
ESI
Client daemon
Client server connection protocols: TCP/IP, IPIC, or SNA
CICS Servers

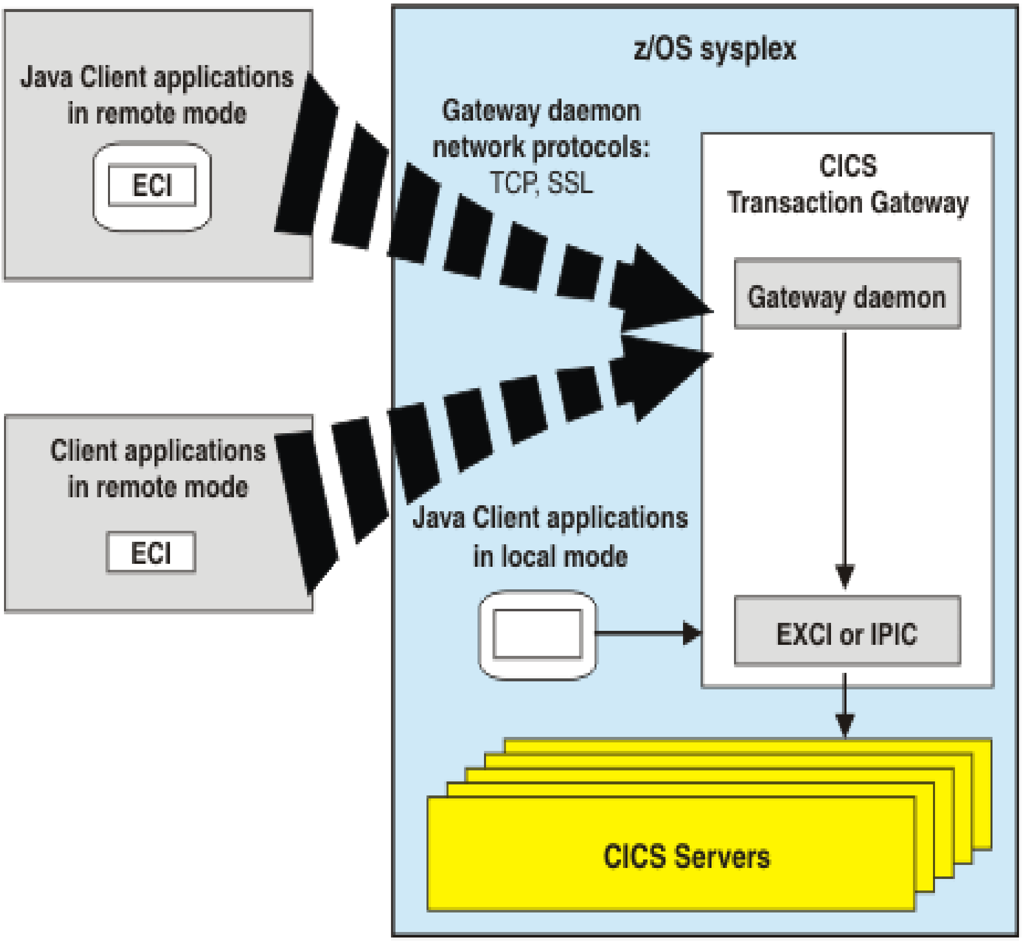
CICS TG for Multiplatform - Notes

CICS TG on Multiplatforms is supported on the following range of operating systems and platforms and is designed to support connectivity to all in-service CICS servers.

- Linux on System z
- Linux on Intel
- Linux on POWER
- AIX
- HP-UX (on PA-RISC and Itanium)
- Sun Solaris (on SPARC)
- Windows XP, Windows 2000, Windows 2003 and Windows Vista

CICS TG for z/OS

- ECI calls only
- Java and ECIv2 remote clients
- Java applications in local mode


Java Client applications in remote mode
ECI
Client applications in remote mode
ECI
z/OS sysplex
Gateway daemon network protocols: TCP, SSL
CICS Transaction Gateway
Gateway daemon
Java Client applications in local mode
EXCI or IPIC
CICS Servers

CICS TG for z/OS - Notes

CICS TG on z/OS v8 is supported on z/OS V1R9 and later and supports connectivity to all in-service releases of CICS Transaction Server for z/OS.

The CICS TG on z/OS uses either the external communication interface (EXCI) or the IP Interconnectivity protocol (IPIC) to communicate with CICS regions on z/OS. It doesn’t include the Client daemon and does not provide any support for non-Java based applications except for ECIV2 based C applications as remote client application.

Application programming interfaces

- External Call Interface (ECI)
  - Equivalent to CICS LINK command (DPL)
  - Using COMMAREA or channels and containers
- External Presentation Interface (EPI)
  - Access to 3270-based transactions
  - EPI program acts as terminal operator
- External Security Interface (ESI)
  - Access to password management functions
  - Uses SNA Password Expiration Management (PEM) functions

API - Notes

CICS Transaction Gateway provides a rich set of programming interfaces that provide access to CICS COMMAREA, channel, and 3270 applications. CICS Transaction Gateway provides APIs for Java, C, C++, and COBOL. JCA resource adaptors and COM objects are also provided.

These programming interfaces provided by CICS Transaction Gateway are the External Call Interface (ECI), the External Presentation Interface (EPI), and the External Security Interface (ESI).

An ECI request calls a CICS® program on a CICS server. If no CICS server is selected, the default CICS server is used.

Program link calls cause the CICS mirror transaction to be attached to run a server program on the CICS server.

ECI request program link calls can be synchronous or asynchronous: Synchronous and asynchronous program link calls can be nonextended or extended:

ECI-based communications between the CICS server and the CICS Transaction Gateway are known as conversations. A nonextended program link ECI call is one conversation. A series of extended ECI calls followed by a commit or rollback is one conversation.

EPI allows a user application program to access 3270–based transactions on one or more CICS servers. The user application can establish one or more resources and act as the operator, starting 3270-based CICS transactions and sending and receiving data associated with those transactions.

ESI provides a security management API which can be used to manage the user IDs and passwords that the ECI and EPI use. To use the ESI interface, the CICS Transaction Gateway must be connected to the CICS server with SNA. An ESM, such as Resource Access Control Facility (RACF), which is part of the z/OS Security Server, or an equivalent ESM, must also be available to the CICS server.

