

Java IC Installation
Oracle FLEXCUBE Universal Banking
Release 14.6.2.0.0
Part No. F72916-01
[November] [2022]



Table of Contents

1. JAVA IC INSTALLATION	1-1
1.1 INTRODUCTION.....	1-1
1.2 PREREQUISITES	1-1
1.3 SERVER SETUP.....	1-1
1.4 WAR DEPLOYMENT AFTER BUILD	1-3
1.5 ORDER OF SERVER START	1-4
2. JAVA IC MAINTENANCE IN FCUBS	2-5
2.1 REQUIRED MAINTENANCE FOR JAVA IC.....	2-5
2.2 SCHEDULER JOB FOR TRIGGERING IC EOD IN FCUBS	2-9
3. SSL SETUP WITH SELF SIGNED CERTIFICATE.....	3-1
4. IC END OF DAY BATCHES.....	4-1
5. IC ADDITIONAL SETUP DETAILS.....	5-1
5.1 IC CONFIGURATIONS.....	5-1
5.1.1 <i>IC Services: Number of Processes, Number of Deployments for a service, Commit Frequency and Fetch Count</i>	<i>5-1</i>
5.1.2 <i>Server-Port settings in case of multiple deployments for a service</i>	<i>5-1</i>
5.1.3 <i>Plato Batch Logging & Persistence Logging</i>	<i>5-2</i>
5.2 WEBLOGIC APPLICATION SERVER CONFIGURATIONS	5-3
5.2.1 <i>Data Source Configuration</i>	<i>5-3</i>
5.2.2 <i>Number of deployments for IC service.....</i>	<i>5-3</i>

1. Java IC Installation

1.1 Introduction

This document lists steps to configure Application Server for JAVA IC Integration with FCUBS.

1.2 Prerequisites

Java IC installation requires a Weblogic domain.

Note:

- a. In the following sections, 10.10.10.10 IP address and 1010 port are used as an example. Please use valid IP and Port of corresponding server.
- b. In case of upgrade from previous release, upgrade scripts if present under UPGRADE folder has to be applied.

1.3 Server Setup

Java IC Setup includes two sets of services:

1. **INFRA Services:** There are two services under this category.
 - a. **Discovery Service:** This service is required for Java IC Services Registration. On start-up all Java IC services will be registered with Discovery Service. The registered services can make inter service calls by making use of Discovery Service.
Service Name: plato-discovery-services.war
 - b. **Config Service:** All the configuration related details will be stored in a database table (table name: PROPERTIES). Config service provides the required configuration details for the corresponding Java IC Services during service start up.
Service Name: plato-config-services.war
2. **Java IC Services:** These Services are Java IC Functional Services. E.g.: CALC Service, ACCR service, LIQD Service etc.

INFRA services and Java IC Services must be deployed on two separate Managed Servers (Any name can be given to Managed Servers).

1. **ConfigServer:** In this managed server, INFRA Services should be deployed (plato-discovery-services.war and plato-config-services.war).
2. **JavalCServer:** In this managed server, all the Java IC services should be deployed.

Following Data Sources have to be created for INFRA and Java IC Services:

Data Source JNDI Name	Type	Targets
jdbc/OBIC	Non-XA Datasource	JavalCServer
jdbc/FCUBS	Non-XA Datasource	JavalCServer
jdbc/PLATO	Non-XA Datasource	JavalCServer, ConfigServer
jdbc/PLATOBATCH	Non-XA Datasource	JavalCServer

Below line must be included in setDomainEnv.cmd or setDomainEnv.sh of the Weblogic domain:

For Linux Server:

```
JAVA_OPTIONS="${JAVA_OPTIONS} ${JAVA_PROPERTIES} -Dflyway.enabled=false
-Dspring.flyway.enabled=false -
Dplato.services.config.uri=http://<config-server-ip>:<config-server-
port> -Dplato.service.logging.path=<Debug Path where Logs are to be
written> -Dserver.id=<server id>"
```

```
export JAVA_OPTIONS
```

E.g.:

```
JAVA_OPTIONS="${JAVA_OPTIONS} ${JAVA_PROPERTIES} -Dflyway.enabled=false
-Dspring.flyway.enabled=false -
Dplato.services.config.uri=http://10.10.10.10:1010 -
Dplato.service.logging.path=/mnt/FC144/ICLogs -Dserver.id=1"
```

```
export JAVA_OPTIONS
```

