z/OS Connect Enterprise Edition V3.0

Performance improvement with the REST client service provider for Db2 connections

Version 1.1

November 2021

Alan Hollingshead
alan_hollingshead@uk.ibm.com

z/OS Connect EE
IBM UK Laboratories
Hursley Park
Winchester
Hampshire
SO21 2JN

1.1 Notices

This report is intended for Architects, Systems Programmers, Analysts and Programmers wanting to understand the performance characteristics of z/OS Connect EE V3.0. The information is not intended as the specification of any programming interfaces that are provided by z/OS Connect EE.

It is assumed that the reader is familiar with the concepts and operation of z/OS Connect EE V3.0.

References in this report to IBM products or programs do not imply that IBM intends to make these available in all countries in which IBM operates.

Information contained in this report has not been submitted to any formal IBM test and is distributed "asis". The use of this information and the implementation of any of the techniques is the responsibility of the customer. Much depends on the ability of the customer to evaluate this data and project the results to their operational environment.

Disclaimer: All performance data this is contained in this report was obtained in the specified operating environment and configurations, and must be considered as an example. Performance characteristics of other operating environments might differ.

IBM does not represent, warrant, or guarantee that the same or similar results will be achieved in a user's environment as the results that are shown in this report.

1.1 Trademarks and service marks

1.2 Terminology

CICS	-	CICS Transaction Server
Cost per transaction (ms)	-	CPU usage per transaction, in milliseconds
CP	-	Central Processor
CPU %	-	Percentage of CPU time used by transactions running on general purpose processors
Db2	-	IBM Relational Database Management System
EE	-	Enterprise Edition
GCP	-	General purpose Central Processor
IMS	-	Information Management System
JIT optimization	-	Just-In-Time optimization. The JIT compiles the JVM bytecode to machine code at runtime and uses optimization techniques such as moving calculations and caching them, moving methods to be inline, removing unnecessary locks, thus allowing the code to become smaller and faster.
MQ	-	IBM Message Queuing
PID	-	Product Identification Number
RMF	-	Resource Measurement Facility
SMF	-	System Management Facility
SoR	-	System of Record; for example, CICS, IMS, Db2.
SP	-	(z/OS Connect EE) Service Provider
TPS	-	Number of Transactions Per Second
Think time	-	In seconds. For each client this is the time the workload driver waits after a response has been received before sending the next request.
Workload Driver	-	An application written for the purpose of these performance tests to simulate multiple simultaneous client requests
zIIP	-	IBM z Systems Integrated Information Processor

2. Overview

This report contains performance measurements for z/OS® Connect EE V3.0, program number (PID) 5655-CEE, demonstrating how the performance of the REST client service provider has been improved in **z/OS Connect EE V3.0.37**.

For this report the REST client service provider was connected to Db2, however the performance improvement made in the REST client service provider equally apply to any HTTP endpoint that it is connected to.

The scenarios used in this report focus on various sizes of JSON response payloads, however, the same performance improvements also apply to JSON request payloads.

Useful pre-requisite reading

Read the *Performance considerations* topic within the ***z/OS Connect EE V3.0 IBM Documentation*** to learn about performance best practices and how to implement these to improve overall performance within your environment.

3. Test Environment

The z/OS Connect EE test server was configured to connect to a Db2 instance using the REST client service provider, as shown in Figure 1.

The environment was installed with z/OS Connect EE 3.0.36, and then z/OS Connect EE 3.0.37, in order to compare the performance between the two releases.

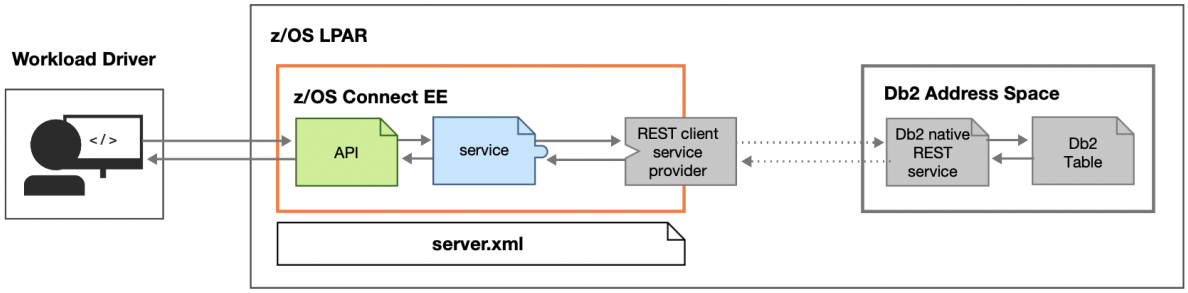

Figure 1 - End-to-end flow for the test API

4. Workload Driver

The workload driver used to simulate multiple client requests is a Java application that runs in its own JVM on the same LPAR as z/OS Connect EE. Requests issued by the workload driver specified 127.0.0.1 (i.e. “localhost”) to minimise potential network latency.

4.1 Flow of work

The flow of work for each request was as follows:

1. The workload driver sent an HTTP GET request to z/OS Connect EE over a persistent HTTP connection.
2. The GET request called the API.
3. The API mapped the GET request into a JSON payload.
4. The API called the service.
5. The service sent the request with the JSON payload to Db2.
6. Db2 sent a JSON payload in the response.
7. The HTTP response containing the JSON payload was sent back to the client.

4.2 HTTP GET Requests

All clients issued HTTP GET requests with no request body (z/OS Connect EE converts each HTTP GET request to an HTTP POST request for Db2).

4.3 Think time

Requests were issued by the workload driver at a regular pace, using a “think time”. For each client this is the time the workload driver waits after a response has been received before sending the next request. All performance runs in this report used a think time of 500ms to simulate a typical customer environment.

4.4 Performance measurement procedure

Db2 was running throughout the performance test period and contained pre-loaded data.

