Integrating IBM WebSphere into CICS with CICS Transaction Gateway on IBM AIX

A case study

Vivek Arora

October 26, 2012

The integration of IBM® WebSphere® with IBM CICS® Transaction Gateway (CICS TG) enables client J2EE applications to access existing programs on the CICS back-end system. This method describes integration in a particular way.

Introduction

This method covers the integration of IBM WebSphere Application Server to CICS using CICS Transaction Gateway on IBM AIX® deployed in the remote mode that communicates with CICS Transaction Server on IBM z/OS® using external call interface (ECI) over Transmission Control Protocol/Internet Protocol (TCP/IP). It enables client J2EE applications designed for heuristic conditions to access existing programs in CICS.

Environment setup

- The CICS Transaction Gateway (CTG) provides the ECI CICS resource adapter that would be installed in the WebSphere Application Server.
- J2EE Connector Architecture (JCA) compliant Java™ applications would be deployed in a WebSphere clustered environment.
- The WebSphere managed environment would provide sharable connections that would not be released back to the pool upon a close call.
- The CTG and WebSphere would run on AIX logical partition (LPAR) nodes.
- The CTG would be deployed in the remote mode of operation with the CTG Gateway daemon providing connection managers and worker threads.
- The CICS Client would connect to CICS Transaction Server using ECI over TCP/IP.
- The CTG, deployed on an AIX LPAR, would be hardwired to a CICS region (z/OS).

Software used

- WebSphere Application Server Network Deployment v7.0.0.13 on AIX 6.1
• CICS Transaction Gateway version 7.2.0.2 for AIX6.1
• CICS Client for AIX version 7.2.0.2
• CICS Transaction Server version 3.2 for z/OS
• IBM DB2® 9.1 on z/OS
• z/OS 1.11
• Resource adapter ECI 7.2.0.2 (that is, the cicseci.rar file)

The software version of CTG can be ascertained with the `ctgstats` administrative command and the `cicscli -v` command would provide the version of the CICS client. It is important that these software versions should be at the same level (for example, V7.2.0.2).

**CICS Transaction Gateway**

The CICS Transaction Gateway on AIX platform will provide the programming and deployment facilities to WebSphere to connect to CICS Server on z/OS, as illustrated in Figure 1.

**Figure 1. WebSphere and CTG using ECI connectivity to CICS**

WebSphere and JCA Resource Adapter

The ECI resource adapter provided by the CICS Transaction Gateway would support the J2EE Common Client Interface (CCI), as defined by JCA, and is illustrated in Figure 2.

**CICS resource adapter**

The ECI resource adapter provided by the CICS Transaction Gateway would support the J2EE Common Client Interface (CCI), as defined by JCA, and is illustrated in Figure 2.
The JCA-compliant application components would interact with the JCA-implemented ECI resource adapter (system-level software driver) in the same way as they would with any other standard resource factory, for example JDBC resource factory (javax.sql.DataSource). The only difference with a JCA resource adapter is that the client can use resource adapter (RA) specific API and CCI API (supported by a resource adapter).

The CICS ECI resource adapter (cicseci.rar) supports one-phase transactional protocol. It avoids any overhead of unnecessary XA flow between WebSphere and CTG, and enables the Java application commit multiple JCA requests to the same CICS region as one unit of work.

- **Remote mode of operation**

The usage of CICS Transaction Gateway components is based on their deployed mode of operation. (a) The Gateway daemon is used in the `remote` mode of operation, as it listens for the incoming work and manages the threads and connections. (b) The Client daemon can be used in the `remote` as well as the `local` mode of operation for communication with CICS.

The Gateway daemon is not required in the `local` mode because the CTG code runs directly in the application server address space. In the `remote` mode, the CTG runs the Gateway daemon in stand-alone mode (as shown in Figure 3).
Figure 3. CTG in remote mode

In the remote mode, the J2EE application uses the J2C connection from WebSphere to communicate (over TCP) with the CTG Gateway daemon's connection manager, that in turn allocates requests to the worker thread. The CTG Gateway daemon's connections is configured in the GATEWAY section of the initialization (CTG.INI) file.