External Call Interface (ECI)

- Calls CICS program
  - Equivalent to LINK/DPL
  - Runs under mirror transaction
- Commarea
  - Input and output in same storage
  - Maximum size 32K
- Containers (passed in channel)
  - Unlimited in size
  - Any number of containers passed in channel
  - Supports passing data in structured manner

ECI - Notes

The External Call Interface (ECI) enables a client application to call a CICS® program synchronously or asynchronously. It enables the design of new applications to be optimized for client/server operation, with the business logic on the server and the presentation logic on the client.

The external interfaces allow non-CICS applications to access and update CICS resources by calling CICS programs. When used in conjunction with CICS communication, the external interfaces enable non-CICS programs to access and update resources on any CICS system. This method of using the external interfaces supports such activities as the development of graphical user interface (GUI) front ends for CICS applications and it allows the integration of CICS systems and non-CICS systems.

The application can connect to several CICS servers at the same time and have several called CICS programs running concurrently. The CICS programs can transfer information using COMMAREAs or channels.

CICS programs that are invoked by an ECI request must follow the rules for distributed program link (DPL) requests.

Channels and containers provide a method of transferring data between CICS® programs, in amounts that far exceed the 32 KB limit that applies to communication areas (COMMAREAs).

Each container is a "named COMMAREA" that is not limited to 32 KB. Containers are grouped together in sets called channels.

The channel and container model has several advantages over the communication areas (COMMAREAs) used by CICS programs to exchange data:

- Unlike COMMAREAs, channels are not limited in size. Any number of containers can be added to a channel, and the size of individual containers is limited only by the amount of storage that you have available. Consider the amount of storage available to other applications when you create large containers.
- Because a channel can consist of multiple containers, it can be used to pass data in a more structured way, allowing you to partition your data into logical entities. In contrast, a COMMAREA is a monolithic block of data.
- Unlike COMMAREAs, channels do not require that the programs that use them to record and keep track of the exact size of the data returned.
- Channels can be used by CICS application programs written in any of the CICS-supported languages. For example, a Java™ client program on one CICS server can use a channel to exchange data with a COBOL server program on a back-end AOR.
- CICS automatically destroys containers and their storage when they go out of scope.

ECIv2

- C language support for remote clients
- Can be used by server-based or desktop applications
  - New alternative to CICS Universal Clients
  - Minor changes to existing ECI application programs to use new ECIv2 function
- Introduced in CICS TG v7.2
  - Commarea-only
- Enhanced in CICS TG v8.0
  - Channels and containers supported

ECIv2 - Notes

C language support for remote clients

A new type of remote client support introduced in CICS TG v7.2 provides support for C-language ECI applications to access CICS servers from multiplatform machines remote to the Gateway daemon. Such an application must connect to a Gateway daemon over TCP/IP before flowing an ECI request to the target CICS server.

A set of C language bindings provides lightweight client support for ECI Version 2 requests from remote client environments. In v7.2, ECIv2 provides access to CICS COMMAREA based applications. CICS TG v8 extends ECIv2 to add support for channels and containers for remote C clients.

This support provides for potential usage in a wide variety of client runtime environments including C++, COBOL, and .NET. Integration with the existing Gateway function provides support for one-phase commit transactions, password authentication, and request timeout

The ECI version 2 interface is for the C-language and it does not contain any C++ classes

External Presentation Interface (EPI)

- Access to 3270-based transactions
- Terminal-oriented applications
  - Non-conversational
  - Conversational
  - Pseudo-conversational
- 3270 programming model
  - handle the 3270 data stream as a simple data record
- Screen programming model
  - handle the 3270 data based on the structure of the fields in the 3270 data stream
  - can import BMS map data to help with this process

EPI - Notes

EPI allows a user application program to access 3270–based transactions on one or more CICS servers. The user application can establish one or more resources and act as the operator, starting 3270-based CICS transactions and sending and receiving data associated with those transactions.

EPI functions can be used to add terminals to CICS and delete them when they are no longer required. The user application that installs a terminal has exclusive use of that terminal until the terminal is deleted.

When a user application has added a terminal to a CICS server, the application can start a transaction from that terminal. To the CICS server it appears as if an operator has entered a transaction name at a terminal.

When a transaction is running on CICS, data is passed between CICS and the user application.

This might be data produced by the transaction or one or more messages from the CICS server, for example terminal error messages. If the data is in the form of BMS map data, CICS also supplies the map name and map set name. If the map is to be returned to CICS for further processing, the user application must also return the map name and map set name.

A *conversational transaction* is one which processes several sets of input from a terminal before returning control to CICS.

The length of time required for a response from a terminal is much longer than the time taken to process it, therefore a conversational transaction lasts much longer than a *nonconversational transaction,* which processes one set of input before relinquishing control. While a transaction is running it is using storage and resources which might be needed by other transactions. For this reason many CICS transactions operate in pseudoconversational mode.

A *pseudoconversational transaction* is one in which the conversation between a terminal and a server is broken up into a number of segments, each of which is a nonconversational transaction. As each transaction ends, it provides the name of the transaction to be run to process the next input from the terminal. When a transaction that has just ended specifies the name of a transaction to process the next input, this name is passed to the user application. The application must not attempt to start a different transaction, but must use the returned information to start the specified transaction and send the data it is expecting.

There are two different programming models for EPI-based applications:

- The screen model allows the user application to handle the 3270 data based on the structure of the fields in the 3270 data stream. In some languages it is also possible to import BMS map data to help with this process.
- With the 3270 model, the user application reads the 3270 data stream as a simple data record and is responsible for parsing the information that it contains.

External Security Interface - ESI

- Verify Password
- Change Password
- Determine if userID is revoked
- Determine if password is expired
- Obtain information about verified user
  - When password is due to expire
  - When userID last accessed
  - Date and time of current verification
  - Number of unauthorized attempts for this user since last valid access
- Specify default userID and password
- Note: to use ESI, CICS TG must connect to CICS with SNA.

ESI - Notes

ESI provides a security management API which can be used to manage the user IDs and passwords that the ECI and EPI use.