For each performance run the following steps were carried out:

1. z/OS Connect EE was started with the “clean” option.
2. The workload was started, and several hundred thousand requests were run to ensure the JIT was optimized before any metrics were gathered.
3. IBM z/OS Resource Measurement Facility (RMF) data was collected at one-minute intervals.

4.5 Performance metrics

The following metrics were gathered during the performance runs:

- API request rate (calculated as “transactions per second”)
- CP (general processor) usage

- Potential zIIP usage
- CPU cost per transaction (calculated)

CP and potential zIIP usage values were extracted from RMF workload activity data. RMF reports CPU utilisation that is based on uniprocessor capacity, which means that a value of 100% represents the capacity of one processor.

Figure 2 shows an extract from an RMF Workload Activity report for a z/OS Connect EE server. For the one-minute interval, the APPL% section shows 19.94% of general processor time was used (the "CP" value) and that 19.52% (the "IIPCP" value) is potentially offloadable to zIIP processors.

The request rate (transactions per second) values were gathered by the workload driver.

```
                                        W O R K L O A D   A C T I V I T Y

        z/OS V2R3              SYSPLEX MV22              DATE 09/10/2020          INTERVAL 01.00.000   MODE = GOAL
                               RPT VERSION V2R3 RMF      TIME 13.33.06

                                        POLICY ACTIVATION DATE/TIME 07/22/2020 15.53.43

POLICY=NORMAL                          REPORT CLASS=ZOSCONN
                                       DESCRIPTION =zosConnect reports

-TRANSACTIONS--  TRANS-TIME HHH.MM.SS.FFFFFF  TRANS-APPL%-----CP-IIPCP/AAPCP-IIP/AAP  ---ENCLAVES---
AVG        1.00  ACTUAL                   0  TOTAL        19.94       19.52    0.00  AVG ENC    0.00
MPL        1.00  EXECUTION                0  MOBILE        0.00        0.00    0.00  REM ENC    0.00
ENDED         0  QUEUED                   0  CATEGORYA     0.00        0.00    0.00  MS ENC     0.00
END/S      0.00  R/S AFFIN                0  CATEGORYB     0.00        0.00    0.00
#SWAPS        0  INELIGIBLE               0
EXCTD         0  CONVERSION               0
                 STD DEV                  0

----SERVICE----   SERVICE TIME  ---APPL %---  --PROMOTED--  --DASD I/O---  ----STORAGE----  -PAGE-IN RATES-
IOC           0   CPU   11.733  CP     19.94  BLK    0.000  SSCHRT    0.0  AVG    231428.1  SINGLE      0.0
CPU      873187   SRB    0.231  IIPCP  19.52  ENQ    0.000  RESP      0.0  TOTAL  231428.1  BLOCK       0.0
MSO           0   RCT    0.000  IIP     0.00  CRM    0.000  CONN      0.0  SHARED    21.00  SHARED      0.0
SRB       17207   IIT    0.000  AAPCP   0.00  LCK    0.006  DISC      0.0                   HSP         0.0
TOT      890394   HST    0.000  AAP      N/A  SUP    0.000  Q+PEND    0.0
/SEC      14840   IIP    0.000                              IOSQ      0.0
ABSRPTN     15K  AAP      N/A
TRX SERV    15K
```

Figure 2 – Extract from RMF Workload Activity report

4.6 Hardware

- IBM System z13 2964-NE1 model 7A5
- 10GB of Central Storage (RAM)
- LPAR with 6 dedicated GCPs (no zIIPs)
- OSA-Express5S 10GB Ethernet

4.7 Software

- z/OS V2.3

- z/OS Connect Enterprise Edition (EE) V3.0.36
  - z/OS Connect EE build 20200817-1534
  - Liberty 20.0.0.6 with Angel V14

- z/OS Connect Enterprise Edition (EE) V3.0.37
  - z/OS Connect EE build 20200911-1125
  - Liberty 20.0.0.6 with Angel V14

- z/OS Connect EE API toolkit V3.0.8

- IBM 64-bit SDK for z/OS Java Technology Edition, Version 8.0
  - Java 1.8 SR6 FP11

- IBM Db2 for z/OS V12

5. Db2 Database

The scenario for this performance report was based around the Db2 sample Employee table, DSN8C10.EMP, with new rows added to increase the number of rows that could be returned in a request. The client could request details about employees that matched particular criteria, such as the work department.

Figure 3 shows an extract of some new rows added to the Db2 Employee table.

