SHARE
EDUCATE • NETWORK • INFLUENCE

TECH375

Open Telemetry Tracing in CICS

Eric Higgins
(erichiggins@us.ibm.com)
CICS Advisory Specialist - WSC

Disclaimer

IBM's statements regarding its plans, directions and intent are subject to change or withdrawal without notice at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

IBM Confidential. Unless specifically advised otherwise, you should assume that all the information presented in CICS Design Forum sessions and contained in any presentations (whether given in writing or orally) is IBM Confidential and restrict access to this information in accordance with the IBM Feedback Agreement.

Content Authority. The workshops, sessions and materials have been prepared by IBM or the session speakers and reflect their own views. They are provided for informational purposes only, and are neither intended to, nor shall have the effect of being, legal or other guidance or advice to any participant. While efforts were made to verify the completeness and accuracy of the information contained in this presentation, it is provided AS-IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation or any other materials. Nothing contained in this presentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

Performance. Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Customer Examples. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

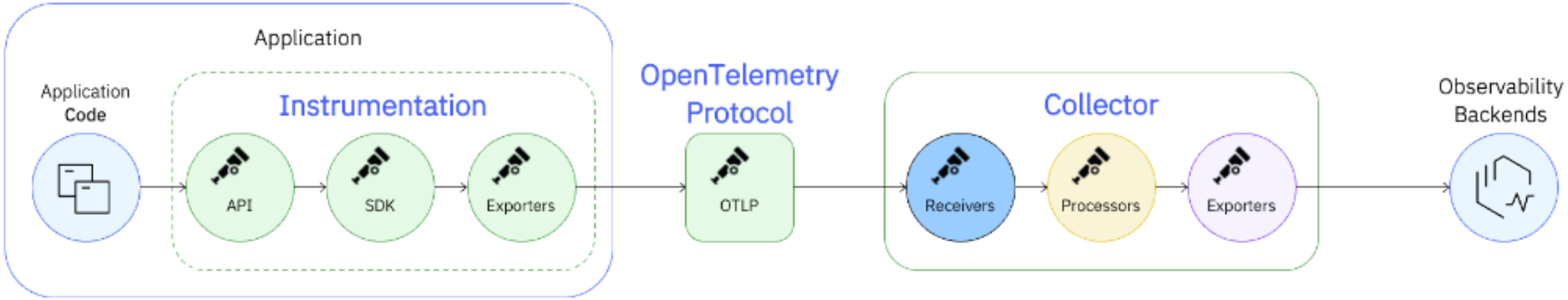
Availability. References in CICS Design Forum sessions and any presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates.

Trademarks. IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at http://www.ibm.com/legal/copytrade.shtml

Agenda

- What is Open Telemetry? A brief introduction
- Why implement Open Telemetry (OTel)?
- Open Telemetry trace for CICS
- Updates available in Open Beta

What is open telemetry?



Open telemetry is a vendor-agnostic observability framework that assists in generating processing and distributing telemetry data such as metrics, logs and traces.

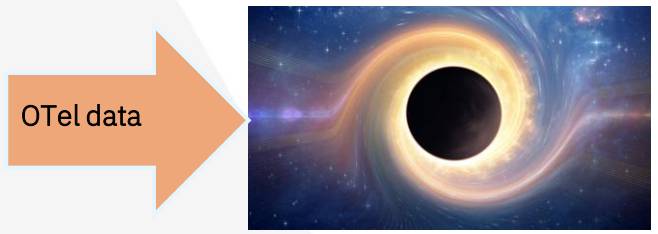
Note : Open Telemetry trace is NOT related to CICS trace

Why OpenTelemetry?

- OpenTelemetry is acknowledged as a cost-effective open framework & protocol for instrumenting cross-platform applications.
- It enables observation of system state (the monitoring aspect).
- It allows the detection of problems (failures and non-optimal operations).
- It helps to identify problem location quickly and with the minimum of fuss (cross-platform/cross-application logging).
- It enables the isolation of where a problem is (the trace aspect).
- There is no proprietary lock-in.
- It's open

Why Open Telemetry tracing?

- To isolate the location of a problem, in order to fix it or decide which team should fix it, it is necessary to see how requests move through a distributed application.
- This may involve complex flows between both distributed products and products on z/OS.
- Currently, there is no way to see the request flow when a distributed transaction goes to z/OS, as Open Telemetry data are not generated on the platform.
  - *The z/OS platform is like a black hole, data goes into it, but nothing comes out ...*
- The goal is to make CICS participate in the global OTel trace framework as an equal contributor with the same capability as distributed products.



A Journey

I travelled to a conference

How did I get there?

Journey to conference

Taxi | Flights | Taxi

Taxi to airport

Flights to Pittsburgh

Taxi to Hotel

How did I get there?