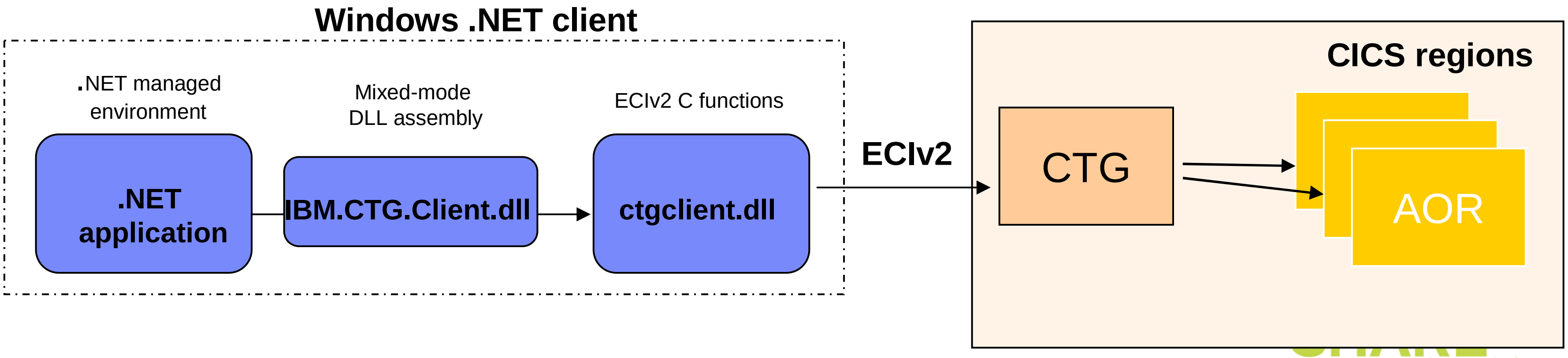
The user application can perform the following functions:

- •Verify that a password matches the password recorded by the CICS External Security Manager (ESM) for a specified user ID.
- •Change the password recorded by the CICS ESM for a specified user ID.
- •Determine if a user ID is revoked, or a password has expired.
- •Obtain additional information about a verified user such as:
  - •When the password is due to expire
  - •When the user ID was last accessed
  - •The date and time of the current verification
  - •How many unauthorized attempts there have been for this user since the last valid access
- •Specify a default user ID and password to be used for communication over a CICS server connection.

To use the ESI interface, the CICS Transaction Gateway must be connected to the CICS server with SNA. An ESM, such as Resource Access Control Facility (RACF), which is part of the z/OS Security Server, or an equivalent ESM, must also be available to the CICS server.

Programming with .NET Framework

- .NET API for ECI calls to CICS
- CTG ECI v2 APIs can be used in .NET managed runtime
  - C#, C++ or VB.NET applications
- Remote clients access CICS servers using ECI requests
  - Commarea or Channel/Containers
- Replaces CA72 and CA73 SupportPacs



.NET Framework - Notes

CICS TG V8 delivers a new .NET API that allows remote clients to access CICS servers using COMMAREA-based ECI requests. ECI requests with channels and containers are not supported.

The .NET framework offers a number of advantages when developing remote client applications.

•A consistent model, provided by the .NET class library, for all supported programming languages.

•High levels of security for applications used in remote mode topologies; method-level security using industry standard security technologies can be explicitly defined.

•Separation of application logic from presentation logic for easier maintenance and upgrade.

•Simplified debugging plus the availability of runtime diagnostics.

•Simpler application deployment.

.NET programming interface

- GatewayConnection class
  - represents a connection to CICS Transaction Gateway
  - opened in the constructor
  - remains open until the Close() method is invoked
  - two methods
    - Flow(request)
    - ListSystems()
- GatewayRequest class
  - base class for other request types
- EciRequest
- Trace class

.NET programming - Notes

.NET classes are supported on all Windows platforms.

•The GatewayConnection class represents a connection to CICS Transaction Gateway. The connection is opened in the constructor and remains open until the Close() method is invoked. The class provides two methods for interacting with CICS Transaction Gateway: Flow(request) which flows an EciRequest to CICS Transaction Gateway, and ListSystems() which returns a list of all CICS servers that have been defined in CICS Transaction Gateway.

•The GatewayRequest class provides a base class for other request types, as with the Java API.

•The EciRequest represents an ECI call to CICS and provides ECI fields such as extend mode, LUW token, and COMMAREA. Channels and containers are not supported.

•The Trace class provide static methods for interacting with the tracing features of the C API.

The .NET framework includes a RequestMinimum security attribute to execute unmanaged code (UnmanagedCode=true).

Tracing can be dynamically enabled and disabled in .NET applications, if the application has been compiled with a /d:TRACE command line switch. If compiled with this switch, tracing can then be enabled or disabled by using an application configuration file.

.NET Programming Considerations

- Threading restrictions
  - GatewayConnection instance used only by thread that creates it
  - Attempts to use by other threads causes exception to be thrown
- 64-bit considerations
  - API not supported running in 64-bit mode
  - Must insure application runs in 32-bit mode
    - Compilation targets x86 platform

.NET considerations - Notes

Threading restrictions

GatewayConnection is the .NET class which represents a TCP/IP connection to a CICS Transaction Gateway.

A GatewayConnection instance can only be used by the thread that creates it. Attempts to use the GatewayConnection from other threads causes an InvalidOperationException to be thrown. This includes flowing requests, listing the servers defined in the Gateway daemon, and closing the connection.

64 bit considerations using the .NET framework

The CICS Transaction Gateway API for .NET does not support running in 64 bit mode. By default, the Common Language Runtime launches .NET applications as 64 bit processes when running on 64 bit operating systems.

To ensure that an application always runs in 32 bit mode, the application must be configured to target the x86 architecture. This can be achieved in the following ways:

•When you compile an application using Microsoft® Visual Studio, set the Platform target project property to x86.

•When you compile an application from a command prompt or script, use the corflags utility which is provided as part of the Microsoft .NET Framework SDK. To modify an application to target the x86 architecture, use the following command:

```
corflags.exe <executable_name> /32BIT+
```

Programming in Java

- Java Base Classes
  - ECI, EPI, and ESI
- Java Support Classes
  - EPI
- J2EE Connector Architecture
  - ECI and EPI

Java Programming - Notes

The CICS Transaction Gateway enables Java Client applications to communicate with programs on a CICS server by providing base classes for the External Call Interface (ECI) and the External Security Interface (ESI), and EPI support classes for the External Presentation Interface (EPI). Note that the EPI classes are not available with the CICS Transaction Gateway for z/OS.