EMPNO	FIRSTNME	MIDINIT	LASTNAME	WORKDEPT	PHONENO	HIREDATE	JOB	EDLEVEL	SEX	BIRTHDATE	SALARY	BONUS	COMM
100000	CAMERON	S	SALISBURY	W01	3076	1992-10-19	FIELDREP	30	M	1970-06-24	31368.00	1940.00	3727.00
100010	DAISY	W	SALISBURY	W01	9378	1998-10-01	OPERATOR	30	F	1978-11-20	28258.00	4410.00	3851.00
100020	BRADLEY	F	PEEL	W01	9537	2000-02-07	SALESREP	32	M	1978-07-11	20819.00	1938.00	2981.00
100030	HOPE	G	DURHAM	W01	3628	2003-09-14	SALESREP	30	F	1983-10-02	31338.00	4078.00	1515.00
100040	GEORGIE	D	WELLS	W01	5287	1985-08-11	FIELDREP	30	F	1965-06-03	26238.00	4594.00	1812.00
100050	ABBY	C	PEEL	W01	8104	1995-05-25	CLERK	32	F	1963-03-04	19782.00	3903.00	2703.00
100060	BRADLEY	D	ELY	W01	0236	1991-08-04	FIELDREP	32	M	1962-03-19	29971.00	4568.00	1128.00
100070	DAISY	W	RIPON	W01	7349	1995-10-27	ANALYST	30	F	1966-01-16	33627.00	1299.00	1555.00
100080	EMMA	D	YORK	W01	3139	2014-10-02	CLERK	32	F	1991-11-11	16119.00	847.00	432.00
100090	ABBY	T	ELY	W01	3985	1995-11-22	DESIGNER	32	F	1965-03-05	30015.00	997.00	99.00
100100	ALAN	X	HOLLS	W01	1279	1987-09-07	SALESREP	30	M	1965-07-13	41874.00	3980.00	2850.00
100110	HOPE	J	ELY	W01	0228	1997-03-05	FIELDREP	30	F	1974-07-25	28713.00	3424.00	4848.00
100120	DOTTIE	B	RIPON	W01	6030	2011-10-13	OPERATOR	30	F	1970-07-19	19618.00	4418.00	3707.00
100130	DAISY	B	SALISBURY	W01	8166	2008-11-07	SALESREP	34	F	1982-04-01	16068.00	2831.00	1848.00
100140	GEORGIE	C	WELLS	W01	4441	2014-09-11	ANALYST	34	F	1984-03-12	23658.00	665.00	3709.00
100150	CAMERON	L	YORK	W01	0349	2012-01-05	DESIGNER	30	M	1994-10-03	29307.00	4396.00	1193.00
100160	CAMERON	C	WELLS	W01	5475	2020-07-01	ANALYST	30	M	2001-04-03	27062.00	1097.00	598.00
100170	BRAD	G	SALISBURY	W01	1659	2020-08-21	ANALYST	30	M	2000-03-20	21766.00	672.00	2683.00

Figure 3 - Extract from Db2 Employee table

6. Db2 Native REST Service

A Db2 native REST service called employeeList was created using the following HTTP POST command from a REST client:

```
https://db2host.ibm.com:32100/services/DB2ServiceManager
```

with header:

```
Content-Type: application/json
```

and request body:

```
{
   "requestType": "createService",
   "sqlStmt": "SELECT EMPNO AS \"employeeNumber\", FIRSTNME AS \"firstName\", LASTNAME
AS \"lastName\", WORKDEPT AS \"workDept\" FROM DSN81210.EMP WHERE WORKDEPT LIKE
:workDept ORDER BY LASTNAME",
   "serviceName": "employeeList",
   "description": "Returns a list of employees.",
   "collectionID": "SYSIBMSERVICE"
}
```

The SQL statement selects employees from the Db2 table DSN81210.EMP and includes a WHERE clause for WORKDEPT so this can be included as an input for mapping in the API toolkit in “8. API Definition”.

7. Db2 Service

Using the API toolkit, a Db2 service called *employeeList* was created. The Db2 native service, *employeeList,* created in “6. Db2 Native REST Service”, was imported by selecting the “Import from Db2 service manager…” button.

(Note that API toolkit V3.0.7/V3.2.7 or later is required to create Db2 services.)

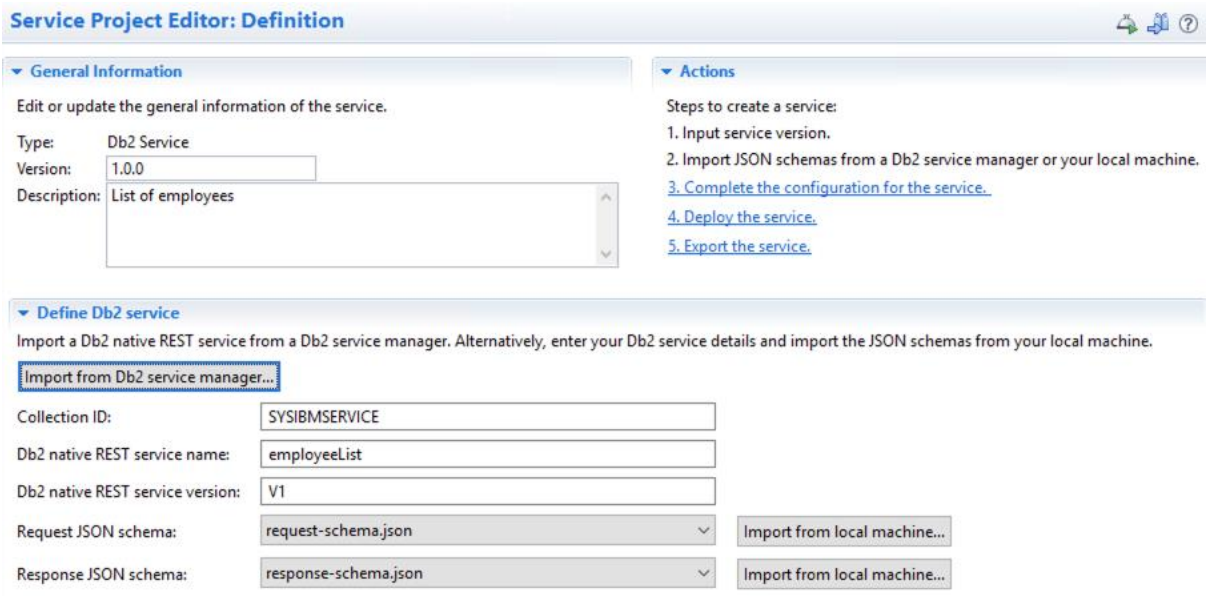

Figure 4 - Db2 service definition in API toolkit

The connection reference on the “Configuration” tab (not shown in Figure 4) was set to “db2Connection” to match the setting defined in the server.xml configuration file (in “ 10.3 The server.xml Configuration File”).

8. API Definition

Using the API toolkit, an API called *projectmanager* was created, as shown in Figure 5.