Journey to conference

Taxi | Flights | Taxi

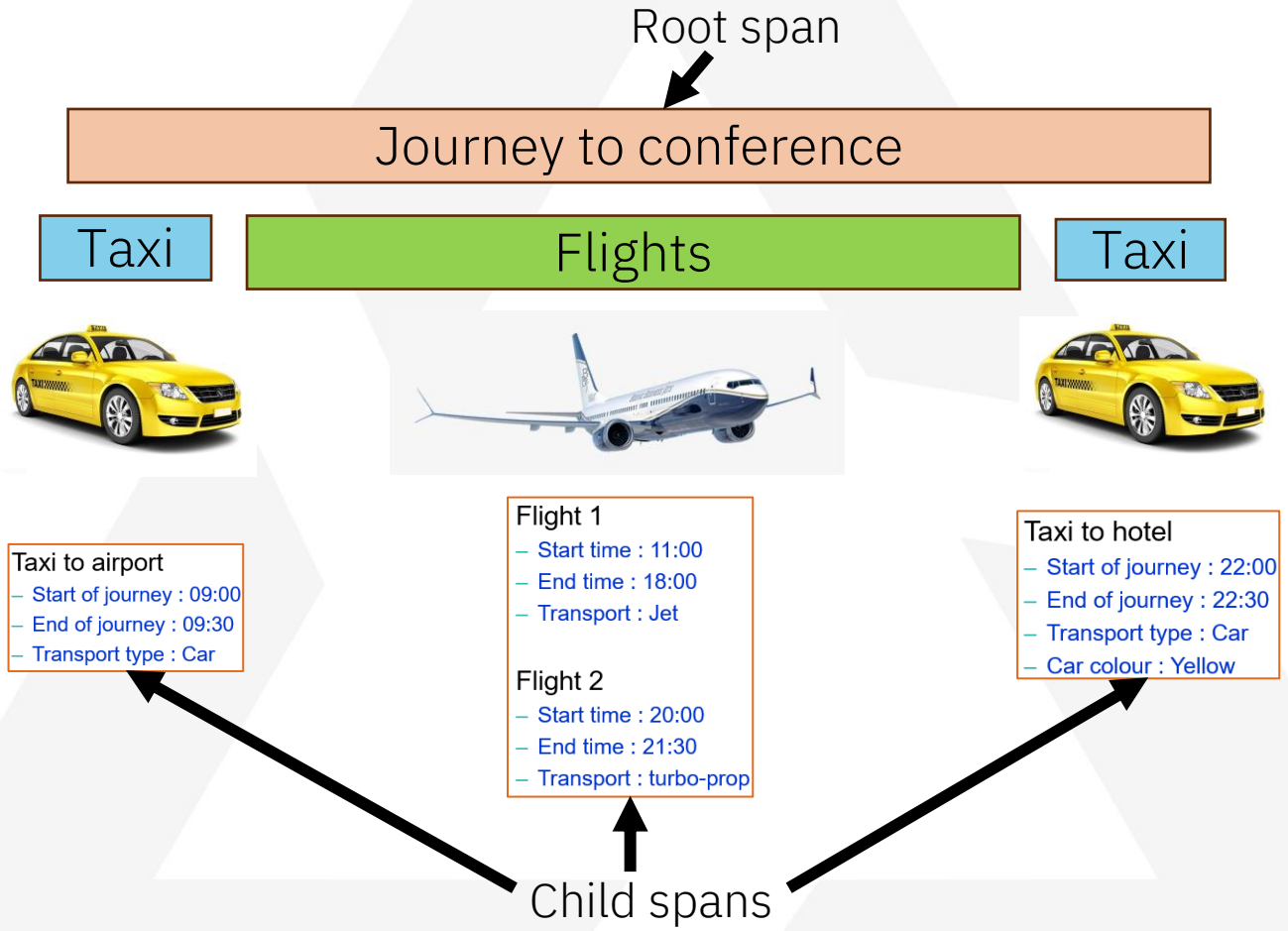
Taxi to airport
- Start of journey : 09:00
- End of journey : 09:30
- Transport type : Car