The purpose of the J2EE Connector Architecture (JCA) is to connect Enterprise Information Systems (EISs), such as CICS®, into the J2EE platform. The JCA offers a number of qualities of service which can be provided by a J2EE application server. These qualities of service include security credential management, connection pooling and transaction management.

These qualities of service are provided by means of system level contracts between a resource adapter provided by the CICS Transaction Gateway, and the J2EE application server. There is no need for any extra program code to be provided by the user. Thus the programmer is free to concentrate on writing the business code and need not be concerned with providing quality of service.

The JCA defines a programming interface called the Common Client Interface (CCI). This interface can be used, with minor changes, to communicate with any EIS. The CICS Transaction Gateway provides resource adapters which implement the CCI for interactions with CICS.

Java Base and Support Classes

- JavaGateway
  - Logical connection to CICS server
  - flow() method passes data to CICS server
- ECIRequest
  - Contains details of ECI (LINK) request to a CICS server
- EPIRequest
  - Base class retained for compatibility
- EPIGateway
  - Support class, recommended to use
- Terminal
  - Controls a 3270 connection to CICS
- ESIRequest
  - Contains details of an ESI request to a CICS server

Base & Support classes - Notes

The CICS Transaction Gateway enables Java Client applications to communicate with programs on a CICS server by providing base classes for the External Call Interface (ECI) and the External Security Interface (ESI), and EPI support classes for the External Presentation Interface (EPI).

The following list of classes are the basic classes provided with the CICS Transaction Gateway. For a full description of all the classes and methods referred to in this section, refer to the Javadoc supplied with the CICS Transaction Gateway.

Note that the EPI classes are not available with the CICS Transaction Gateway for z/OS.

com.ibm.ctg.client.JavaGateway

- This class is the logical connection between a program and a CICS Transaction Gateway. You need a JavaGateway object for each CICS Transaction Gateway that you want to talk to.

com.ibm.ctg.client.ECIRequest

- This class contains the details of an ECI request to the CICS Transaction Gateway.

com.ibm.ctg.epi.Terminal

- This class controls a 3270 terminal connection to CICS. The Terminal class handles CICS conversational, pseudoconversational, and ATI transactions. A single application can create many Terminal objects.

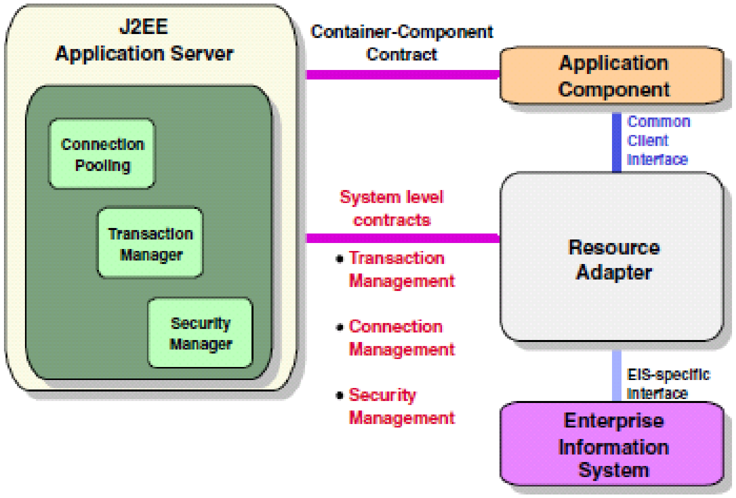
com.ibm.ctg.client.ESIRequest

- This class contains the details of an ESI request to the CICS Transaction Gateway.

Note: The **com.ibm.ctg.client.EPIRequest** base class is supported only for compatibility with earlier releases of the product. New programs should use the EPI support classes.

J2EE Connector Architecture

- Standard Components and API
- Common Client Interface
- Resource Adapter
  - Managed environment
  - Non-managed environment



JCA - Notes

The purpose of the J2EE Connector Architecture (JCA) is to connect Enterprise Information Systems (EISs), such as CICS, into the J2EE platform. The JCA offers a number of qualities of service which can be provided by a J2EE application server. These qualities of service include security credential management, connection pooling and transaction management.

These qualities of service are provided by means of system level contracts between a resource adapter provided by the CICS Transaction Gateway, and the J2EE application server. There is no need for any extra program code to be provided by the user. Thus the programmer is free to concentrate on writing the business code and need not be concerned with providing quality of service.

The JCA defines a programming interface called the Common Client Interface (CCI). This interface can be used, with minor changes, to communicate with any EIS. The CICS Transaction Gateway provides resource adapters which implement the CCI for interactions with CICS.

Applications using the CCI have a common structure, independent of the EIS that is being used. The JCA defines Connections and ConnectionFactories which represent the connection to the EIS. These objects allow a J2EE application server to manage security, transaction context, and connection pools for the resource adapter.

Common Client Interface

- Standard client API
  - Independent of specific EIS
  - Consistent with other J2EE APIs (JDBC, for example)
  - Targeted toward AD tools and integration frameworks
- Framework classes
  - Connection object and ConnectionFactory
    - Connect to resource adapter and disconnect
    - Specify the interaction properties or characteristics
    - Execute the interaction
- Input and output classes
  - Pass or receive data to/from framework classes
    - ConnectionSpec objects
    - InteractionSpec objects
  - Pass or receive data to/from EIS
    - Record objects

CCI - Notes

The Common Client Interface (CCI) of the J2EE Connector Architecture provides a standard interface that allows developers to communicate with any number of Enterprise Information Systems (EISs) through their specific resource adapters, using a generic programming style.

The CCI is closely modeled on the client interface used by Java™ Database Connectivity (JDBC), and is similar in its idea of Connections and Interactions.

The generic CCI classes define the environment in which a J2EE component can send and receive data from an EIS.

When you are developing a J2EE component you must implement the following steps:

1. Use the ConnectionFactory object to create a Connection object.
2. Use the Connection object to create an Interaction object.
3. Use the Interaction object to execute commands on the EIS.
4. Close the Interaction and Connection.