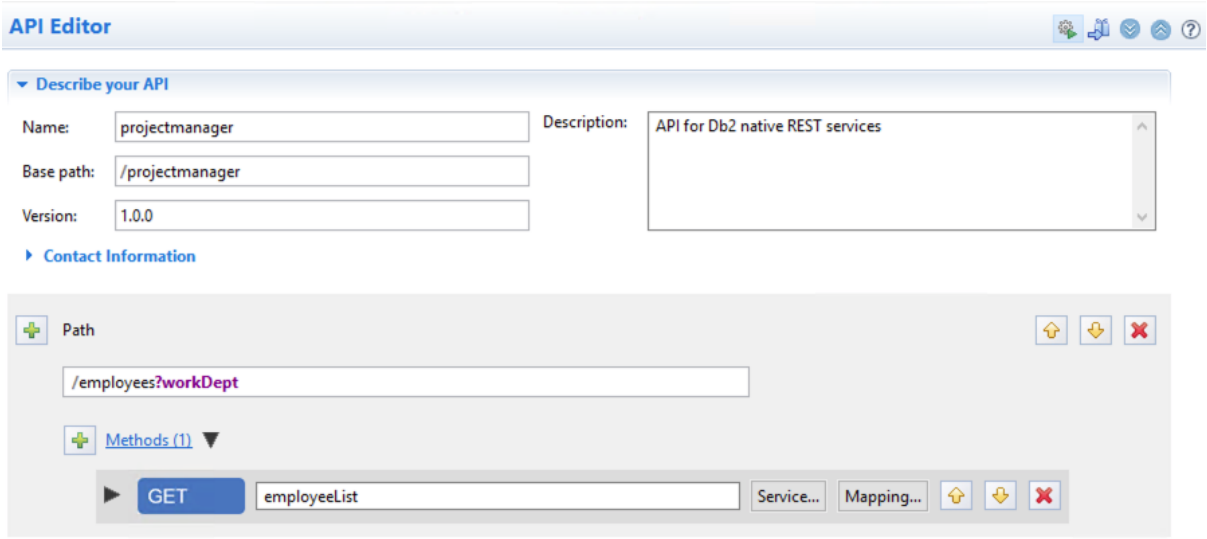

Figure 5 - Definition for API "projectmanager"

To allow for different response payload sizes, the API took a query parameter of *workDept* which was mapped to *workDept* in the Db2 request payload, as shown in Figure 6.

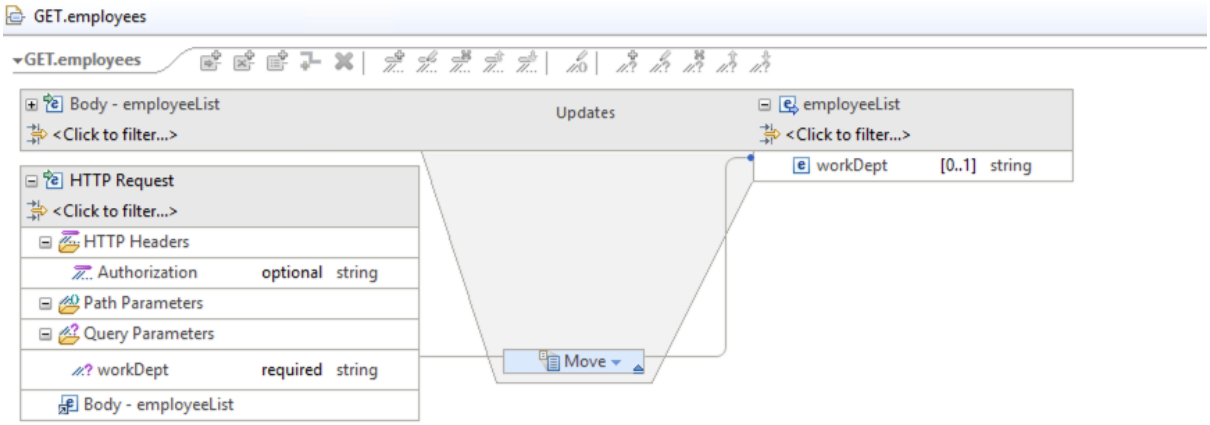

Figure 6 - Mapping of workDept query parameter

9. Calling the API

All API calls from the workload driver specified a URL in the following format:

```
http://<host>:<port>/<basepath>/<relative  path>?query_parm1
```

For example:

```
http://myHost:2222/projectmanager/employees?workDept=W%25
```

Note: the example results in a query parameter of “W%” (the %25 is specified to escape the wildcard % sign in a URL).

10. z/OS Connect EE Environment

This section describes in detail the z/OS Connect EE environment the performance tests were run in.

10.1 JCL Configuration for z/OS Connect EE

The started procedure for z/OS Connect EE was configured with:

- REGION size 0M
- MEMLIMIT 4G

10.2 JVM Configuration

Details of the JVM used by z/OS Connect EE:

- IBM 64-bit SDK for z/OS Java Technology Edition, Version 8.0
  - Java 1.8 SR6 FP11
- Maximum Java heap size (Xmx) 1G
  - Other Java heap (Garbage Collector) values set to minimise the impact of the heap variation during performance runs:
    - -Xms1G to match the maximum Java heap size
    - -Xmnx512M to set the maximum new (nursery) size
    - -Xmns512M to set the minimum new (nursery) size
    - -Xmox512M to set the maximum old (tenured) size
    - -Xmos512M to set the minimum old (tenured) size
- All other JVM system properties used default values (including garbage collection mode: gencon)

10.3 The server.xml Configuration File

The server.xml file used for the z/OS Connect EE server contained the following definitions:

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="REST client connection to Db2. Polling not enabled.">

    <config updateTrigger="mbean"></config>

    <!-- Enable features -->
    <featureManager>
        <feature>zosconnect:zosConnect-2.0</feature>
        <feature>zosconnect:zosConnectCommands-1.0</feature>
    </featureManager>

    <!-- ZosConnectManager definition -->
    <zosconnect_zosConnectManager
          requireSecure="false"
          requireAuth="false"
          operationMode="ASYNC">
    </zosconnect_zosConnectManager>