For Windows Server:

```
set JAVA_OPTIONS=%JAVA_OPTIONS% %JAVA_PROPERTIES% -
Dplato.services.config.uri=http://<config-server-ip>:<config-server-
port> -Dflyway.enabled=false -Dspring.flyway.enabled=false -
Dplato.service.logging.path=<Debug Path where Logs are to be written>
-Dserver.id=<server id>
```

E.g.:

```
set JAVA_OPTIONS=%JAVA_OPTIONS% %JAVA_PROPERTIES% -
Dflyway.enabled=false -Dspring.flyway.enabled=false -
Dplato.services.config.uri=http://10.10.10.10:1010 -
Dplato.service.logging.path=D:/ICLogs -Dserver.id=1
```

Note:

The parameter mentioned below also needs to be added

```
- Dspring.cloud.loadbalancer.ribbon.enabled=false
```

server id parameter should be a number used to uniquely identify an application instance. If only one deployment of a service is present then this value has to be set to 1. In case of multiple deployment, number from 1 to the number of instances can be assigned to the server where deployment is done.

Alternatively, if the parameters are to be set specific to a Managed Server where Services are deployed, then these properties can be set in Servers->Managed Server->Server Start in the argument section. Note: It will be useful only if Node-Manager is used to start managed servers.

BEA Home:	<input type="text"/>	The BEA home directory (path on the machine running Node Manager) to use when starting this server. More Info...
Root Directory:	<input type="text"/>	The directory that this server uses as its root directory. This directory must be on the computer that hosts Node Manager. If you do not specify a Root Directory value, the domain directory is used by default. More Info...
Class Path:	<input type="text"/>	The classpath (path on the machine running Node Manager) to use when starting this server. More Info...
Arguments:	<pre>-Dflyway.enabled=false -Dspring.flyway.enabled=false - Dplato.services.config.uri=http://10.10.10:1010 - Dplato.service.logging.path=D:/ICLogs -Dserver.id=1</pre>	The arguments to use when starting this server. More Info...
Security Policy File:	<input type="text"/>	The security policy file (directory and filename on the machine running Node Manager) to use when starting this server. More Info...
User Name:	<input type="text"/>	The user name to use when booting this server. More Info...
Password:	<input type="password"/>	The password of the username used to boot the server and perform server health monitoring. More Info...
Confirm Password:	<input type="password"/>	
<input type="button" value="Save"/>		

1.4 WAR Deployment after Build

As part of FCUBS EAR build, in addition to FCUBS EAR, Java IC wars and Java IC INFRA wars will get copied into the destination location.

Below are the locations where the wars will be copied after build:

1. **FCUBS Application EAR and All Adapter EARs:** Available in the destination folder.
2. **INFRA Service WARs:** plato-discovery-services war and plato-config-services war will be available in the destination folder.
Deploy all the INFRA Service WARs in **ConfigServer**.
3. **Java IC Service WARs:** All the Java IC Service WARs will be copied in "IC" folder under the destination folder.
Deploy all the Java IC Service WARs are in **JavalCServer**.



Note:

fcubs-interest-allocate-services, fcubs-charge-calc-services and fcubs-interest-batch-services wars currently do not support multiple deployments. Only one instance of these services are to be deployed.

1.5 Order of Server Start

After deployment or server restart, services have to be started in following sequence:

- a. plato-config-service
- b. plato-discovery-service
- c. Java IC Services

When servers are restarted, ensure to start **ConfigServer** first and then then **JavalCServer**.

On every restart of **ConfigServer**, plato-discovery-service must be stopped and started. This is required as Discovery requires properties entries for self-registration to be picked from plato-config-service.

In order to check if all the services have started, below discovery URL can be checked:

<http://<config-server-ip>:<config-server-port>/plato-discovery-service>

E.g.:

<http://10.10.10.10:1010/plato-discovery-service>

All the deployed Java IC Services should get listed in the service discovery URL.