The CICS Transaction Gateway resource adapters provide additional classes specific to CICS. The following object types are used to define the ECI- and EPI-specific properties:

- InteractionSpec objects
- ConnectionSpec objects

Spec objects define the action that a resource adapter carries out, for example by specifying the name of a program which is to be executed on CICS®.

Record objects store the input/output data that is used during an interaction with an EIS, for example a byte array representing an ECI COMMAREA.

Resource Adapters

- ECI resource adapters
  - cicseciXA.rar
    - Supports both global transactions (XATransaction) and local transactions (LocalTransaction)
  - cicseci.rar
    - Supports only non-transactional or LocalTransaction
- EPI resource adapter
  - cicsepi.rar
    - Supplied only with CICS TG for Multiplatforms

Resource Adapters - Notes

Two resource adapters are supplied, one provides LocalTransaction support and the other both LocalTransaction and XA Transaction support.

cicseci.rar

This resource adapter provides LocalTransaction support when deployed on any supported J2EE application server. Local transactions are not supported when using WebSphere Application Server for z/OS with CICS Transaction Gateway on z/OS in local mode, because the resource adapter and MVS RRS provide global transaction support.

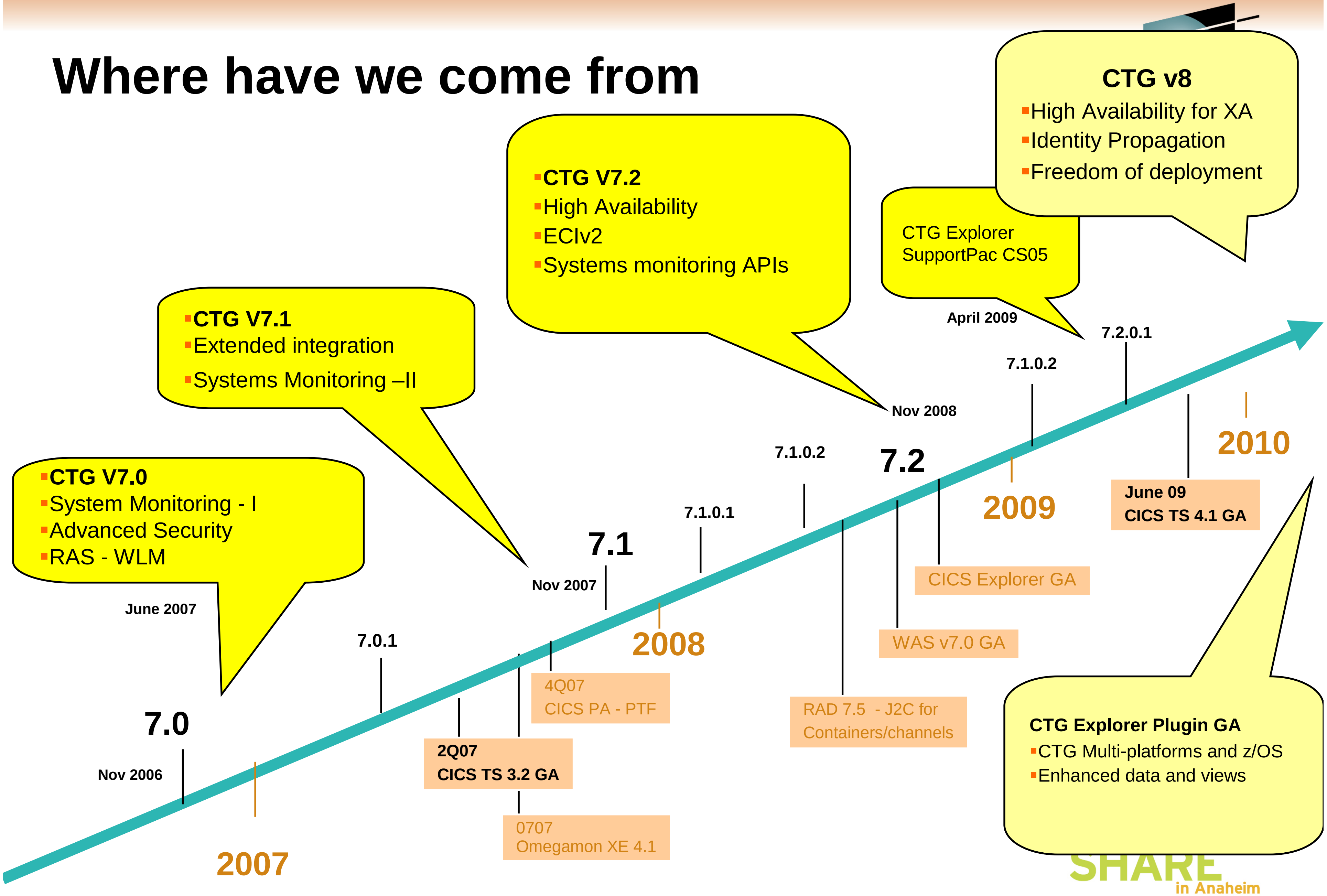
cicseciXA.rar

This resource adapter provides both XATransaction and LocalTransaction support when deployed on any supported J2EE application server connecting to a remote CICS Transaction Gateway for z/OS. It also provides global transaction support when using WebSphere Application Server for z/OS with CICS Transaction Gateway on z/OS in local mode.

With the CICS EPI resource adapter a J2EE component can communicate with CICS transactions that use 3270 data streams for input and output.

The resource adapter provides access to the CICS 3270 interface because each EPIConnection object is treated as a 3270 terminal by CICS.