    <!-- REST client connection to Db2 with basic authentication -->
    <zosconnect_zosConnectServiceRestClientConnection
          id="db2Connection"
          host="db2host.ibm.com" port="42100">
             <basicAuth userName="db2user" password="db2userPassword"/>
    </zosconnect_zosConnectServiceRestClientConnection>

    <!-- zosConnect APIs -->
    <zosconnect_zosConnectAPIs updateTrigger="mbean"/>

    <!-- zosConnect services -->
    <zosconnect_services updateTrigger="mbean"/>

    <!-- applicationMonitor is not applicable for z/OS Connect EE servers -->
    <applicationMonitor updateTrigger="disabled" dropinsEnabled="false"/>

    <!-- Server port -->
    <httpEndpoint id="defaultHttpEndpoint"
          enabled="true"
          host="127.0.0.1"
          httpPort="2222" />
</server>
```

Figure 7 - The server.xml configuration

Note that polling was not enabled in the server.xml configuration file as this can have an impact on performance and is not recommended in a production environment. This includes proactively disabling the applicationMonitor which is not used by z/OS Connect EE servers, but if unspecified, polling is enabled by default by Liberty.

The REST client connection included basic authentication credentials to connect to Db2. For the purposes of this performance report, no other security was enabled.

11. Scenarios

Each client called the *projectmanager* API to obtain data from Db2 of the employees assigned to a particular department.

Using wildcards, for example “W%” for all departments beginning with “W”, allowed different API requests to return different sized payload responses.

The employees in the database were assigned to one of the following departments: C01, T05, W01, W03, W50

Scenario (JSON response payload size)	**Number of Db2 rows returned**	**Query string “workDept” value**	**Departments used**
1K response	4	C%	C01
5K response	21	T%	T05
10K response	42	W01%	W01
30K response	126	W0%	W01 and W03
50K response	210	W%	W01, W03 and W50

Table 1 - Summary of scenarios, rows returned and JSON response payload sizes.

12. Results

The following sections of this report show the results of the performance runs for each scenario, comparing z/OS Connect EE V3.0.36 with V3.0.37.

Disclaimer: All performance data in this report was obtained in the specified operating environment and configurations, and must be considered as an example. Performance characteristics of other operating environments might differ.

