CICS Transaction Gateway – High Availabilty Exit samples

Current Version: v1.1, May 2010
Authors: Phil Wakelin, Andrew Smithson, IBM UK, Hursley

Description

This SupportPac provides sample CICS request exits to be used with the CICS Transaction Gateway for z/OS. These exits provide the ability to quickly create high availability and request validation rules for use with the CICS Transaction Gateway.

Version 1 of the SupportPac provides a set of sample CICS request exits to be used with the CICS Transaction Gateway for z/OS V7.2. Three different CICS request exit implementations are provided to either perform round robin or failover based workload balancing or alternatively request validation. Only one CICS request exit can be deployed at any one time, and each exit must be configured using a HFS configuration file as described in the documentation.

Three different CICS request exit implementations are provided as follows:

1. FailOver – This exit uses a list of CICS servers to define how server names are remapped. Requests will be retried using a static list of servers, each server being used in turn from the list until the list is exhausted or the request succeeds.
2. RoundRobin – This exit distributes ECI requests around a list of CICS servers in round-robin fashion. Requests will be distributed evenly across the servers, and retried until the request succeeds or all serves have been used.
3. RequestValidation – This exit validates the transaction ID, program and user ID parameters in the ECI request against a defined list. No server name remapping is performed.

System Requirements

CICS Transaction Gateway for z/OS – V7.2
Supported configuration – Any ECI request using the Gateway daemon on z/OS in remote mode

Skill level required

System administrators with an understanding of the configuration of the CICS TG

Changes in v1.1

- Comment lines in configuration file
- Version numbers at startup

Supplied files

- ca1t.pdf - Documentation
- ctgha.jar - JAR file containing Java source and compiled byte code
- ctgha.ini - Sample configuration file for RoundRobin or FailOver exits
- ctgrv.ini - Sample configuration file for RequestValidation exit
- license - Directory for translated license files

Installation – Gateway daemon:

To install the SupportPac perform the following steps:

1. Copy the file ca1t.zip to a temporary directory on your chosen system and uncompress using unzip.

2. Transfer the ctgha.jar to a directory on your z/OS system, for example you could use the HFS /u/cicstg/tmp directory. If using ftp ensure the file is transferred in *binary* mode to ensure correct data conversion.

3. Add the location of the ctgha.jar to the CLASSPATH used to start your Gateway.
   For example update your CLASSPATH statement in your STDENV as follows:
   ```
   CLASSPATH=/u/cicstg/tmp/ctgha.jar
   ```

4. Create a configuration file in your HFS and define the location to your Gateway using the variable CTG_HACONFIG. For example to use the supplied sample file ctgha.ini, copy this to the HFS directory /u/cicstg/tmp and set the following variable in your Gateway STDENV definition.

   ```
   CTG_HACONFIG=/u/cicstg/tmp/ctgha.ini
   ```

For more details on how to create configuration files refer to the section Configuration – High availability or Configuration – Request Validation

5. Ensure that your Gateway has read access to the configuration file.

   For example to give the file owner rw access and group r access issue the following USS command:
   > chmod 640 /u/cicstg/tmp/ctgha.ini

6. Update your Gateway daemon configuration file (ctg.ini) to name the fully-qualified package name of the cicsrequestexit in the SECTION GATEWAY, for example:

   To enable use of the round robin exit specify:

```
cicsrequestexit=com.ibm.ctg.samples.ha.RoundRobinExit
```

To enable use of the fail over exit specify:
```
cicsrequestexit=com.ibm.ctg.samples.ha.FailOverExit
```

To enable use of the request validation exit specify:
```
cicsrequestexit=com.ibm.ctg.samples.ha.RequestValidationExit
```

7. Start the Gateway daemon using your normal operating procedures

Configuration – High availability exits

1. Create a configuration file in your HFS filing system using the following format:

JOBNAME=SERVER1,SERVER2,SERVERn

2. The rules for the configuration file are as follows:

    a. Each line must start with a key which will be used to match the jobname of the Gateway daemon
    b. There should be a single key in each line delimited by the equals character (‘=’), and one or more values, delimited by the comma character (‘,’)
    c. Each value should be a CICS server name to be used for server name remapping. Server names must either be a CICS APPLID (for EXCI requests) or an IPICSERVER name defined in the ctg.ini file.
    d. Multiple lines can be specified, this allows a single configuration file to be shared across a Gateway group.