Flight 1
- Start time : 11:00
- End time : 18:00
- Transport : Jet

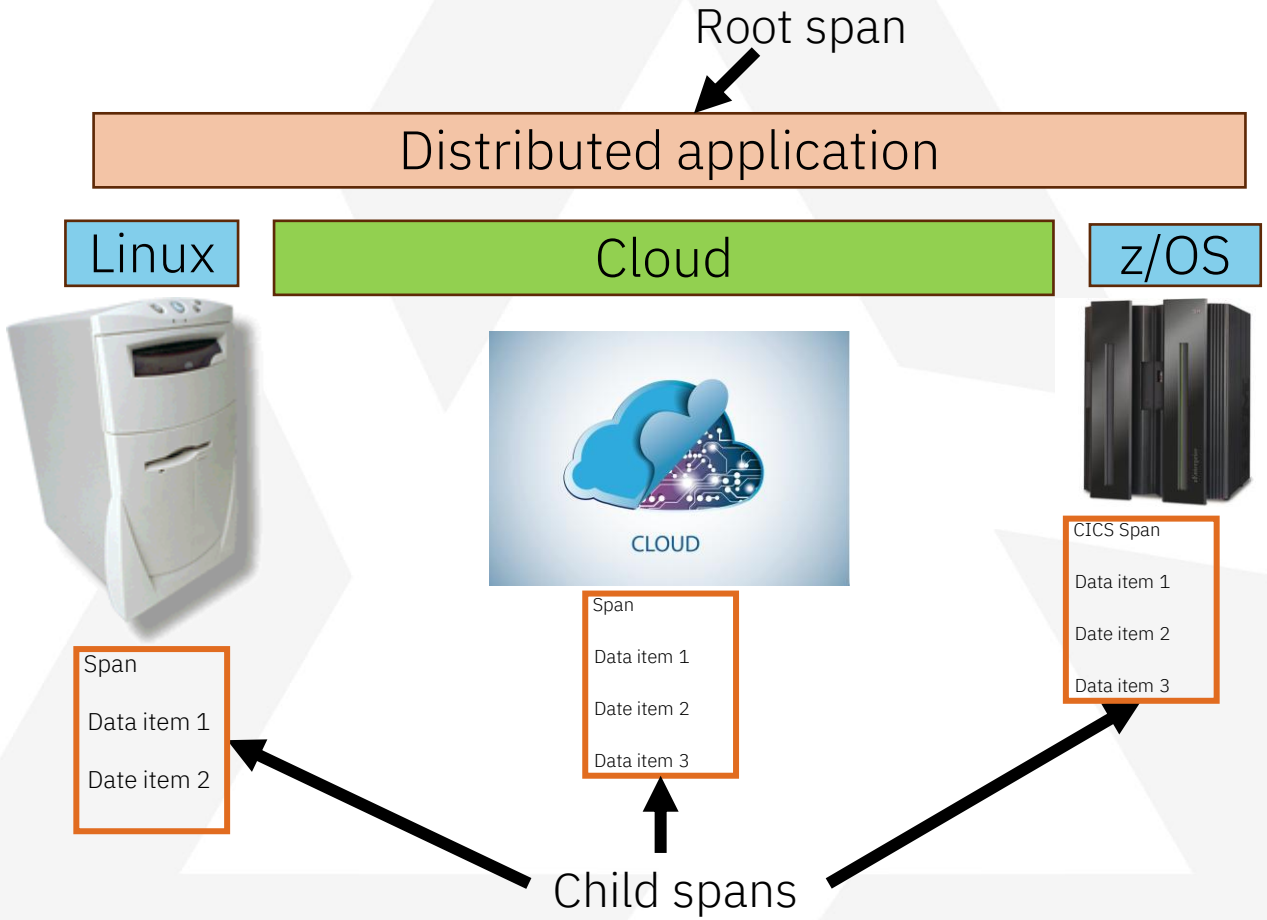
Flight 2
- Start time : 20:00
- End time : 21:30
- Transport : turbo-prop

Taxi to hotel
- Start of journey : 22:00
- End of journey : 22:30
- Transport type : Car
- Car colour : Yellow

How did I get there?