Listing 1. CTG.INI - GATEWAY section

```ini
SECTION GATEWAY
initconnect=1
initworker=1
maxconnect=100
maxworker=100
ENDSECTION
```

In the above configuration, the maximum number of connection managers (that is, `maxconnect`) and worker threads (that is, `maxworker`) have been configured to 100 (default). This configuration should be optimized based on the load pattern or anticipated workloads. The initial number of connection manager threads (that is, `initconnect`) and worker threads (that is, `initworker`) have been configured to 1.

The CTG Client daemon is configured in the CTG.INI file in the CLIENT section.
Listing 2. CTG.INI - CLIENT section

SECTION CLIENT = *
  CPIPADDRESMASK=00000000
  LOGFILE=CICSLCLI.LOG
  MAXBUFFERSIZE=32
  MAXREQUESTS=256
  MAXSERVERS=10
  MAXWRAPSIZE=0
  REMOTENOIDEINACTIVITYPOLLINTERVAL=60
  SRVRETRYINTERVAL=60
  TERMINALEXIT=EXIT
  TRACEFILE=CICSLCLI.BIN
ENDSECTION

Communication protocol

The CICS client (part of CTG) accesses the CICS region using ECI over the TCP/IP communication protocol, as illustrated in Figure 4.

Figure 4. CTG - CICS communication protocol

CICS Transaction Gateway Communication Protocol

The TCP/IP protocol is configured in the CTG.INI file in the GATEWAY, SERVER, and DRIVER sections.

Listing 3. CTG.INI – TCP/IP configuration

SECTION GATEWAY
protocol@tcp.handler=com.ibm.ctg.server.TCPHandler
protocol@tcp.parameters=connecttimeout=2000;dropworking;idletimeout=600000;
pingfrequency=60000;port=xxxx;solinger=0;sotimeout=1000;
log@info.dest=console
log@error.dest=console
ENDSECTION

SECTION SERVER = xxxxx
UPPERCASESECURITY=N
PROTOCOL=TCPIP
NETNAME=xxxxxxx
PORT=xxxx
CONNECTTIMEOUT=0
TCPKEEPALIVE=N
ENDSECTION

SECTION DRIVER = TCPIP
  DRIVERNAME=CCLIBMIP
ENDSECTION
An alternative communication protocol, Internet Protocol interconnectivity (IPIC) over TCP/IP), suitable for Java clients (WebSphere) using the ECI resource adapter can be used for communication between CTG V7.2 on AIX 6.1 and CICS Transaction Server V3.2 on z/OS.

**WebSphere-managed connection environment**

In the WebSphere-managed connection environment, the JCA compliant J2EE application uses the quality of service provided by WebSphere Application Server. The connections are defined in the WebSphere (resource adapter) J2C connection factory definitions, and the connections are sharable (by default), that is, the connections would be held and reused by WebSphere. In the managed mode, the Java application code would not be expected to open and close the connections with the CTG.

**Transaction management and heuristic conditions**

CICS and DB2 on z/OS supports full two-phase commit protocol for transactions. The CTG, however, supports one-phase commit with last participant support and it mimics being a two-phase capable resource manager. The last participant support could have a heuristic outcome to the transaction in which the transaction manager (WebSphere) might not be sure whether all resources were completed in the same direction (either committed or rolled back). With the acceptance of heuristic hazard in last participant support, lies the increased risk of a heuristic outcome. A heuristic outcome might occur if the transaction service (JTS) receives no response from the commit one-phase flow on CTG. In this situation, the transaction service cannot determine whether the changes for the CTG were committed or rolled back, so cannot reliably drive the correct outcome of the global transaction on the remaining two-phase commit resources (CICS/DB2).

The **Heuristic retry limit** value can be configured in WebSphere (as shown in figure 5) which can be set depending on the application design, and implications of a missed update compared to a repeated update.
The following two transactional scenarios can be considered in deciding the heuristic retry limit.

- Scenario 1 - The transaction fails on the CICS system due to a network or similar condition and so the data is not updated in CICS.
- Scenario 2 - The transaction works on the CICS system and the data is updated, but the response back to WebSphere is interrupted so that WebSphere thinks it failed.

Retrying would be appropriate for scenario 1, but not for scenario 2. If the scenario is unknown, the **Heuristic retry limit** value could be set to 0 (as shown in Figure 5).