Two different exits are provided to help in building high availability configurations with the CICS Transaction Gateway. The RoundRobinExit remaps server names and retries ECI requests in a round-robin fashion. The FailOverExit remaps server names and retries requests in a set order according to the SERVER values defined in the configuration file.

Note: Only synconreturn and extended LUW requests will be remapped by the CICS request exit infrastructure, and XA requests will not drive the exit. In addition the RoundRobin and FailOver exits ignore any server name that is passed in on an ECI request and will remap all ECI requests as specified in the configuration file.

FailOver exit:
When using the FailOver exit the following configuration sample file could be used:

CTGJOBA=SERVERA1,SERVERA2,SERVERB1
CTGJOBB=SERVERB1,SERVERB2,SERVERC1

CTGJOBC=SERVERC1,SERVERC2,SERVERA1

The configuration is to be used by 3 different Gateway jobs, (CTGJOBA, CTGJOBB and CTGJOBC). CTGJOBA, will remap all ECI requests it receives to the CICS server called SERVERA1, if this fails, then the request would first be retried to SERVERA2 and if this fails to SERVERB1. Different rules will apply to the Gateway jobs CTGJOBB and CTGJOBC as defined in the subsequent lines.

RoundRobinExit

When using the RoundRobin exit each server is used in a round-robin fashion from the list so that requests are evenly distributed across the defined servers. For example the following configuration could be used to define a Round Robin policy to distribute requests across three servers:

CTGJOBA=SERVERA1,SERVERA2,SERVERA3
CTGJOBB=SERVERB1,SERVERB2,SERVERB3
CTGJOBB=SERVERC1,SERVERC2,SERVERC3

The configuration would again be used for 3 different Gateway jobs, (CTGJOBA, CTGJOBB and CTGJOBC). CTGJOBA, would redistribute requests in across the three defined CICS servers (SERVERA1, SERVERA2 and SERVERA3) such that one third of requests went to each server. If a request failed it would be automatically retried to the next server, and if this failed to the remaining server.

Configuration – RequestValidation exit

1. Create a configuration file in your HFS filing system using the following format:

TRANSID=*TXN1,TXN2,TXNn*
PROGRAM=*PROGRAM1,PROGRAM2,PROGRAMn*
USERID=*USER1,USER2,USERn*

The rules for the configuration file are as follows:

1. Each line must start with the key USERID, PROGRAM or TRANSID
2. There should be a single key in each line delimited by the equals character (‘=’), and one or more values, delimited by the comma character (‘,’)
3. The value is either
    a. A CICS transaction ID.
    b. A CICS program name
    c. A RACF user ID
    d. The symbolic value “NULL”

When using the RequestValidation exit only the values defined in the TRANSID, PROGRAM and USERID lines will be used. Each ECI request will be validated against the listed values by the Gateway daemon, and if the values in the ECI request are not listed the request will be rejected with the response ECI_ERR_INVALID_REQUEST (-14). The request will not be retried. The value NULL is a special value that allows an unset (null) value to be validated as a user ID or transaction ID. If a KEY (such as PROGRAM) is not defined, then validation of that ECI parameter will not be performed. Only synconreturn and extended LUW requests will be validated by the CICS request exit infrastructure, and XA requests will not drive the exit.

For example the following line would cause the Gateway to reject any requests that do not specify the mirror transaction CSMI

```
TRANSID=CSMI
```

The following lines would reject any request that specified a userid, or that did not specify PROG1 or PROG2 as the program.

```
USERID=NULL
PROGRAM=PROG1,PROG2
```

Advanced usage

1. To enable the use of a log file rather than the default stdout destination for log messages set the CTGSTART_OPTS variable in STDENV as follows:

```
 CTGSTART_OPTS=
-j-Dcom.ibm.ctg.samples.cicsrequestexit.out=/myhfs/ctg.log
```

Alternatively to log to JES add the following DD statement to the JCL used to start the Gateway job:

```
 //EXITLOG   DD   SYSOUT=*
```

Then set the `CTGSTART_OPTS` variable in your STDENV as follows:

```
  CTGSTART_OPTS=
-j-Dcom.ibm.ctg.samples. cicsrequestexit.out=//DD:EXITLOG
```

To log to an MVS dataset, set the `CTGSTART_OPTS` variable in your STDENV as follows

```
  CTGSTART_OPTS=
-j-Dcom.ibm.ctg.samples. cicsrequestexit.out=//'XX.YY.ZZ'
```