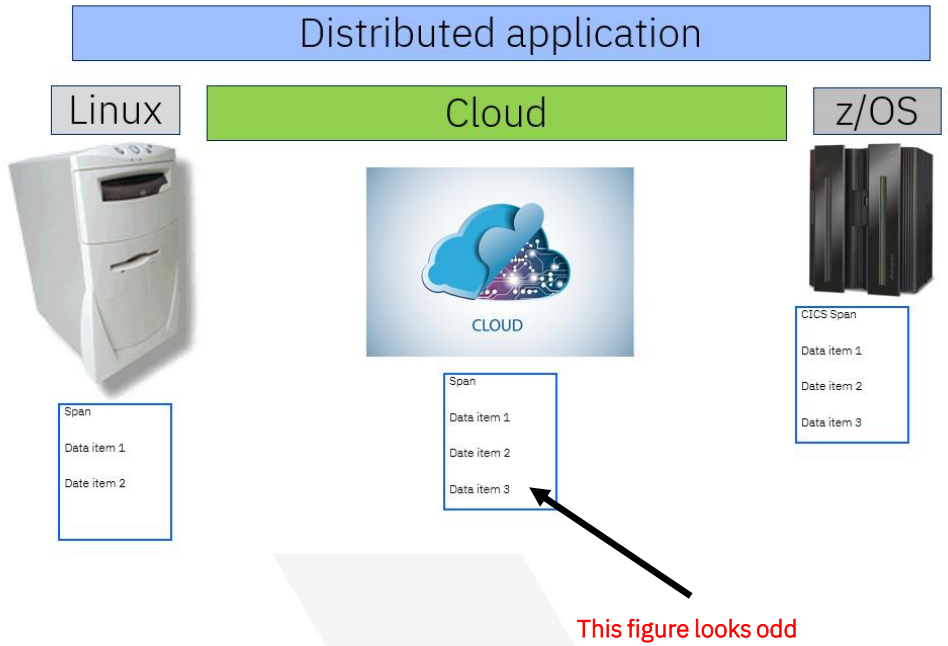


How did the application get there?



What do spans give you?

- Spans aim to provide a way to identify **where** a problem is in a distributed application.
  - Help to identify which team needs to be called in to resolve it?
- Spans should reduce the "**time to innocence**" when a problem arises in a complex distributed application.



What do spans not give you?

Span data will not generally be sufficient to diagnose the root cause of a problem in most cases

CICS Span

Data item 1

Date item 2

Data item 3

Conventional diagnostic tools are still needed to diagnose root cause.

In general spans indicate WHERE but not WHAT a problem is

```
TCP   QR    DS 0005 DSSR  EXIT  WAIT_OLDW/OK
TCP   QR    XM 0801 XMSR  ENTRY INQUIRE_MXT
TCP   QR    XM 0802 XMSR  EXIT  INQUIRE_MXT/OK        3B,0
TCP   QR    AP FCE2 ZFRE  ENTRY FREEMAIN              RPL BUFFERLIST LUC 2AE72670,T193
TCP   QR    AP FCE3 ZFRE  EXIT  FREEMAIN/OK           2AE9A030
TCP   QR    AP 4D00 CQCQ  ENTRY MERGE_CIB_QUEUES
TCP   QR    AP 4D01 CQCQ  EXIT  MERGE_CIB_QUEUES/OK
TCP   QR    AP 4D00 CQCQ  ENTRY GET_CIB
TCP   QR    AP 4D01 CQCQ  EXIT  GET_CIB/EXCEPTION     CIB_QUEUE_EMPTY,00000000,00000000
TCP   QR    DS 0004 DSSR  ENTRY WAIT_OLDW             TCP_NORM,00042540,CSTP,NO,IDLE,DFHZDSP
TCP   QR    DS 0005 DSSR  EXIT  WAIT_OLDW/OK
TCP   QR    XM 0801 XMSR  ENTRY INQUIRE_MXT
TCP   QR    XM 0802 XMSR  EXIT  INQUIRE_MXT/OK        3B,0
TCP   QR    AP FD14 ZRAC  EVENT RECV_ANY              2AE72670,T193,ZSDX
TCP   QR    AP FCE0 ZGET  ENTRY GETMAIN               RPL 2AE72670,T193
TCP   QR    AP FCE1 ZGET  EXIT  GETMAIN/OK            2AE9A030
TCP   QR    AP FC90 VIO   EVENT TCTTE(2AE72670)       IYCWT193,0020,RECEIVE,DATA,RQE1,OIC,BB,CD
TCP   QR    SM 0C01 SMMG  ENTRY GETMAIN               100,2AE72670,NO,TERMINAL
TCP   QR    SM 0C02 SMMG  EXIT  GETMAIN/OK            2AE3C6E0
TCP   QR    AP FD11 ZATT  ENTRY ATTACH                2AE72670,T193
```

Open Telemetry Trace context

How are the spans connected?

Journey to conference

Taxi | Flights | Taxi

Taxi to airport
- Start of journey : 09:00
- End of journey : 09:30
- Transport type : Car