IBM does not represent, warrant, or guarantee that the same or similar results will be achieved in a user's environment when compared to the results that are shown in this report.

12.1 Transactions Per Second

For each scenario the performance runs simulated:

- 100 clients
- 200 clients
- 300 clients

Using a think time of 500ms the rate of transactions per second (TPS) was maintained for each run for all the JSON response payload sizes (1K to 50K):

	100 clients	200 clients	300 clients
TPS:	200	400	600

13. CPU cost per transaction

The cost per transaction in CPU terms is measured in milliseconds (ms).

The calculation is made by dividing the CPU service time by the number of seconds in the SMF interval; this is then divided by the number of transactions per second (TPS) for the same SMF interval; and finally multiplied by one thousand to get the result in milliseconds.

The CPU cost per transaction was the same for any given scenario, regardless of the number of clients running concurrently (for details, see 15. Scalability on page 24), therefore only the data for 200 clients is shown in this section.

CPU cost per transaction for each scenario

For each scenario the CPU cost per transaction was measured.

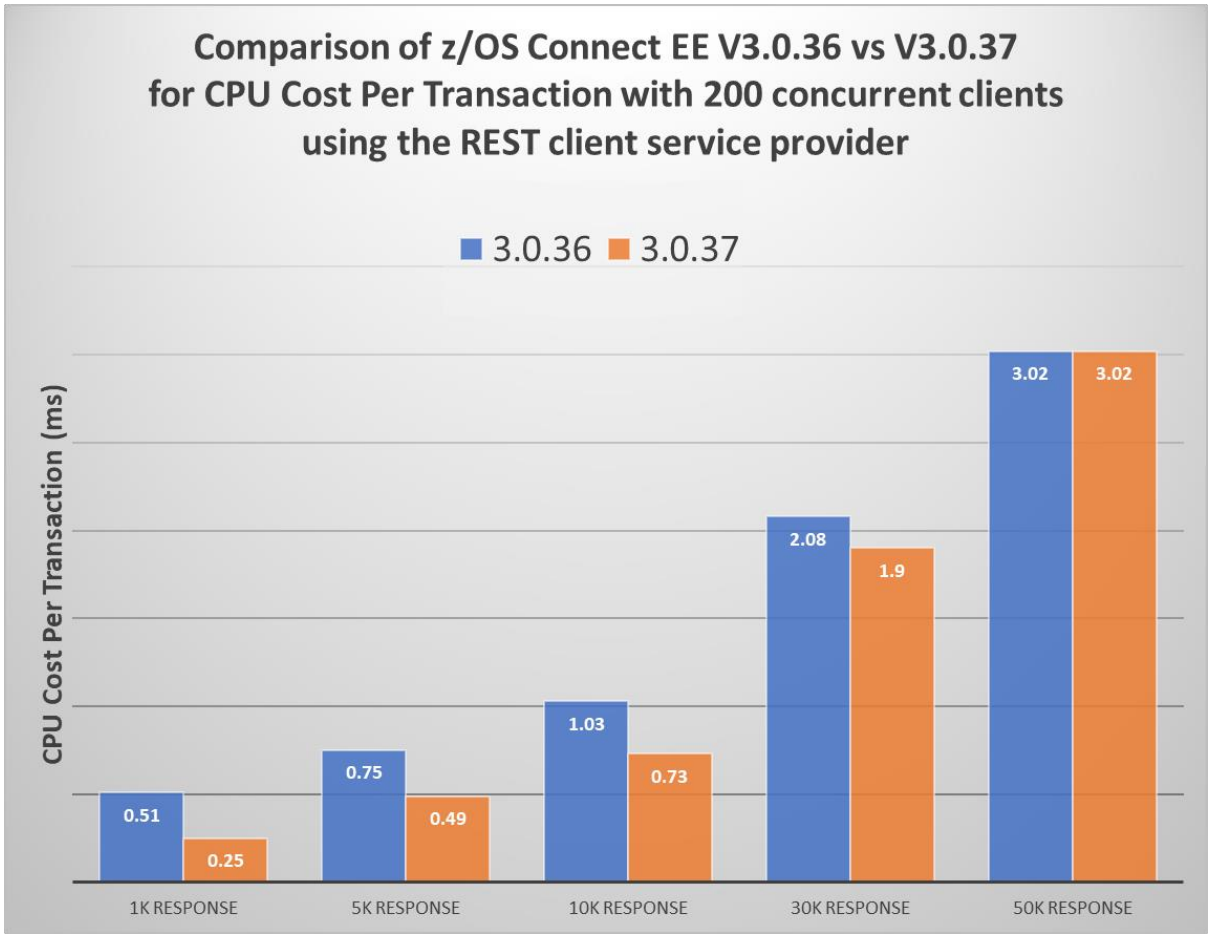

Figure 8 - CPU cost per transaction comparing z/OS Connect EE V3.0.36 with V3.0.37

Performance improvements in terms of percentage

Using the same set of results for 200 concurrent clients, Table 2 on page 21 shows the performance improvement in CPU cost per transaction in terms of percentage.

Scenario	CPU cost per transaction (ms): z/OS Connect EE V3.0.36	CPU cost per transaction (ms): z/OS Connect EE V3.0.37	% Improvement in CPU cost per transaction
1K response	0.51	0.25	104%
5K response	0.75	0.49	53%
10K response	1.03	0.73	41%
30K response	2.08	1.90	9%
50K response	3.02	3.02	0%

Table 2 - Percentage improvement for CPU cost per transaction

Observations

- ✓ Smaller payloads benefit from a greater performance.
- ✓ For the smaller JSON response payload sizes, 1K, 5K, 10K, the performance improvement between z/OS Connect EE V3.0.36 and V3.0.37 showed a considerable improvement in the CPU cost per transaction.
- ✓ For the JSON response payload size of 30K, the performance improvement is less noticeable. This is because the benefits gained from the optimisations in the service provider are exceeded by the additional processing required to handle larger messages.
- ✓ For the largest JSON response payload size tested, which was 50K, there is no performance benefit gained by the optimisations in the service provider due to the additional processing required to handle messages of this size, or larger.