2. Java IC Maintenance in FCUBS

2.1 Required Maintenance for Java IC

Below maintenances are required in FCUBS

1. Properties Maintenance (CSDPROPM):
 - a. Launch the screen and query for entry present in LOV for Reference Number:

Unlock Authorize Enter Query

Reference Number: 10101096

Service URL: http://10.10.10.10:8080/plato-discovery-service/eureka

Service Port: 8080

Update Service Details: Service All

Service Names		Service Details	
Service Name	Service Description	Key	Value
<input checked="" type="checkbox"/> obic-charge-calc-services	obic charge calc services	<input checked="" type="checkbox"/> plato.services.eureka.uri	http://10.10.10.10:8080/plato-discovery-service/eureka
<input type="checkbox"/> obic-intchg-accntg-services	obic intchg accntg services	<input type="checkbox"/> server.port	8080
<input type="checkbox"/> obic-interest-accrual-services	obic interest accrual services	<input type="checkbox"/> plato.services.entityservices.port	8080
<input type="checkbox"/> obic-interest-allocate-services	obic interest allocate services		
<input type="checkbox"/> obic-interest-batch-services	obic interest batch services		

Maker LUJ004 Date Time: 2014-01-01 12:40:59 Mod No 42 Record Status Open
 Checker Date Time: Authorization Status Unauthorized

- b. Unlock the screen, Select All for “Update Service Details” and update the Service URL and Service Port to as below:

Key	Value
plato.services.eureka.uri eureka.client.serviceUrl.defaultZone	<a href="http://<discovery-service-ip>:<discovery-port>/plato-discovery-service/eureka">http://<discovery-service-ip>:<discovery-port>/plato-discovery-service/eureka
server.port and plato.services.entityservices.port	Managed Server port where the service is deployed

Note:

- a. Above properties are to be updated for all INFRA and Java IC Services.
- b. If SSL setup is required for calls through discovery, below properties are to be updated: (This step is not mandatory)
 - i. server.port: Update to SSL Port of the managed server where service is deployed.
 - ii. plato.services.entityservices.port: Update to SSL Port of the managed server where service is deployed.
 - iii. isSslEnabled: Value has to be set to true.
 - iv. apiProtocol: Value has to be changed to https.
 - v. nonsecure.port.enabled: value has to be false
 - vi. secure.port.enabled: value has to be true

If step b has been followed for SSL Setup and if services are deployed on different

servers, please follow section 3 (SSL Setup with Self Signed Certificate, Scenario2) for ssl handshake between fcubs-interest-batch-service managed server and other services managed server.

2. External Service Maintenance (IFDEXSER):

Prior to this step, user must maintain external system “OBIC” in CODSORCE screen.

User has to query for External System “OBIC” in IFDEXSER and following details have to be modified:

- a. Rest Service IP : The server IP where **fcubs-interest-batch-services.war** has been deployed.
- b. Rest Service Port : The Managed Server port where **fcubs-interest-batch-services.war** has been deployed.

If SSL is enabled in FCUBS properties file,

- Update Rest Service Port to SSL port of the managed server where **fcubs-interest-batch-services.war** has been deployed.
 - SSL configuration has to be done as per steps mentioned in section 3 (SSL Setup with Self Signed Certificate, Scenario1).
- c. External User: User ID of the Oracle FLEXCUBE Universal Banking user used for invoking the Java IC Services.

External Service Maintenance

New Enter Query

External System *
External System Type Default
External User *

External System AppID
Read Time Out (In Seconds)
Connection Time Out (In Seconds)
Retry Count
Archival Days
Rest Service Secured

1 Of 1 Go

Type	Service Name	WS Endpoint URL	Rest Service Context	Rest Service IP	Rest Service Port	Rest Service Path

Maker Date Time: Mod No Record Status
Checker Date Time: Authorization Status
Ok Exit

Note: Only Rest Service IP, Port, and the external user have to be updated. Values for other fields shouldn't be modified.

Maintain the details as follows to perform the maintenance in the External Service Maintenance (IFDEXSER) screen:

- **Type:** REST request
- **External System:** OBIC
- **System Type:** OBMA
- The user must specify **Rest Service IP & Rest Service Port** of the server where service is deployed.
- The user must specify the **Rest Service Context**, and **Rest Pattern** as given in the below table.