Flight 1
- Start time : 11:00
- End time : 18:00
- Transport : Jet

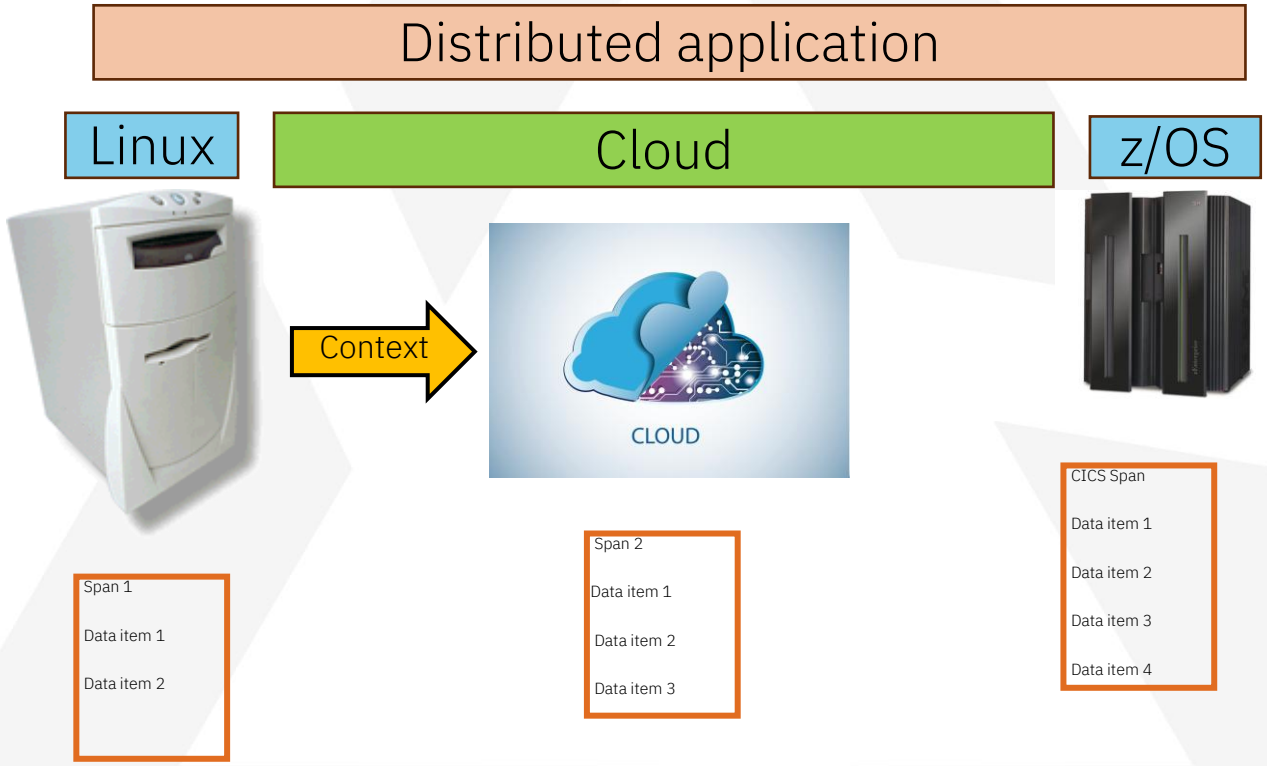
Flight 2
- Start time : 20:00
- End time : 21:30
- Transport : turbo-prop

Taxi to hotel
- Start of journey : 22:00
- End of journey : 22:30
- Transport type : Car
- Car colour : Yellow

The common factor is me.

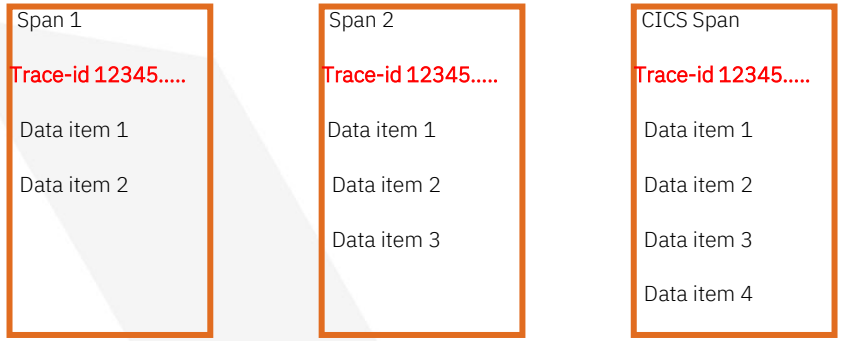
I travelled on all these forms of transportation.

What is the common factor here?



Trace Context

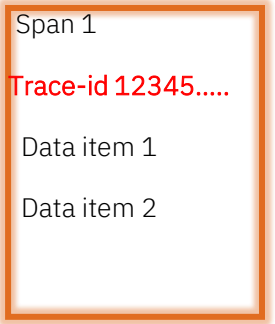
- Context contains what is known as the **traceparent**
  - this identifies the incoming request in a tracing system
  - it is 55 bytes long and contains various fields, one of which is the **trace-id**
    - the trace-id
      - this is the ID of the whole trace “forest” and is used uniquely to identify a distributed trace through a system
      - it is 32 bytes long in printable format
      - it never changes during distribution between systems
  - The trace-id is used by the collector to identify spans which belong to the same trace



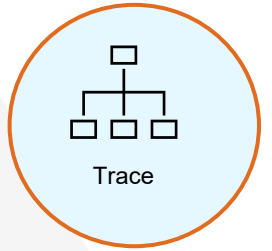
https://www.w3.org/TR/trace-context/

What are OTel Spans and traces?