- ✓ The improvement in CPU cost reduces with payload size.

14. CPU % usage and zIIP eligibility

The zIIP eligibility values were obtained from the SMF 72 records.

As z/OS Connect EE V3.0 is almost entirely written in Java the zIIP eligibility was observed to understand the benefits of offloading work to zIIP engines.

The environment is configured with six GCPs. Although physical zIIPs have not been used, zIIP eligibility was captured and is shown alongside the actual GCP usage.

Figure 9 shows the CPU % usage and zIIP eligibility for the different JSON payload response sizes.

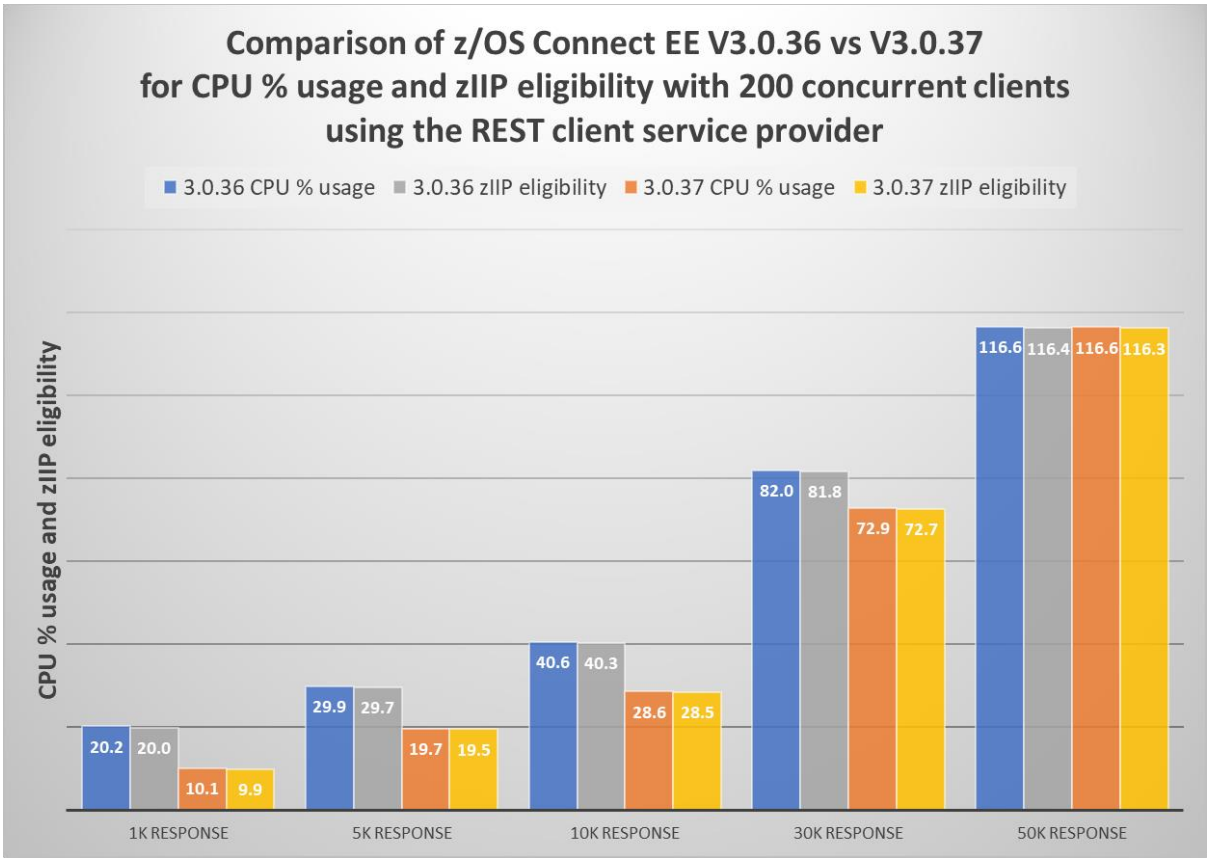

Figure 9 - CPU % usage and zIIP eligibility for each JSON payload response size

Observations

- ✓ Smaller payloads benefit from a greater performance.
- ✓ For the smaller JSON response payload sizes, 1K, 5K, 10K, the performance improvement in the REST client service provider between z/OS Connect EE V3.0.36 and V3.0.37 showed that less CPU processing was required with V3.0.37.
- ✓ For the JSON response payload size of 30K, the performance improvement is less noticeable. This is because the benefits gained from the optimisations in the service provider are exceeded by the additional processing required to handle larger messages.
- ✓ For the largest JSON response payload size tested, which was 50K, there is

no performance benefit gained by the optimisations in the service provider due to the additional processing required to handle messages of this size, or larger.

- ✓ z/OS Connect EE is a Java-based product, and the majority of the product is eligible to be offloaded to zIIP. Native code is not zIIP eligible.
- ✓ It is important to ensure that the zIIP engines do not become overloaded with work. When this happens, zIIP eligible requests are sent back to run on a GCP which is inefficient and costly in terms of overall CPU processing required.
- ✓ The improvement in CPU cost and zIIP usage reduces with payload size.

15. Scalability

Tests were run to see how z/OS Connect EE scaled with an increasing number of clients to see if there was an impact requesting more rows from the database and therefore returning larger JSON response payload sizes. The results compared z/OS Connect EE V3.0.36 with V3.0.37.

15.1 CPU cost per transaction for increasing number of clients