Function id	Rest Service Context	Service name	Rest Pattern
ICDOLIQ/ ICDLIQAC	obic-online-liquidation-services	OBICLiqacService	onliq/liquidateOnlineUBS
ICDCALAC	obic-interest-calc-services	OBICIntCalcService	intCalc/multipleAccountOnlineCalculation
ICDMCALC	obic-interest-calc-services	OBICIntCalcServiceM	intCalc/multipleAccountOnlineCalculation
ACDENTRY	fcubs-post-accounting-services	OBICDeferredHandoffService	postaccounting/deferredhandoffUBS

3. External Service Maintenance (IFDEXSER):

Prior to this step, the user must maintain external system “FCJAVA” in CODSORCE screen. The user has to query for External System “FCJAVA” in IFDEXSER and the following details have to be modified:

- Rest Service IP : The server IP where **fcubs-co-batch-services.war** has been deployed.
- Rest Service Port : The Managed Server port where **fcubs-co-batch-services.war** has been deployed.

If SSL is enabled in FCUBS properties file,

- Update Rest Service Port to SSL port of the managed server where **fcubs-co-batch-services.war** has been deployed.
 - SSL configuration has to be done as per steps mentioned in section 3 (SSL Setup with Self Signed Certificate, Scenario1).
- External User: User ID of the FLEXCUBE user FCUBSUSER is used for invoking the Java IC Services.

4. IC Param Maintenance (ICDPARAM):

Launch the screen and unlock and modify the parameters.

Below are the parameters which can be configured as per requirement:

PARAM_NAME	PARAM_VAL	Description
JAVA_BATCH_SLEEP_TIME	5	Sleep time in seconds to verify the status of Java IC service submitted

SKIP_OBIC_ACNTG_ERR	0	O -Mark the accounting failures and complete the EOC batch, E - Fails the EOC batch when getting any accounting failures.
IC_JOB_SLEEP_TIMER	30	Sleep time to check the status of IC parallel streams - Conventional IC
JAVA_RETRY_COUNT	150	Maximum retry count to fail the EOC batch when the submitted java service is not picked up by scheduler
RESOLVE_MAX_TRY_WITHOUT_FAIL	5	Maximum retry to obtain account lock for resolution
DB_ACNTG_COMMIT_FREQ	200	IC Accounting Commit Frequency
DB_ACNTG_FETCH_SIZE	2000	IC Accounting Fetch Size

5. PLATO_LOGGER_PARAM_CONFIG has to be updated with the log path for IC logs corresponding to LOG_PATH param value.
6. After the above maintenances, restart FCUBS Application and all the servers in the order mentioned in the section 1.5 Order of Server Start.

2.2 Scheduler Job for Triggering IC EOD in FCUBS

Scheduler Job “FCEODJ_BATCH” has to be scheduled for IC EOD in Flexcube. After the above maintenances are done, resume FCEODJ_BATCH Job from SMSJOBBER screen before triggering FCUBS EOD:

Note:

1. FCEODJ_BATCH Job Scheduler interval is set by default as 5 seconds.
2. FCEODJ_BATCH Job has been released with start-up mode as Manual. Hence after every deployment of FCUBS application or restart of server, the job needs to be manually scheduled.

3. Before triggering UBS EOD job kindly ensure that FCEODJ_BATCH Job is running.

3. SSL Setup with Self Signed Certificate

Please follow this section for below configurations:

Scenario1: If SSL is enabled for FCUBS Application, the call to fcubs-interest-batch-services will be in SSL mode. This requires SSL Configuration to be done in both the FCUBS Server and the OBIC Server.

Scenario2: If SSL is to be enabled for calls through discovery service and services are deployed on different managed server. SSL handshake would be required between fcubs-interest-batch-service managed server and other services managed server.

SSL Configuration can be done with Self Signed Certificate in non-production environment only. Since SSL Handshake would be required between the two servers, below steps can be followed in order to setup self signed SSL Setup:

1. Run the below command to create keystore for each of the servers.
To create keystore for FCUBS and OBIC server, below command has to be run. This will create keystore in the specified path:

```
keytool -genkey -keystore /path/to/keystore/server1.jks -alias  
<server1_cert_alias> -  
dname "CN=<server1_server_host>,OU=<organization>" -keyalg "RSA" -  
sigalg "SHA256withRSA" -keysize 2048 -validity <noOfDays>
```

```
keytool -genkey -keystore /path/to/keystore/server2.jks -alias  
<server2_cert_alias> -  
dname "CN=<server2_server_host>,OU=<organization>" -keyalg "RSA" -  
sigalg "SHA256withRSA" -keysize 2048 -validity <noOfDays>
```

In the above commands server1 and server2 has been used to indicate two servers involved in handshake.