- Spans are blocks of data giving information about what is happening
  - Spans can contain whatever a product decides to include along with some mandatory standard items



- **Traces** in OpenTelemetry are defined implicitly by their **Spans**. In particular, a **Trace** can be thought of as a graph of **Spans**, where the edges between **Spans** are defined as parent/child relationship.

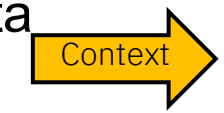


- Traces allow visibility of the full “path” a request takes in an application

Open Telemetry tracing for CICS

Support for OTel trace

- To participate in Open Telemetry trace, CICS needs to :-
  - Be capable of receiving and propagating open telemetry trace context data



  - Be capable of generating and issuing spans

CICS Span

Data item 1

Date item 2

Data item 3

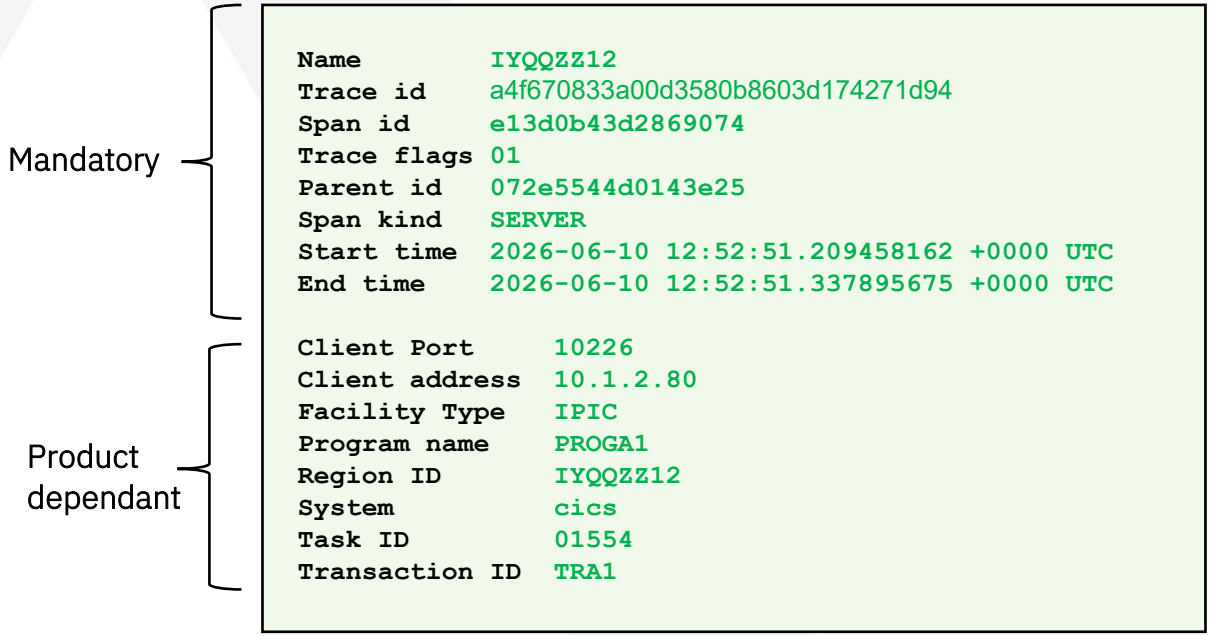
An Open Telemetry span

- Spans contain information about the implementation of an application
  - Mandatory data
    - Start time
    - End time
    - Type of span
    - A trace ID (all child spans contain the traceid of the root span)
  - Product specific content
    - Attributes chosen by the system/product issuing the span