2. The logging level from the exits can be dynamically controlled whilst the Gateway daemon is running using the Gateway trace level. This enables problem with the configuration files or server name remapping function to be isolated. To dynamically enable tracing use the tlevel option, where tlevel=2 enables product tracing and tlevel=0

disables. For example the following MVS command will enable exit tracing for problem determination purposes:

```
/F <jobname>,APPL=TRACE,TLEVEL=2
```

To start the Gateway with tracing enabled specify the '-trace' flag for ctgstart, for example when using CTGBATCH add the following to the PARM statement

```
// PARM='//u/ctg720/bin/ctgstart -noinput -trace'
```

Application Development

The supplied ctghav.jar contains both the compiled byte code and the Java source enabling further customisation if required. To modify and compile the code you will need the following:

- IBM SDK V5 (on any supported platform)
- ibmjzos.jar - Available in the Java 5 IBM SDK for z/OS in the lib/ext directory.
- ctgclient.jar – Available with the CICS TG V7.2 in the classes directory

Example output:

If the exit is correctly configured then at start-up you will see the following messages at Gateway start-up:

```
11/26/08 13:39:47:996 CA1T Starting CICS request exit FailOverExit V1.1
in Gateway CTGJOBC
11/26/08 13:39:47:999 CA1T Reading file /u/ctg/tmp/ctgha.ini
11/26/08 13:39:48:055 CA1T Server selection for CTGJOBC built with
servers: SERVER1,SERVER2,SERVER3,SERVER4,
```

If logging is enabled the following additional messages will be output when the exit remaps a server name

```
11/26/08 14:39:47:996 CA1T  Retry:0 remapped server:CICS to:SERVER1
11/26/08 14:39:48:996 CA1T  Retry:1 remapped server:CICS to:SERVER2
```

Usage notes:

1. The following message indicates that the exit cannot be loaded. Check that you have correctly set your CLASSPATH variable to point to a valid jar file containing the sample exit, and that the cicsrequestexit parameter is correctly configured in your ctg.ini file.

```
11/04/08 16:21:22:604 [0] CTG8446E CICS Request Exit
com.ibm.ctg.samples.ha.RoundRobinExit failed to initialize with
exception java.lang.ClassNotFoundException: com.ibm.ctg.samples.
ha.RoundRobinExit
```

2. The following message indicates that a line in the configuration file did not contain a valid key value. Ensure that each key is followed by the equals character ('=')

```
11/04/08 16:16:14:951 CA1T Reading file <file>
11/04/08 16:16:14:954 CA1T Error reading file java.io.IOException: Key
parsing failed
11/04/08 16:16:14:956 CA1T Terminating with java.lang.Exception:
java.io.IOException: Key parsing failed
11/04/08 16:16:14:983 [0] CTG8446E CICS Request Exit com.ibm.ctg.
samples.ha.RoundRobinExit failed to initialize with exception
java.lang.Exception: java.lang.Exception: java.io.IOException: Key
parsing failed
```

3. The following message indicates that the variable CTG_HACONFIG is not set. Ensure that the variable is set in your STDENV configuration member

```
11/04/08 16:55:43:177 CA1T Error reading file java.io.IOException:
CTG_HACONFIG variable not set
11/04/08 16:55:43:180 CA1T Terminating with java.lang.Exception:
java.io.IOException: CTG_HACONFIG variable not set
```

4. The following message indicates that the Gateway does not have the correct permissions to access the HA configuration file. Ensure that the Gateway region user ID has read access to the configuration file.

```
11/04/08 16:58:30:993 CA1T Error file not found java.io.FileNot
FoundException: <file> ha (EDC5111I Permission denied.)
11/04/08 16:58:30:995 CA1T Terminating with java.lang.Exception:
java.io.FileNotFoundException: <file> (EDC5111I Permission denied.)
```

Further information

For further information on usage of the CICS request exit refer to the CICS TG V7.2 information center:
http://publib.boulder.ibm.com/infocenter/cicstgzo/v7r2/index.jsp?topic=/com.ibm.cics.tg.zos.doc/progde/cicsreqexit.html

For further information on developing a CICS request exit refer to the com.ibm.ctg.ha package in the CICS TG javadoc:
http://publib.boulder.ibm.com/infocenter/cicstgzo/v7r2/index.jsp?topic=/com.ibm.cics.tg.zos.doc/javadoc/index.html

For further information on the ibmjzos package used in this SupportPac refer to:
http://www.ibm.com/developerworks/java/zos/javadoc/jzos/overview-summary.html