In each scenario, using a think time of 500ms to gain consistent throughput (transactions per second), the CPU cost per transaction was observed. Figure 10 shows the results for 10K JSON response payloads (42 rows returned from Db2).

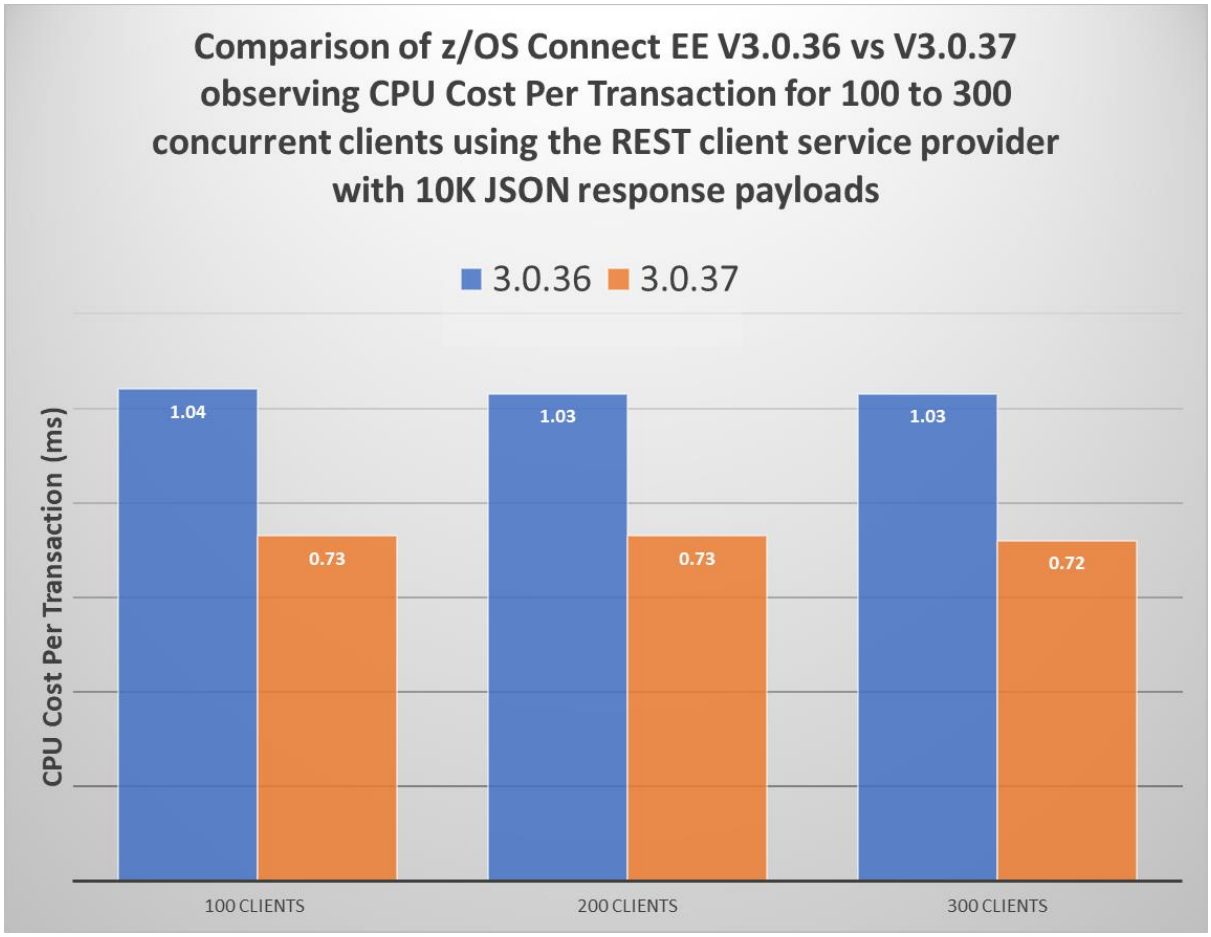

Figure 10 - CPU cost per transaction for increasing number of concurrent clients

Observation

- ✓ Figure 10 demonstrates that z/OS Connect EE scaled well with the CPU cost per transaction remaining constant at approximately 1.03ms for z/OS Connect EE V3.0.36 and 0.73ms for z/OS Connect EE V3.0.37 and was not affected by the number of clients increasing from 100 up to 300 clients.

15.2 CPU % usage for increasing number of clients

The CPU % usage will naturally increase as the number of clients running in parallel increases along with the increased number of transactions per second.

The CPU % includes all the GCPs (six in this environment), allowing a theoretical maximum of 600% for all products (including z/OS Connect EE, Db2 and the workload driver, which all ran on the same LPAR).

Figure 11 shows the CPU % usage and zIIP eligibility for the z/OS Connect EE server for 100 to 300 clients running concurrently with 10K JSON response payloads (42 rows returned from Db2).

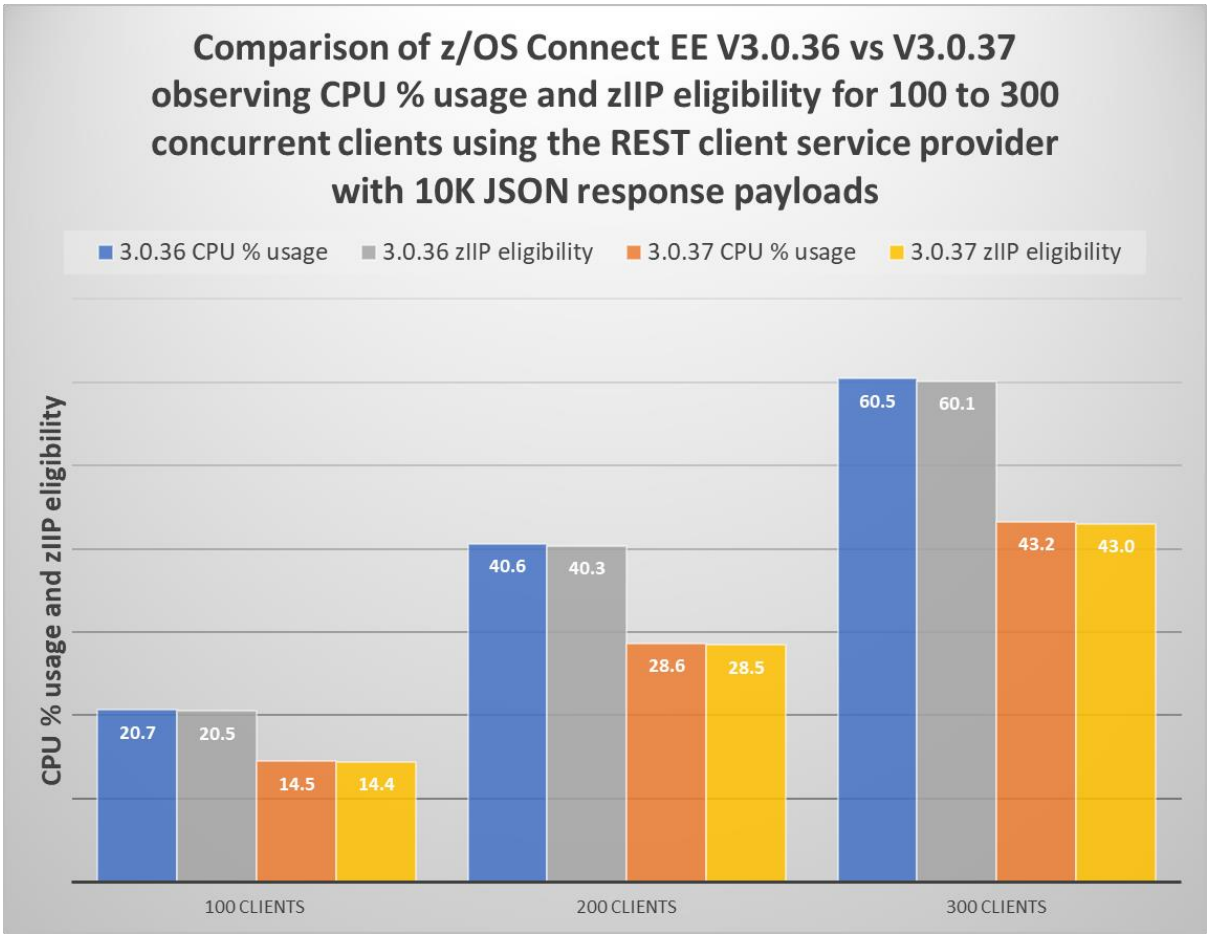

Figure 11 - CPU% usage for increasing number of clients and transaction rates

Observations

- ✓ z/OS Connect EE demonstrated good scalability.
- ✓ The CPU % usage increased linearly for both z/OS Connect EE V3.0.36 and V3.0.37 as the number of clients running concurrently were increased.

16. Conclusions

The following conclusions can be drawn from this report:

1. z/OS Connect EE V3.0.37 (or later) demonstrates improved performance over z/OS Connect EE V30.36 when using the REST client service provider, particularly for smaller JSON payload sizes such as 1K, 5K, 10K. The CPU usage is reduced and therefore the CPU cost per transaction is also reduced.

2. z/OS Connect EE V3.0 scales well when
   - ✓ the number of concurrent clients increases
   - ✓ the JSON payload sizes increase.

3. The majority of z/OS Connect EE V3.0 is written in Java and so benefits can be made by offloading work to a sufficient number of zIIP engines.

Note:

1. Analysis of other payload sizes, datatypes, or running on different hardware, have not been completed at this time, so there is no guarantee that equivalent observations will be seen in other configurations.

2. Due to the effects on system performance of machine hardware, levels of software configuration and variations in payload, equivalent observations might not be seen on other systems.