Where have we come from



- **CTG V7.0**
- System Monitoring - I
- Advanced Security
- RAS - WLM

June 2007

7.0

Nov 2006

2007

7.0.1

2Q07
CICS TS 3.2 GA

0707
Omegamon XE 4.1

4Q07
CICS PA - PTF

- **CTG V7.1**
- Extended integration
- Systems Monitoring –II

7.1

Nov 2007

2008

7.1.0.1

7.1.0.2

RAD 7.5 - J2C for Containers/channels

- **CTG V7.2**
- High Availability
- ECIv2
- Systems monitoring APIs

Nov 2008

7.2

WAS v7.0 GA

CICS Explorer GA

2009

7.1.0.2

CTG Explorer SupportPac CS05

April 2009

7.2.0.1

June 09
CICS TS 4.1 GA

2010

CTG v8

- High Availability for XA
- Identity Propagation
- Freedom of deployment

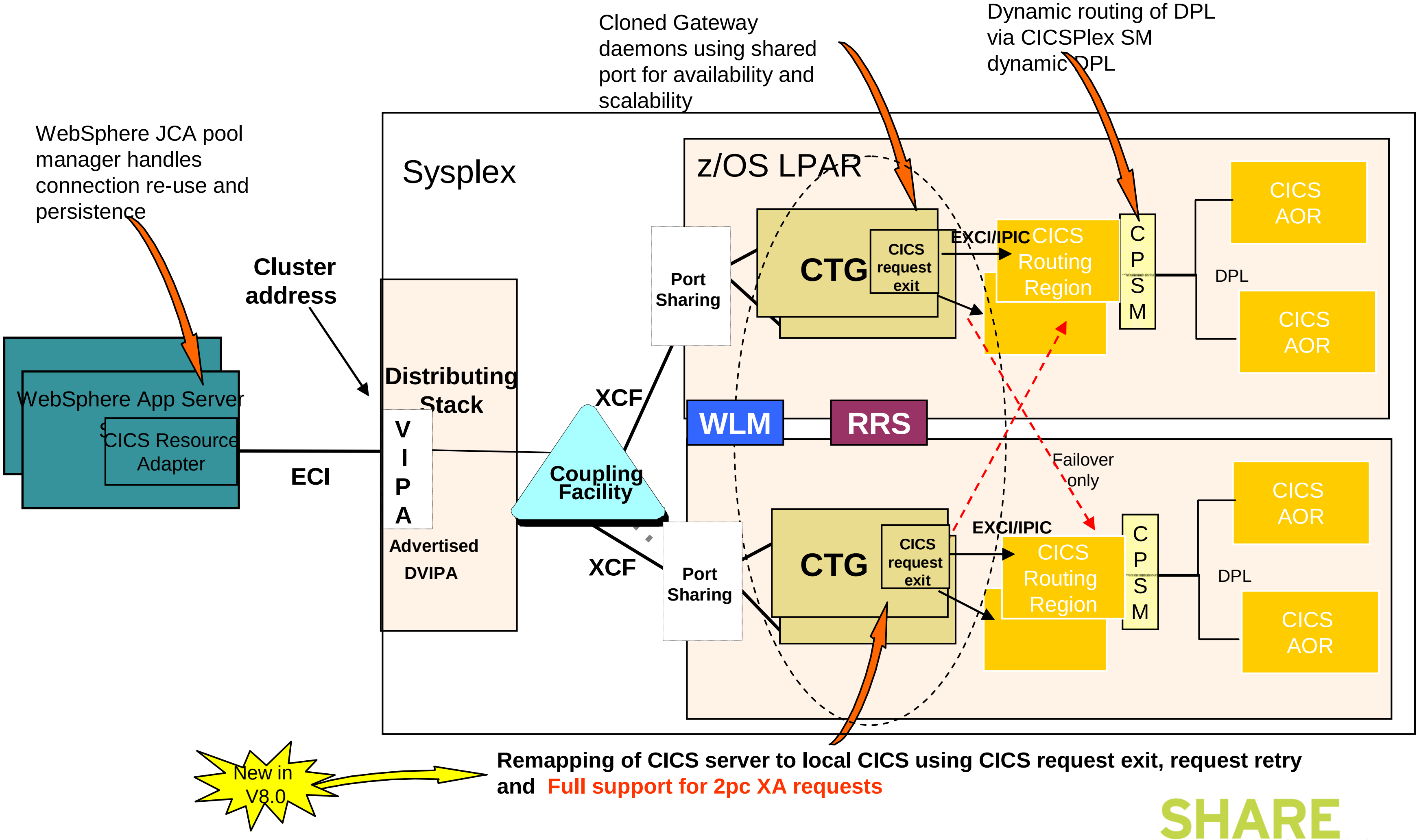
CTG Explorer Plugin GA

- CTG Multi-platforms and z/OS
- Enhanced data and views

Open Integration

- J2EE Install Verification Test (IVT) program
  - Entitles customers to full CICS TG support from IBM; provided their J2EE 1.4 application server passes the IVT
  - Extends interoperability of CICS TG to a wider range of J2EE platforms
  - Productized version of SupportPac CH91
- Local mode IPIC connectivity from 64-bit J2EE Application Servers
  - 64-bit capable CTG for Multiplatforms in local mode
  - Avoids the need to have an intermediate 32-bit Gateway daemon
- New platform support including
  - Windows 2008 Server (including R2)
  - Windows 7
- Java 6
  - Java 6 is required level support for the Gateway daemon in the v8.0 GA release
  - API JARs and Resource adapters remain compatible with Java 5 & Java6 SDKs
  - Java 6 provides numerous enhancements
    - Performance benefit with improved garbage collection
    - Shared class cache can reduce Gateway daemon memory requirements and improve startup

CICS TG z/OS High Availability



WebSphere JCA pool manager handles connection re-use and persistence

Cloned Gateway daemons using shared port for availability and scalability

Dynamic routing of DPL via CICSPlex SM dynamic DPL

Remapping of CICS server to local CICS using CICS request exit, request retry and Full support for 2pc XA requests

CICS request exit – CA1T V2

- High availability user exit:
  - Deployed inside Gateway daemon and invoked for all types of ECI requests
  - Integrated with Gateway HA/XA support to provide transactional integrity for extended UOW and XA requests
  - Supersedes function of EXCI user exit – DFHXCURM
  - Provided as category 2 SupportPac
  - Configured using simple HFS or MVS configuration file – no coding required - source code supplied
- Functions:

1. Remap server name
2. Workload balance requests across defined CICS servers

   Round robin or fail over policies can be used
3. Dynamically retry failed requests

   Stores state of failed servers to prevent dead servers being used
4. Validate requests

   based on: Userid, Tranid, Program, Payload type (container/commarea)