For Scenario1, server1 is the server where scheduler application is deployed and server2 is the server where fcubs-interest-batch-services is deployed.

For Scenario2, server1 is the server where fcubs-interest-batch-services is deployed and server2 is each of the managed server where IC services are deployed. If SSL Configuration is already done for fcubs-interest-batch-services server, then Keystore need not be created again and only step involving certificate import to trust needs to be done.

<server1_cert_alias> Any alias name can be provided here. Alias name shouldn't be repeated.

<server2_cert_alias> Any alias name can be provided here. Alias name shouldn't be repeated.

<organization> has to be replaced with the organization code.

<noOfDays> has to be replaced with validity days of the certificate.

<server1_server_host> ip of the server1 can be mentioned here.

<server2_server_host> ip of the server2 can be mentioned here.

User will be prompted to enter password while running the above command. The same passphrase has to be entered in the further steps.

2. Configuring SSL in Weblogic server. This step has to be repeated for both the servers. Check "SSL Listen Port Enabled" Flag and enter SSL Listen Port

[Save](#)

Use this page to configure general features of this server such as default network communications.

[View JNDI Tree](#)

Name:	OBICTServer	An alphanumeric name for this server instance. More Info...
Template:	(No value specified) Change	The template used to configure this server. More Info...
Machine:	Machine-0	The WebLogic Server host computer (machine) on which this server is meant to run. More Info...
Cluster:	(Stand-Alone)	The cluster, or group of WebLogic Server instances, to which this server belongs. More Info...
Listen Address:	<input type="text"/>	The IP address or DNS name this server uses to listen for incoming connections. For example, enter 12.34.5.67 or mymachine, respectively. More Info...
<input checked="" type="checkbox"/> Listen Port Enabled		Specifies whether this server can be reached through the default plain-text (non-SSL) listen port. More Info...
Listen Port:	<input type="text" value="8030"/>	The default TCP port that this server uses to listen for regular (non-SSL) incoming connections. More Info...
<input checked="" type="checkbox"/> SSL Listen Port Enabled		Indicates whether the server can be reached through the default SSL listen port. More Info...
SSL Listen Port:	<input type="text" value="8031"/>	The TCP/IP port at which this server listens for SSL connection requests. More Info...
<input type="checkbox"/> Client Cert Proxy Enabled		Specifies whether the HttpClusterServlet proxies the client certificate in a special header. More Info...
Java Compiler:	<input type="text" value="javac"/>	The Java compiler to use for all applications hosted on this server that need to compile Java code. More Info...
Diagnostic Volume:	<input type="text" value="Low"/>	Specifies the volume of diagnostic data that is automatically produced by WebLogic Server at run time. Note that the WLDJF diagnostic volume setting does not affect explicitly configured diagnostic modules. For example, this controls the volume of

Switch to Keystores tab and follow below steps:

- Change Keystores option to "Custom Identity and Java Standard Trust"
- For Custom Identity Keystore, enter the keystore file path.
- For Custom Identity Keystore Type, enter JKS.
- For Custom Identity Keystore Passphrase and Confirm Custom Identity Keystore Passphrase, enter the passphrase entered when creating keystore.
- For Java Standard Trust Keystore Passphrase and Confirm Java Standard Trust Keystore Passphrase, enter passphrase as "changeit".

[Save](#)

Keystores ensure the secure storage and management of private keys and trusted certificate authorities (CAs). This page lets you view and define various keystore configurations. These settings help you to manage the security of message transmissions.