**CTG topology**

On each AIX LPAR node, there is one CTG and two WebSphere instances (part of a cluster). The CTG is linked to a specific CICS region on z/OS to service the client requests along a transaction path.

There are a total of eight nodes/CTGs and 16 WebSphere application servers in the topology (as shown in Figure 6). The WebSphere J2EE application invokes the CICS programs through CTG and accesses DB2 using JDBC.
The two WebSphere Application Servers on each node are configured with maximum 50 web container threads each (to service client requests over the HTTP protocol) in the thread pool, with the combined total maximum of 100 connections on the node.

Each CICS resource adapter is configured with maximum 50 J2C connections in the connection factory definition, and as such, there is a total combined maximum of 100 J2C connections to one CTG on the node. The CTG Gateway daemon is configured with 100 connection managers (maxconnect) and 100 worker threads (maxworker).

The data source connections configured in each WebSphere Application Server is set to a maximum of 50 in the JDBC connection pool so that a combined total of 100 connections (with the two application servers on the node) can be requested at a time to the DB2 database.

**Limitations of CTG deployment**

- The ECI communication protocol lacks the capability of providing the conversational global unit of work ID for transactions, which makes it virtually impossible to link failed transactions from CTG through to CICS.
- The CTG's resource adapter used with the one-phase commit transaction protocol and last participant support requires the acceptance of heuristic hazard. This introduces increased risk of heuristic outcome, whereby the transaction service might fail to determine if changes for the one-phase commit resource (that is, CTG) were committed or rolled back, and hence cannot drive reliably the correct outcome of the global transaction on other participating two-phase commit resources (that is, CICS and DB2).
- The CTG lacks the capability of identifying orphaned threads or threads in the limbo state, and as such those connection managers that are no longer connected to a client, cannot be
returned to the free pool because there is still an outstanding worker thread, waiting for work to complete.

Problem management

Problematic behavior, if any, in the integrated environment with Websphere and CICS Transaction Gateway might require appropriate diagnostic information to be gathered and analysed in conjunction with IBM Technical Support. The WebSphere accelerated value program (as offered to premium support customers) facilitates in problem management of such situations.

For troubleshooting, some of the following logs and trace data might be needed.

(a) WebSphere logs

The WebSphere system output logs (such as SystemErr and SystemOut) provide details on error messages when they occur, for example, insufficient ConnectionManagers (CTG6562E) as given in Listing 4.

Listing 4. Insufficient ConnectionManagers - CTG6562E

```
09:19:38:054 tcp: .ThreadManager: + Trying to allocate object again
09:19:38:054 tcp: .ThreadManager: + !! Reached maximum number of objects !!
09:19:38:054 tcp: .ThreadManager: + Timeout period is over
09:19:38:054 tcp: S-C: CTG6562E Connection to tcp:@Socket[addr=127.0.0.1, port=xxxx, localport=xxxx] rejected due to insufficient ConnectionManagers.
```

In such circumstances, the configuration of connection managers (maxconnect), as defined in the CTG.INI file, can be increased. In this case study, it was increased from 100 to 150. However, it only helped in delaying this scenario (of insufficient connections) from occurring.

(b) CTG tracing data

For troubleshooting CTG problems, tracing data on CTG Gateway would be required. In some scenarios, Client daemon trace (that is, cicscli.log) might be required.

The `ctgstart -x` command starts CTG and `trace` is enabled, whereas the `ctgstart` command without the `-x` parameter starts CTG but the `trace` would be disabled.

Alternatively, the Gateway daemon trace on CTG 7.2 for AIX can be enabled and disabled dynamically from the command prompt. For example:

- To dynamically enable full debug trace, use: `ctgadmin -a trace -tlevel=4`
- To dynamically disable trace, use: `ctgadmin -a trace -tlevel=0`

The `tlevel` options are as follows:

- 0 - No trace information is output.
- 1 - Exception tracing. Only exceptions are traced.
- 2 - Trace exceptions, and entry and exit of methods.
• 3 - Trace exceptions, some internals, and entry and exit of methods.
• 4 - Full debug tracing (all trace points, equivalent to `ctgstart -x`).

