

IBM – CICS Workshop



## **L01 – CICS – Resource Builder Lab**

*Lab Version V0.1*

---

***October, 2022***

Please send any comments on this lab exercise to:  
SangSoo HAN or Manjunath D  
[sshan@sg.ibm.com](mailto:sshan@sg.ibm.com)  
[manjud46@sg.ibm.com](mailto:manjud46@sg.ibm.com)

## **Overview**

- CICS® Transaction Server (TS) resource builder, a DevOps utility that provides a way to automate the creation of CICS resource definitions using a *configuration as code* approach.

CICS resource definitions are closely tied to the function of an application and frequently need tailoring for the application. But, at a system level, they have wide ranging impacts. They must be organized and controlled and the authority to change them is typically with a system programmer. CICS resource builder provides CICS application developers with a controlled way of creating and modifying CICS resource definitions. They can be confident that what they do meet application and system standards and is approved by CICS system programmers. CICS system programmers can be confident that application developers can only make changes to CICS resource definitions within a scope that they have set.

The configuration of application resources is defined in a simple YAML format. This can be stored directly in Source Control Management (SCM) along with other application code for version control and automation. CICS resource builder fits into a DevOps pipeline to allow you to easily orchestrate and provision your CICS applications.

CICS resource builder has a command-line interface that can be run from Linux®, Linux on Z, Mac, Windows, or z/OS® UNIX System Services. CICS resource builder works with the CSD update batch utility, DFHCSDUP, in CICS Transaction Server for z/OS to update the resource definitions that are used in the CICS TS region.

As a result, developers have a quicker turnaround time to getting resources defined in the CICS region. System programmers capture their best practices and standards in configuration to automate the process in a controlled way.

## **Lab Scenario**

1. Extract CSD definitions and import it to YAML file.
2. Generate the DFHCSD commands to re-define the resources into the CICS regions.

## **Lab Requirements**

Please note that there are often several ways to perform functions in and for CICS. This lab exercise will present one of the ways. If you are familiar with CICS, you will notice that some of the statements are general, and not necessarily true for every situation.

This lab uses the PCOMM and CICS Explorer. If you are not familiar with these, please contact one of the lab instructors for assistance.

The following are other assumptions made in this lab exercise.

- **CICS TS V6.1:** This lab exercise only works in CICS V6.1. You have your own z/OS image you can change all resources in four CICS regions.
- **Login:** A TSO userid is available with the appropriate password provided, and you will also use the same TSO userid with the z/OS Explorer.
- **The CICS Explorer:** In the lab environment we have installed the CICS Explorer to configure CICS resources and the security request recording function.

## **Lab Step Overview**

**Part 1: Try the CBSA (CICS Bank Sample Application )**

**Part 2: Extract CSD definitions by using the DFHCSUP**

**Part 3: Import the CSD definitions and create the YAML definitions**

**Part 4: Create the CSD definitions into the new group**

**Part 5: Run the batch job to define the new group into the CICS61F1 region**

**Part 6: Install the new group (BANKRB) by adding it to the list (61FLIST)**

**Part 7: Restart the CICS region (CICS61F1) and then, test the CBSA application in the CICS61F1 region.**

**Part 8: Summary**

## **Part 0: Check the CICS Explorer Connection to CICS**

In this part of the lab exercise you will configure the connection between the CICS Explorer running on your workstation to CICS running on z/OS.

### **Start the CICS Explorer**

- \_\_\_1. From the **desktop**, **double-click** the **CICS Explorer** icon to start the Explorer if it is not already running.



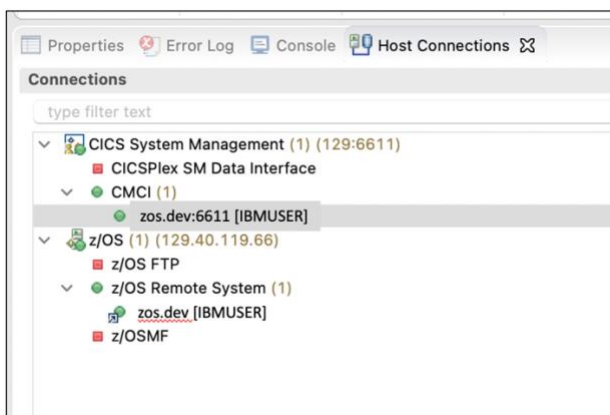
- \_\_\_2. When you start the Explorer, if you are prompted for a workspace, click the **OK** button to select the default.
- \_\_\_3. If the Explorer shows you a **Welcome page** click the **Workbench icon** in the upper-right corner to go to the workbench. Then **maximize** the Explorer window.