Keystores:	Custom Identity and Java Standard Trust Change	Which configuration rules should be used for finding the server's identity and trust keystores? More Info...
Identity		
Custom Identity Keystore:	<input type="text" value="/path/to/keystore/keystore.jks"/>	The source of the identity keystore. For a JKS keystore, the source is the path and file name. For an Oracle Key Store Service (KSS) keystore, the source is the KSS URI. More Info...
Custom Identity Keystore Type:	<input type="text" value="JKS"/>	The type of the keystore. Generally, this is JKS. If using the Oracle Key Store Service, this would be KSS. More Info...
Custom Identity Keystore Passphrase:	<input type="password" value="*****"/>	The encrypted custom identity keystore's passphrase. If empty or null, then the keystore will be opened without a passphrase. More Info...
Confirm Custom Identity Keystore Passphrase:	<input type="password" value="*****"/>	
Trust		
Java Standard Trust Keystore:	<input type="text" value="/scratch/soft/jdk1.8.0_211/jre/lib/security/cacerts"/>	The location of the java standard trust keystore. More Info...
Java Standard Trust Keystore Type:	<input type="text" value="jks"/>	The type of the java standard trust keystore. Generally, this is JKS. More Info...
Java Standard Trust Keystore Passphrase:	<input type="password" value="*****"/>	The password for the Java Standard Trust keystore. This password is defined when the keystore is created. More Info...
Confirm Java Standard Trust Keystore Passphrase:	<input type="password" value="*****"/>	

[Save](#)

- In SSL tab, do the below changes:
 - Enter Private Key Alias as entered while creating keystore.
 - For Private Key Passphrase and Confirm Private Key Passphrase, enter the passphrase entered when creating keystore.

Configuration Protocols Logging Debug Monitoring Control Deployments Services Security Notes

General Cluster Services Keystores **SSL** Federation Services Deployment Migration Tuning Overload Concurrency Health Monitoring Server Start Web Services Coherence

Save

This page lets you view and define various Secure Sockets Layer (SSL) settings for this server instance. These settings help you to manage the security of message transmissions.

Identity and Trust Locations: Keystores [Change](#) Indicates where SSL should find the server's identity (certificate and private key) as well as the server's trust (trusted CAs). [More Info...](#)

— Identity —

Private Key Location: from Custom Identity Keystore The keystore attribute that defines the location of the private key file. [More Info...](#)

Private Key Alias: FCUBSHOST The keystore attribute that defines the string alias used to store and retrieve the server's private key. [More Info...](#)

Private Key Passphrase: The keystore attribute that defines the passphrase used to retrieve the server's private key. [More Info...](#)

Confirm Private Key Passphrase: [More Info...](#)

Certificate Location: from Custom Identity Keystore The keystore attribute that defines the location of the trusted certificate. [More Info...](#)

— Trust —

Trusted Certificate Authorities: from Java Standard Trust Keystore The keystore attribute that defines the location of the certificate authorities. [More Info...](#)

— Advanced —

Save

Click "Advanced" block in the SSL Tab and set Hostname Verification to "None"

Info...

— Advanced —

Hostname Verification: None Specifies whether to ignore the installed implementation of theweblogic.security.SSL.HostnameVerifier interface (when this server is acting as a client to another application server). [More Info...](#)

Custom Hostname Verifier: The name of the class that implements theweblogic.security.SSL.HostnameVerifier interface. [More Info...](#)

Export Key Lifespan: 500 Indicates the number of times WebLogic Server can use an exportable key between a domestic server and an exportable client before generating a new key. The more secure you want WebLogic Server to be, the fewer times the key should be used before generating a new key. [More Info...](#)

Use Server Certs Sets whether the client should use the server certificates/key as the client identity when initiating an outbound connection over https. [More Info...](#)

Two Way Client Cert Behavior: Client Certs Not Requested The form of SSL that should be used. [More Info...](#)

Cert Authenticator: The name of the Java class that implements theweblogic.security.ad.CertAuthenticator class, which is deprecated in this release of WebLogic Server. This field is for Compatibility security only, and is only used when the Realm Adapter Authentication provider is configured. [More Info...](#)

SSLRejection Logging Enabled Indicates whether warning messages are logged in the server log when SSL connections are rejected. [More Info...](#)

Allow Unencrypted Null Cipher Test if the AllowUnEncryptedNullCipher is enabled [More Info...](#)

Inbound Certificate Validation: Builtin SSL Validation Only Indicates the client certificate validation rules for inbound SSL. [More Info...](#)

Outbound Certificate Validation: Builtin SSL Validation Only Indicates the server certificate validation rules for outbound SSL. [More Info...](#)

Save

4. Run below command for each of the servers to extract certificate from the corresponding keystores:

```
keytool -export -v -alias <server1_cert_alias> -file
/path/to/cert/server1.cer -keystore /path/to/keystore/server1.jks
```

```
keytool -export -v -alias <server2_cert_alias> -file
/path/to/cert/server2.cer -keystore /path/to/keystore/server2.jks
```

<server1_cert_alias> Any alias name can be provided here. Alias name shouldn't be repeated.