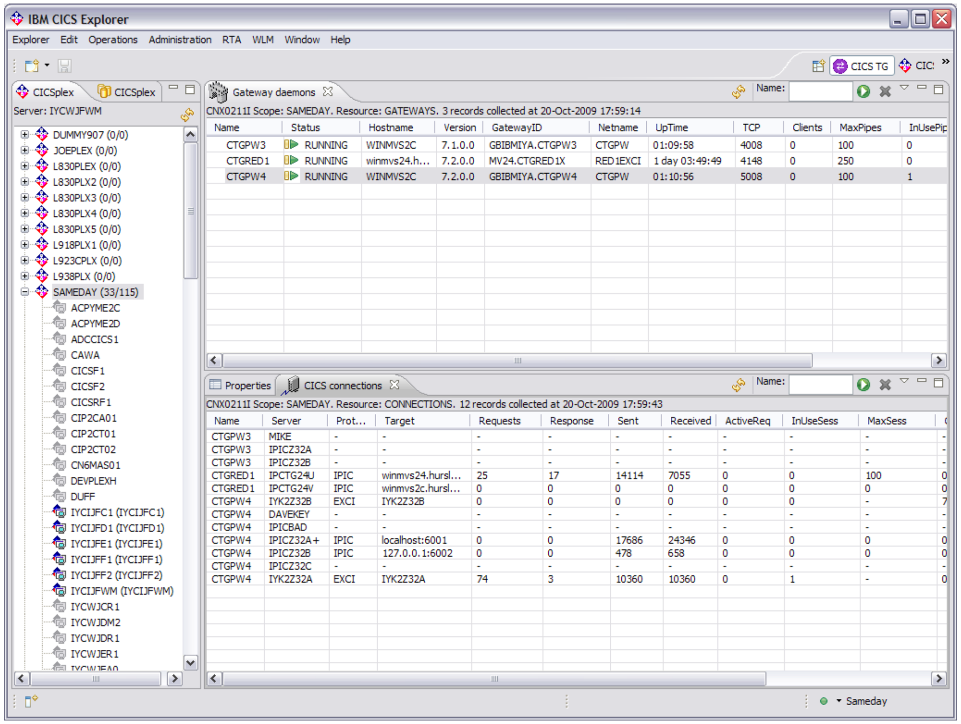
   ECI_ERR_INVALIDCALLTYPE (-14) will be returned on failure
5. Dynamic command support

   Dynamically reconfigure, trace, and enable/disable exit

CICS Explorer – The New Face of CICS

SHARE
Technology • Connections • Results

The CICS Explorer provides an intuitive, easy-to-use way of managing one or more CICS regions and associated systems



Server: IYCWJFWM

- DUMMY907 (0/0)
- JOEPLEX (0/0)
- L830PLEX (0/0)
- L830PLX2 (0/0)
- L830PLX3 (0/0)
- L830PLX4 (0/0)
- L830PLX5 (0/0)
- L918PLX1 (0/0)
- L923CPLX (0/0)
- L938PLX (0/0)
- SAMEDAY (33/115)
  - ACPYME2C
  - ACPYME2D
  - ADCCICS1
  - CAWA
  - CICSF1
  - CICSF2
  - CICSRF1
  - CIP2CA01
  - CIP2CT01
  - CIP2CT02
  - CN6MAS01
  - DEVPLEXH
  - DUFF
  - IYCIJFC1 (IYCIJFC1)
  - IYCIJFD1 (IYCIJFD1)
  - IYCIJFE1 (IYCIJFE1)
  - IYCIJFF1 (IYCIJFF1)
  - IYCIJFF2 (IYCIJFF2)
  - IYCIJFWM (IYCIJFWM)
  - IYCWJCR1
  - IYCWJDM2
  - IYCWJDR1
  - IYCWJER1
  - IYCWJEA0

CNX0211I Scope: SAMEDAY. Resource: GATEWAYS. 3 records collected at 20-Oct-2009 17:59:14

Name	Status	Hostname	Version	GatewayID	Netname	UpTime	TCP	Clients	MaxPipes	InUsePip
CTGPW3	RUNNING	WINMVS2C	7.1.0.0	GBIBMIYA.CTGPW3	CTGPW	01:09:58	4008	0	100	0
CTGRED1	RUNNING	winmvs24.h...	7.2.0.0	MV24.CTGRED1X	RED1EXCI	1 day 03:49:49	4148	0	250	0
CTGPW4	RUNNING	WINMVS2C	7.2.0.0	GBIBMIYA.CTGPW4	CTGPW	01:10:56	5008	0	100	1

CNX0211I Scope: SAMEDAY. Resource: CONNECTIONS. 12 records collected at 20-Oct-2009 17:59:43

Name	Server	Prot...	Target	Requests	Response	Sent	Received	ActiveReq	InUseSess	MaxSess	
CTGPW3	MIKE	-	-	-	-	-	-	-	-	-	-
CTGPW3	IPICZ32A	-	-	-	-	-	-	-	-	-	-
CTGPW3	IPICZ32B	-	-	-	-	-	-	-	-	-	-
CTGRED1	IPCTG24U	IPIC	winmvs24.hursl...	25	17	14114	7055	0	0	100	0
CTGRED1	IPCTG24V	IPIC	winmvs2c.hursl...	0	0	0	0	0	0	0	0
CTGPW4	IYK2Z32B	EXCI	IYK2Z32B	0	0	0	0	0	0	-	7
CTGPW4	DAVEKEY	-	-	-	-	-	-	-	-	-	-
CTGPW4	IPICBAD	-	-	-	-	-	-	-	-	-	-
CTGPW4	IPICZ32A+	IPIC	localhost:6001	0	0	17686	24346	0	0	0	0
CTGPW4	IPICZ32B	IPIC	127.0.0.1:6002	0	0	478	658	0	0	0	0
CTGPW4	IPICZ32C	-	-	-	-	-	-	-	-	-	-
CTGPW4	IYK2Z32A	EXCI	IYK2Z32A	74	3	10360	10360	0	1	-	0