### **Verify that you have an FTP connection to z/OS in your CICS Explorer**

- \_\_\_4. If you have not already created connections to the z/OS host system, check the connection as in the screen shot. Both the **Remote System Explorer** and **CMCI** connections should be started and active.

**IP : zos.dev / CMCI port : 6611 / User : IBMUSER / Password : sys1**



## **Part 1: Try the CICS bank sample application (hereafter CBSA) in 4 CICS regions**

In this part of the lab exercise you will try the CICS bank sample application.

### **Try the CICS Bank Sample Application**

- \_\_\_1. Open a session in the PCOMM, and type “L CICS61T1” to logon.
- \_\_\_2. Clear the screen and type “OMEN” to start the main screen.

```
BNK1MA                      CICS Bank Sample Application - Main Menu

Select an option. Then press Enter.

Action . . . . . _ 1. Display/Delete/Update CUSTOMER information
                   2. Display/Delete ACCOUNT information
                   3. Create CUSTOMER
                   4. Create ACCOUNT
                   5. Update ACCOUNT
                   6. Credit/Debit funds to an ACCOUNT
                   7. Transfer funds

                   A. Look up Accounts with Customer Number

F3=Exit  F12=Cancel
```

- \_\_\_3. Try to display the customer number 1 and the account number 1. Also, you can create customer and account numbers as you want to test the CICS Bank Sample Application. ( CBSA )

## **Part 2: Extract CSD definitions by using the DFHCSUP**

1. Extract the resources from the **BANK** group by running the JCL, which copies the output to the CBDOUT dataset.  
CICS.CICS61.JCL(DFHCSD)
2. FTP the extracted output to a USS file using ASCII mode  
*ftp zos.dev*  
*> asc*  
*> get 'CICS.CICS61.JCL(CSDOUT)' c:/dfhrb.out*
3. Replace the GROUP 'BANK' to 'BANKRB' in the file "c:/dfhrb.out"

## **Part 3: Import the CSD definitions and create the YAML definitions**

### **a. Identify resources in CSD**

1. To identify the resources in the existing CSD, the existing CSD is processed by using a simple import file. The zrb import command builds the basic resource model, application, and resource definition files using the attributes within the cics-import.yaml.

```
resourceImport:
  destinations:
    application: cics-bankrb-application.yaml
    model: cics-bankrb-model.yaml
    resources: cics-bankrb-resources.yaml
  applicationInput:
    name: My BankRB Application
    description: Import BankRB application tutorial
  modelInput:
    target: cicsts-6.1.0
```

2. Run the zrb import command, inputting the import YAML file cics-import.yaml and DFHCSUP extract file definitions.csdup:  
Command:  
*zrb import --source c:/dfhrb.out --import c:/cics-import.yaml --output-dir c:/bundleOutput/*

Within the command above, several arguments are used:

- The --source argument is the path to the input file of resource definition commands, and the short name is -s.
- The --import argument is the path to the import YAML file that defines the resource imports, and the short name is -i.
- The --output-dir argument is the path to the output directory, and the short name is -o

The `zrb import` command builds empty application constraints, resource model, and resource definition YAML files.

```
Importing CICS Transaction Server for z/OS resources from dfhrb.out

Unselected resources:
  DB2CONN:          1
  DB2ENTRY:         1
  DB2TRAN:         11
  ENQMODEL:         1
  FILE:             2
  LIBRARY:          1
  PIPELINE:         1
  PROGRAM:          59
  TCPIPService:     2
  TRANSACTION:      18

Output directory files created:
  Resource model:      cics-bankrb-model.yaml
  Application constraints: cics-bankrb-application.yaml
  Resource definitions: cics-bankrb-resources.yaml
```