<server2_cert_alias> Any alias name can be provided here. Alias name shouldn't be repeated.

5. Since SSL Handshake will be done between server1 and server2, the extracted certificate from the keystore of both the servers are to be imported into Trust Store of the corresponding servers. In step 2, since Java Standard Trust has been selected as the Trust store, corresponding certificates are to be imported into Java Standard Trust of the server. server1 Certificate has to be imported into server2 Java Standard Trust and server2 certificate has to be imported into server1 Java Standard Trust in order for the SSL Handshake to be successful. Below command has to be run to import the certificate in to Java Standard Trust store "cacerts" file in the path mentioned in the "Java Standard Trust Keystore" path taken by weblogic in Step 2 for both the servers.

```
keytool -import -v -trustcacerts -alias <server1_cert_alias> -file  
/path/to/cert/server1.cer -keystore  
/path/to/jdk/jre/lib/security/cacerts
```

```
keytool -import -v -trustcacerts -alias <obic_cert_alias> -file  
/path/to/cert/obic.cer -keystore  
/path/to/jdk/jre/lib/security/cacerts
```

<server1_cert_alias> Any alias name can be provided here. As a practice, we can set hostname of the server1 server.

<server2_cert_alias> Any alias name can be provided here. As a practice, we can set hostname of the server2 server.

For Java Standard Trust Keystore, default password will be "changeit"

4. IC End of Day Batches

Following batches must be maintained for Interest processing.

For End OF Transaction Input stage following batches must be maintained in the given order. Other batches of EOC in this stage should have sequence number less than IC batch sequence number (below listed IC batches should be after all non IC batches).

EOC Group	Batch Name	Module	Frequency	Maintenance order
End OF Transaction Input	ACBCUTOF	AC	D	1
End OF Transaction Input	ICBCUTOF	IC	D	2
End OF Transaction Input	ICJRPBAT	IC	D	3
End OF Transaction Input	ICBRESOL	IC	D	4
End OF Transaction Input	TDJEOD	TD	D	5
End OF Transaction Input	ICBEOD	IC	D	6
End OF Transaction Input	ICJACPST	IC	D	7
End OF Transaction Input	DABHOFF	AC	D	8

No Other batch should be configured in between the above batches and all the batches of EOC in End OF Transaction Input stage should be of lower sequence number.

For Beginning of the day stage following batches to be maintained in the given order. Other batches of EOC in this stage should have a sequence number less than these batches (below listed IC batches should be after all non IC batches).

EOC Group	Batch Name	Module	Frequency	Maintenance order
Beginning of the Day	ICJBOD	IC	D	1
Beginning of the Day	ICJDUCOL	IC	D	2

Beginning of the Day	TDJBOD	TD	D	3
Beginning of the Day	DABHOFF	AC	D	4

5. IC Additional Setup Details

The section is to provide details regarding IC setup, so it helps teams in configuring IC with Weblogic Application Server.

5.1 IC Configurations

This section describes the required IC configurations for tuning the application.

5.1.1 IC Services: Number of Processes, Number of Deployments for a service, Commit Frequency and Fetch Count

Number of Processes spawned by IC Services, Fetch Count & Commit Frequency can be configured for IC Services.

Max Process: Number of processes to configure for a service (per deployment).

Fetch Count: Number of records picked up for execution by a process.

Commit Frequency: System will commit after processing these many number of records.

Note: Here by processes we mean the number of parallel processes spawned by the Service and not about OS level processes or threads.

e.g. If for Calc Service Max Process is 20, Fetch Count is 10000 and Commit Frequency is 200 then once Allocation Service has allocated records for Calc, processing would be as follows:

- Max Process (20) * Fetch Count (10000) = 200,000, hence maximum 200,000 record's will be updated in Allocation table for a branch (*maximum is mentioned as its possible there are less than 200,000 records available for the branch to process*), this would happen as part of a single parent process in Calc Service.
- Then 20 processes will be spawned to process records updated in previous step.
- Each Process will pick up maximum 10000 records (as Fetch Count is 10000) and process it and commit after every 200 records (as Commit Freq is 200).
- This will happen for each Calc Service deployed.