https://opentelemetry.io/docs/specs/otel/trace/api/#span

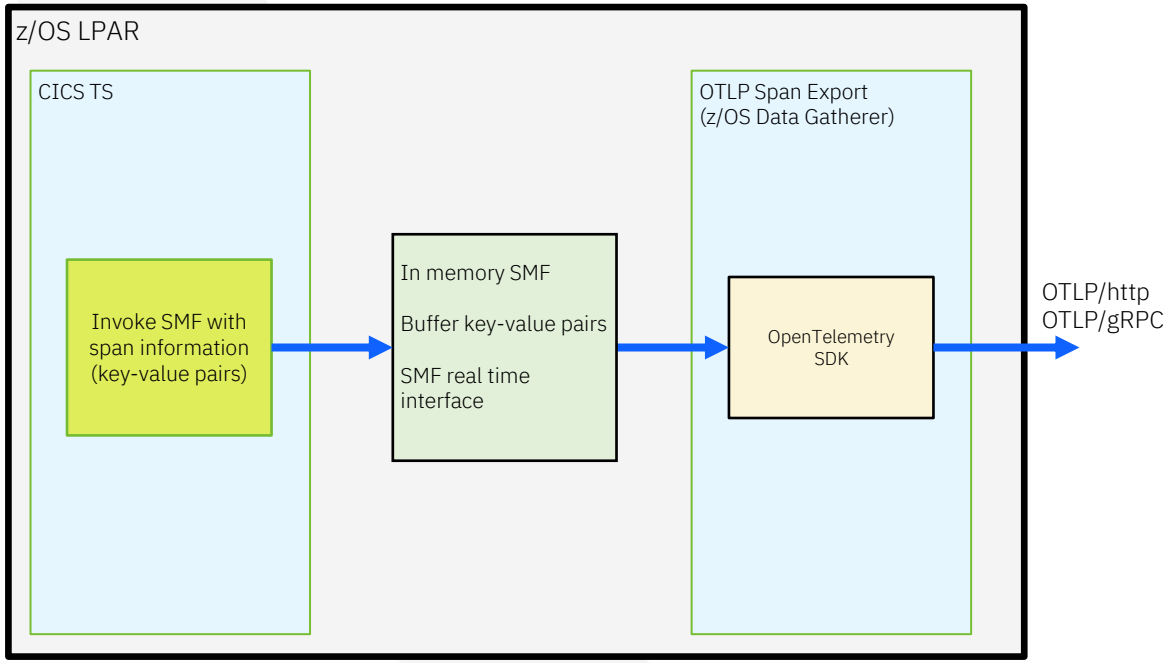
What an OTel span emitted by CICS will contain?

- Spans must contain certain “mandatory” items
- Spans will contain CICS specific items



Where do the spans go?

- Span data are emitted by CICS
  - Emission is at end of task
- Written to SMF using the real time interface
  - New record type 1159 for CICS OTel output
- Read out of SMF by an “emitter” service
  - APAR OA66345 for z/OS 3.1 or 3.2
- Exported via OTLP/http or OTLP/gRPC



Span data

- Most span data are collected at transaction attach time in CICS
  - Stored in a control block for each task
- Span data are completed at task termination time
  - End time, abend code etc.
- Emitted to SMF using the usual CICS SVC call
  - CICS already uses SMF for monitoring records (110 records)
  - OTEL data uses a new SMF type 1159 record
    - Each z/OS product has a unique record type for it's OTEL data

Scenarios where CICS supports OTEL tracing?

- Protocols over which OTel data into and out of a CICS region are supported.
  - HTTP
  - SOAP
  - REST
  - DPL over IPIC and MRO
  - Function shipping over IPIC and MRO
  - JCICS
  - MQ
  - Db2
  - More being worked on

Trace context propagation

- To get an OTel trace, the context needs to be propagated
  - For inbound HTTP requests, there is an architected header which will be added by client systems
  - For CICS-to-CICS propagation over protocols like IPIC and MRO, CICS will use a container or a TIOA to pass the data

CICS Configuration for OTel

CICS configuration requirements

- Two things need configuration in CICS for OTel
  - Trace context propagation
    - Allows CICS to pass trace context to downstream systems
  - Span data emission
    - Enables CICS to participate in data emission so that CICS spans form part of a complete OTel trace

OTel tracing configurations in CICS – Region level

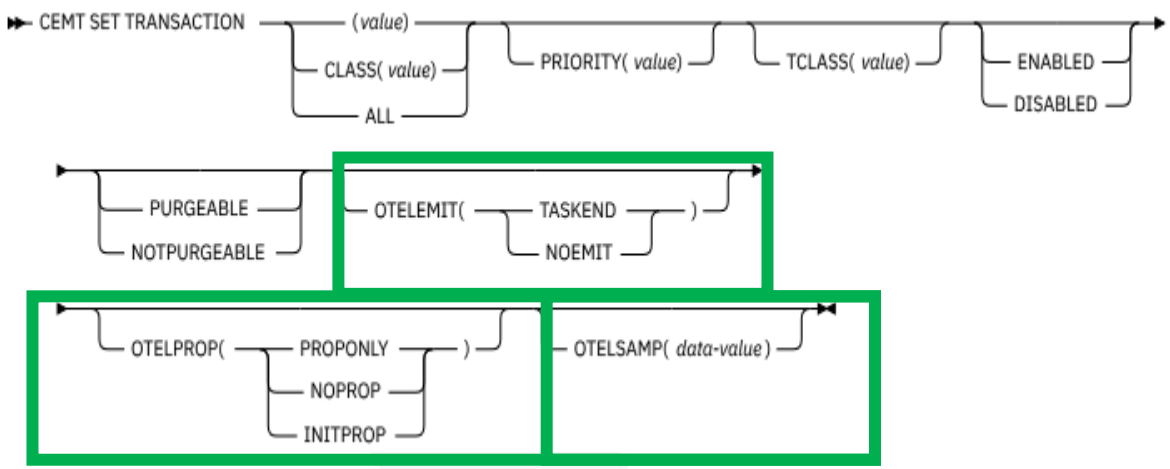
- SIT parameter OTELTRACE=(YES|NO)
- Default to NO
- With NO - tasks in CICS region do not participate OTEL tracing, regardless of the transaction setting. This provides for a predictable upgrade experience.
- With YES – non-system tasks in the CICS region are eligible to participate OTEL tracing subject to transaction setting.