## ***b. Perform Selection Stage***

Choose which resource type of those shown in the **unselected resources** within the import report in the above section. Within the `cics-import.yaml` file add an `imports` yaml block. This should have the attribute `type`, with a value of resource type selected.



```

  resourceImport:
  destinations:
    application: cics-bankrb-application.yaml
    model: cics-bankrb-model.yaml
    resources: cics-bankrb-resources.yaml
  applicationInput:
    name: My BankRB Application
    description: Import BankRB application tutorial
  modelInput:
    target: cicsts-6.1.0
  imports:
    - type: file
    - type: program
    - type: transaction
    - type: enqmodel
    - type: library
    - type: db2entry
    - type: db2tran
    - type: tcpipservice
    - type: pipeline
    - type: db2conn

```

#### Totals:

##### Imported:

DB2CONN:	1
DB2ENTRY:	1
DB2TRAN:	11
ENQMODEL:	1
FILE:	2
LIBRARY:	1
PIPELINE:	1
PROGRAM:	59
TCPIPSERVICE:	2
TRANSACTION:	18

#### Unselected resources:

#### Output directory files created:

Resource model:	cics-bankrb-model.yaml
Application constraints:	cics-bankrb-application.yaml
Resource definitions:	cics-bankrb-resources.yaml

## **Part 4: Create the CSD definitions into the new group**

CICS® resource builder uses the YAML files to validate and build a DFHCSDUP commands file, using the `zrb build` command. The output of `zrb-build` command is used as input to the CSD update utility, DFHCSDUP.

The following command generates a DFHCSDUP commands file(BankrbYaml.out) from a resource model that has an application constraints file, by using the optional --application argument of the `zrb build` command.

```
zrb build --model c:/bundleOutput/cics-bankrb-model.yaml --application
c:/bundleOutput/cics-bankrb-application.yaml --resources c:/bundleOutput/cics-
bankrb-resources.yaml --output c:/bundleOutput/BankrbYaml.out
```

## **Part 5: Run the batch job to define the new group into the CICS61F1 region**

Ftp the *BankrbYaml.out* to z/OS dataset

```
C:\Users\Administrator>ftp zos.dev
Connected to zos.dev.
220-FTPD1 IBM FTP CS V2R5 at S0W1, 12:24:01 on 2022-10-10.
220 Connection will close if idle for more than 5 minutes.
501 command OPTS aborted -- no options supported for UTF8
User (zos.dev:(none)): IBMUSER
331 Send password please.
Password:
230 IBMUSER is logged on. Working directory is "IBMUSER.".
ftp>
ftp> asc
200 Representation type is Ascii NonPrint
ftp> put C:\bundleOutput\BankrbYaml.out 'CICS.CICS61.JCL(BANKRB)'
200 Port request OK.
125 Storing data set CICS.CICS61.JCL(BANKRB)
250 Transfer completed successfully.
ftp: 41738 bytes sent in 0.73Seconds 57.25Kbytes/sec.
ftp>
```

The commands file (BankrbYaml.out) that is built by the `zrb build` command is used as input to the CSD update batch utility program, DFHCSDUP, which updates the CSD data set with the new resource definitions. The updated CSD resource definitions are added to the CICS region. Use the following JCL to update the resource definitions to CICS regions:  
*CICS.CICS61.JCL(DFHCSDIN)*

## **Part 6: Install the new group (BANKRB) by adding it to the list (61FLIST)**

Install the NEW group (BANKRB) by adding it to the GRPLIST (61FLIST) of the CICS region by using CEDA command.

```
ADD
OVERTYPE TO MODIFY
CEDA  ADd
  Group      ==> BANKRB
  List       ==> 61FLIST
```

### **Part 7: Restart the CICS region (CICS61F1) and then, test the CBSA application in the CICS61F1 region.**

Restart the CICS region **CICS61F1** and see the resources installed from the new group (**BANKRB**). Issue “OMEN” transaction to test the CBSA application in CICS61F1 region.

### **Part 8: Summary**

**Congratulations**, you have installed NEW CICS TS resource using YAML files

In this lab you performed the following steps:

- Extract CSD definitions from CICS region DFHCSD file.
- Import CSD definitions into a YAML file
- Generate the DFHCSD commands to define the resources back into CICS region.
- Run the batch job to re-define the CICS resources with a different group name (BANKRB).