(c) WebSphere connector trace

In relation to investigating any problem on the connection from WebSphere to CICS TG, a connector trace would be required.

The connector trace string to be used for connecting to the enterprise information system (CICS) is given in Listing 5.

**Listing 5. Connector trace string**

```
*=info:
WAS.j2c=all:
com.ibm.connector2.*=all:
Transaction=all
```

This connector trace string can be configured in WebSphere by clicking **Troubleshooting > Logs and Trace > Change Log Detail Levels**.

The connector trace should be started before the problem manifests itself. Sometimes, it might be difficult to predict the manifestation of the problem. At other times, when the tracing has been configured and turned **ON**, the server might appear to be busy in doing the tracing, and the load balancer (in the topology) could farm out the requests to other servers in the cluster, with the result the problem might not reoccur on the application server where the connector trace was being obtained.

(d) Java system jumps and jextract data

Java system dumps may be required (by IBM Technical Support) for problem determination in relation to the status of CTG threads, which can be collected using the following command:

**Listing 6. System dump**

```
ctgamin -a dump -system
```

If the command specified in Listing 6 does not work, then it may be necessary to configure dump agents with -Xdump on the CTG. The system dump (as obtained in Listing 6) should be processed using jextract (from Java’s bin directory) which produces a zip file. The syntax for the jextract command is shown in Listing 7.

**Listing 7. jextract command**

```
jextract [system_dump_name_as_produced_above] (without the [ ] chars)
```

The level of jextract run must be from the same level of Java that created the system dump.

(e) Statistics information
The statistical information about a running CTG in relation to the usage of connections and workers can be obtained with the statistics API handler of the CTG. The stats output can be gathered using command shown in Listing 8.

**Listing 8. Statistical data**

```bash
ctgadmin -a stats -gs > statscollected.out
```

Connection creeps on the CTG (for example, from transaction abends) that might cause connection time-outs, can be demonstrated with the collected statistical data on the current and allocated connection manager (CM_CURRE/CM_CALLOC) and worker threads (WT_CCURR/WT_CALLOC). The bar chart graph, as in Figure 7, illustrates the collected statistical data on the number of allocated connection managers and worker threads over a period of time.

**Figure 7. CTG statistics on allocated connection managers and worker threads**

![Bar chart showing CTG statistics](image)

In the statistic data, as shown in Listing 9, there is one connection time out (CM_LTIMEOUTS=1) when the allocated (CM_CALLOC=150) and currently used (CM_CCURR) connection managers have reached the limit of maximum number of connection managers (CM_SMAX=150).

**Listing 9. Statistical data**

```
CTG8218I The command completed successfully
===================================
CTGAdmin - CICS Transaction Gateway Administration Program, version 7.2.0.2
(C) Copyright IBM Corporation 2002, 2010. All rights reserved.
CM - Connection manager
  CM_SMAX=150 (Maximum number of connection managers)
  CM_SINIT=1 (Initial number of connection managers)
  CM_CCURR=150 (Current number of connection managers)
  CM_CALLOC=150 (Currently allocated connection managers)
  CM_LTIMEOUTS=1 (Number of times connect time out reached)
WT - Worker thread
  WT_SMAX=150 (Maximum number of worker threads)
  WT_SINIT=1 (Initial number of worker threads)
  WT_CCURR=145 (Current number of worker threads)
  WT_CALLOC=136 (Currently allocated worker threads)
```
The statistical information, therefore, helps in the examination of how requests from a user application progressed through the transaction gateway.

**Conclusion**

The integration of WebSphere with CICS Transaction Gateway to access CICS has been demonstrated with a peculiar topology, with a focus on diagnosing and managing problematic scenarios to ensure the availability of the integrated environment. This would be of help to those deciding on the topology or maintaining operational systems.
Related topics

- Bottleneck analysis of CICS TG using OMEGAMON XE to improve CICS performance
- Troubleshooting connection pooling (J2C) problems in WebSphere Application Server
- CTG statistical API
- When the CICS TG V7.0 process is locked on AIX platform – Problem Determination.
- Tuning WebSphere Application Server V7 and V8 for performance

© Copyright IBM Corporation 2012
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)