Configuring Max Process, Fetch Count & Commit Frequency.

Max Process: Value of max process can be set between 8 to 40.

Fetch Count: Fetch Count can be set between 5000 to 10000.

Commit Frequency: Commit Frequency can be set between 100 to 500, setting it to 200 is ideal for bulk processing.

5.1.1.1 Number of Deployments for a service

To Scale out IC services, multiple deployments of IC Services can be done. To achieve the same, you can configure additional managed server and deploy IC service which you want to scale (e.g. Calc is deployed on 8 managed servers & accrual is deployed on 4 managed servers etc.).

For example, if you have 4 services deployed for Calc and Max Process is set to 20 then ideally $20 * 4 =$ maximum 80 processes for Calc will be processing at any given point of time. Also, each of these deployments of Calc service (based on max process-20 in this case) will initially pick up a branch for processing (so with 4 services deployed 4 branches will be processed parallelly).

You will deploy number of services depending on the system throughput requirement. Also, rather than increasing max process beyond 40, it is better to add one more service deployment and reduce number of processes per deployment.

Note: You need not deploy 1 service for each branch. E.g. if you have 100 branches for processing and you wish to process maximum 5 branches at any given point of time then you can deploy 5 instances of service.

Interest Batch Service, Allocation Service, Resolution Service (if volume expected is less) & Charge Service (if charge volume is less) can be deployed on a single managed server. Other services like Calc, Accr, Liqd & IC accounting can be deployed either 1 or 2 services on a single managed server.

After configuring Max Process, Fetch Count & Commit Freq, if there is no performance constraint or bottleneck then to further improve throughput you can add deployment for a given service and monitor to see if throughput is increasing or not. By doing this you can reduce overall processing time.

Recommendations:

- Calc service involves heavier processing than Accr service, hence it is recommended to deploy more Calc Services than Accr Services.
- Accounting service will also need to have similar or higher deployments than Calc Service.
- If number of branches is high then it is recommended to deploy more services rather than increasing Max Process for the service, as each deployment can cater to a branch.

5.1.2 Server-Port settings in case of multiple deployments for a service

For each service, managed server's port value is set in Properties Maintenance (CSDPROP) 'server.port' key. While having multiple deployments this might require different handling, depending on the setup. Port details can be set at managed server level, in which case the entry for key 'server.port' should be deleted. To set it at managed server level, argument -Dserver.port can be set through either Server Startup Arguments or by setting it in setUserOverrides.sh.

e.g. -Dserver.port=13009 //Where by 13009 would be the managed server's port.

Check below points to identify your case and decide approach accordingly:

- If you have managed servers on different physical servers and you want to have same port for a given service on all physical servers, then you need not delete entry from properties maintenance and you can continue with same port for all deployed services (e.g. Calc Service is deployed on 8 different managed server, each on a different VM and managed server is having same port across all 8 VMs)
- In any other case, like having multiple managed servers on same physical server OR having multiple managed servers each of a different physical server but with a different port number, then you need to delete entry for 'server.port' from properties maintenance for that service and set -Dserver.port explicitly.

5.1.3 Plato Batch Logging & Persistence Logging

The default value for Plato Batch Logging & Persistence Logging for IC services is set such that it will not log details, this is to avoid overhead caused by these loggings. Properties Maintenance (CSDPROP) screen reflects these values. Value for key 'platobatch.inMemoryJobRepository' should be true so logging is only to memory and not persisted, similarly value for key 'platobatch.persistenceLogReqd' should be false so persistence logs are not written.

5.2 Weblogic Application Server Configurations

This section describes about the required Weblogic configurations.

5.2.1 Data Source Configuration

Connection Pool Statement Cache Size & Connection Pool Max Capacity should be set as recommended in Weblogic Server Best Practices document, kindly refer the same.

5.2.2 Number of deployments for IC service

Refer section [5.1.1.1 Number of Deployments for a service](#) under IC Configuration.



Java IC Installation
[November] [2022]
Version 14.6.2.0.0

Oracle Financial Services Software Limited
Oracle Park
Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India

Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax: +91 22 6718 3001
<https://www.oracle.com/industries/financial-services/index.html>

Copyright © [2007], [2022], Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.