- **CICS TG Explorer Plugin V1**
  - Explorer, Gateway and CICS connections views
  - Test connection function
  - Supports CTG V7.1 or later on any platform
  - Requires CICS Explorer V1.0.0.6 or later
  - Licensed for use with CICS TS or TXSeries
  - Replaces SupportPac CS05

http://www.ibm.com/software/htp/cics/explorer

SHARE in Anaheim 2011

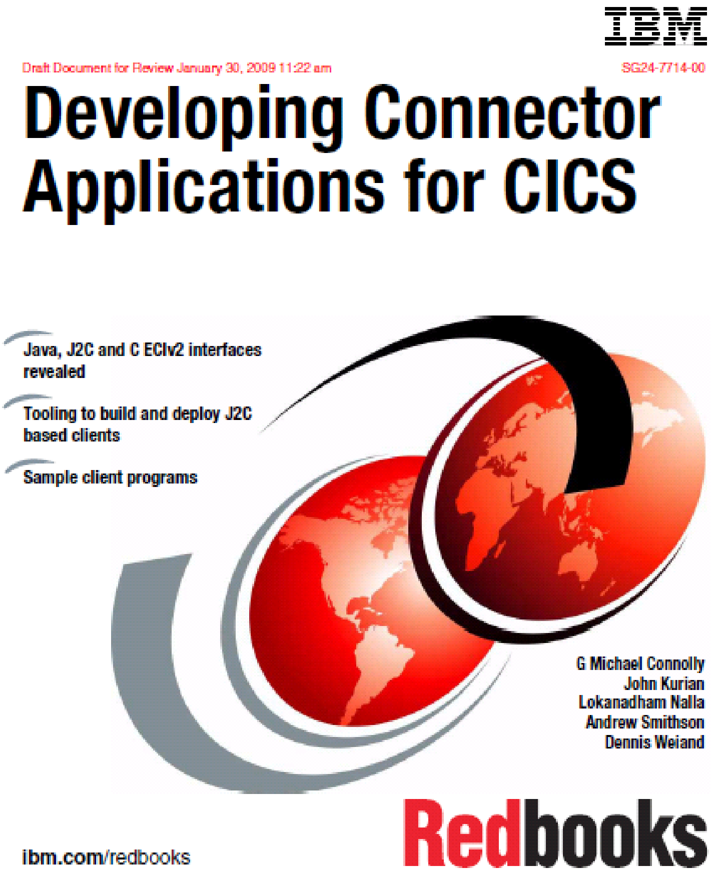
V8 - Other Enhancements

- Applets support for SSL
  - Java applets can now use an encrypted SSL or Transport Layer Security (TLS) connection to a remote Gateway daemon
  - In addition to the existing SSL support for remote Java clients
- IPIC enhancements:
  - CICS TG supports IPIC connections to CICS TS V4.1 that use multiple sockets per connection.
  - Alleviates lock contention under load
  - Reduce CPU usage when compared with the previous version of the product
- Other capabilities
  - A new set of sample configuration files is provided for the CICS TG for z/OS v8.0
  - Replaces usage of Configuration tool

Summary

- Gateway deployment topologies
- Application programming interfaces
  - ECI, EPI, and ESI
  - ECIv2
  - Programming with .NET framework
  - Programming in Java
- CICS Transaction Gateway v8.0
  - New and improved capabilities
  - High availability configurations

Developing Connector Applications for CICS SG24-7714



IBM

Draft Document for Review January 30, 2009 11:22 am

SG24-7714-00

Developing Connector Applications for CICS

Java, J2C and C ECIv2 interfaces revealed

Tooling to build and deploy J2C based clients

Sample client programs

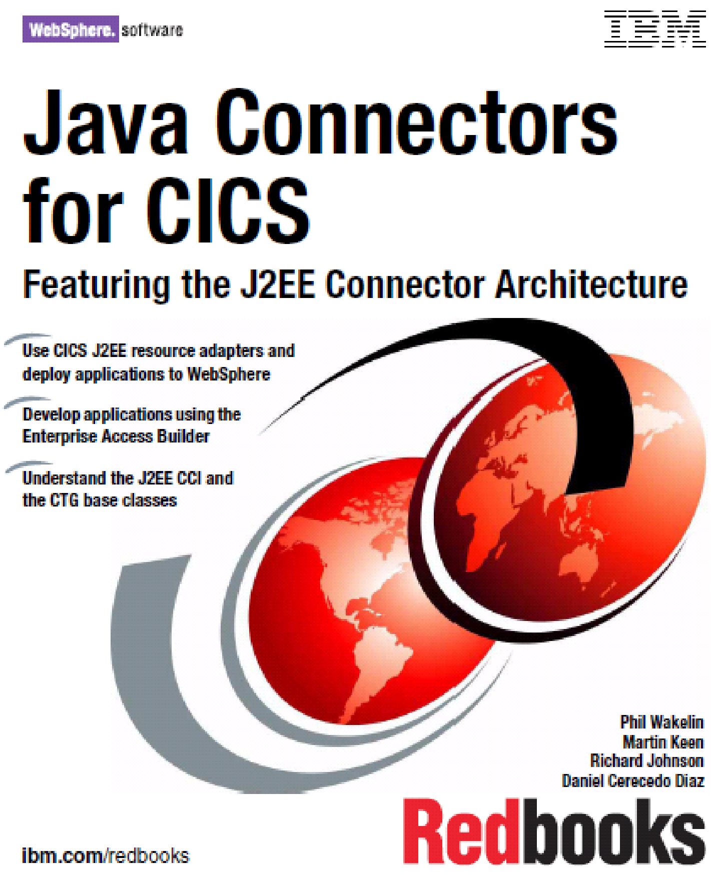
G Michael Connolly
John Kurian
Lokanadham Nalla
Andrew Smithson
Dennis Weiand

ibm.com/redbooks

Redbooks

- Topics:
  - Developing applications using Java, JCA, and ECIv2 interfaces
  - Using IDE tooling to develop JCA applications
  - Programming with channels and containers
  - Samples for Java and C/ECIv2 applications

Java Connectors for CICS SG24-6401



WebSphere. software

IBM

Java Connectors for CICS

Featuring the J2EE Connector Architecture

Use CICS J2EE resource adapters and deploy applications to WebSphere

Develop applications using the Enterprise Access Builder

Understand the J2EE CCI and the CTG base classes

Phil Wakelin
Martin Keen
Richard Johnson
Daniel Cerecedo Diaz

Redbooks

ibm.com/redbooks

- Topics
  - Deploying CICS resource adapters in J2EE servers
  - Developing Java applications
    - CICS TG base classes
    - JCA CCI classes
    - ECI and EPI