- The region level setting can be inquired and changed via a new SPI:
  - INQUIRE OTEL TRACE
  - SET OTEL TRACE(NOOTELTRACE|OTELTRACE)
    - Setting from OTELTRACE to NOOTELTRACE, any accumulated span data for tasks already completed are written immediately to SMF; any active task span data will not be accumulated.

OTel tracing configurations in CICS – Transaction level

OTELPROP

- Controls propagation of OTel trace context
  - **Noprop** – OTel trace context is not propagated
  - **Proponly** – OTel trace context is propagated (Default)
  - **INITPROP** – CICS initiates OTel trace context for this transaction if no context is received beta



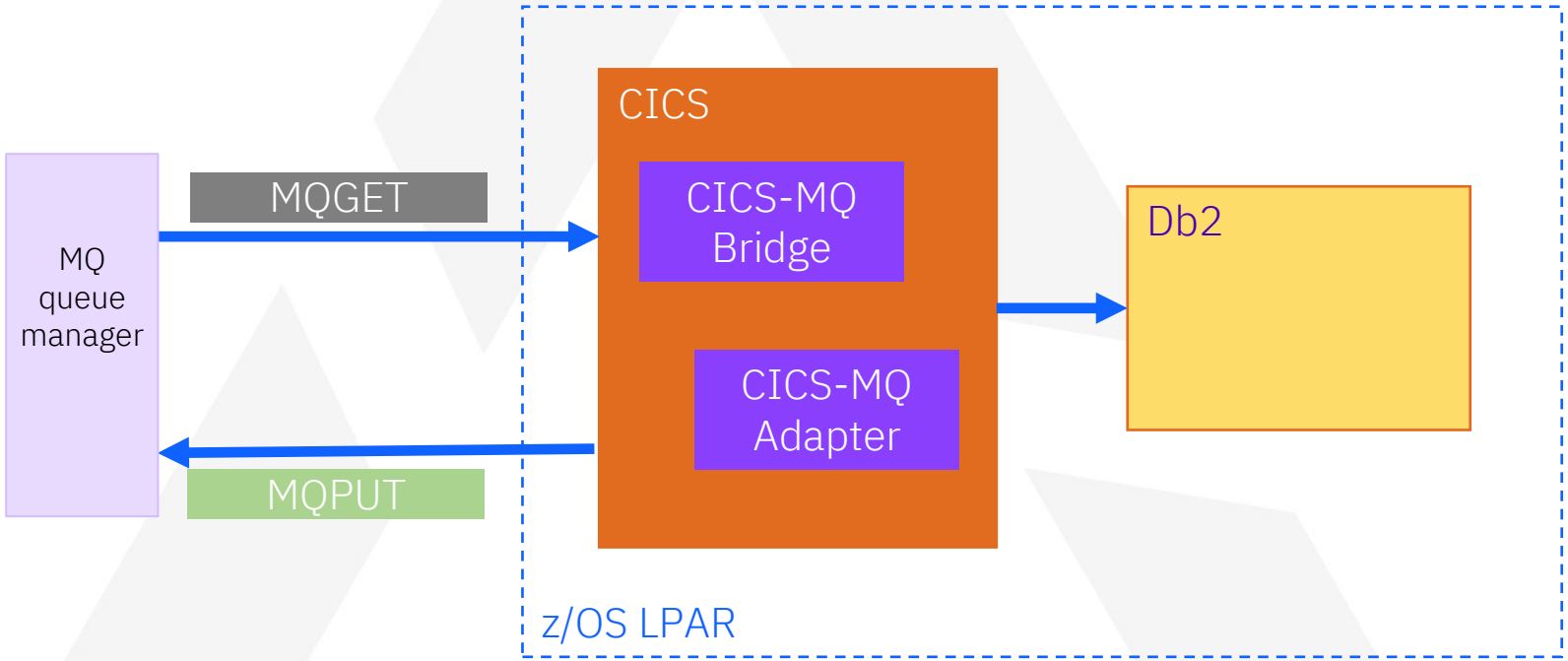
OTELEMIT

- Controls whether CICS is to emit span data
- Can be set to Noemit or Taskend
  - **Noemit** – CICS will not emit span data to SMF
  - **Taskend** – CICS will emit span data to SMF at end of task (Default)

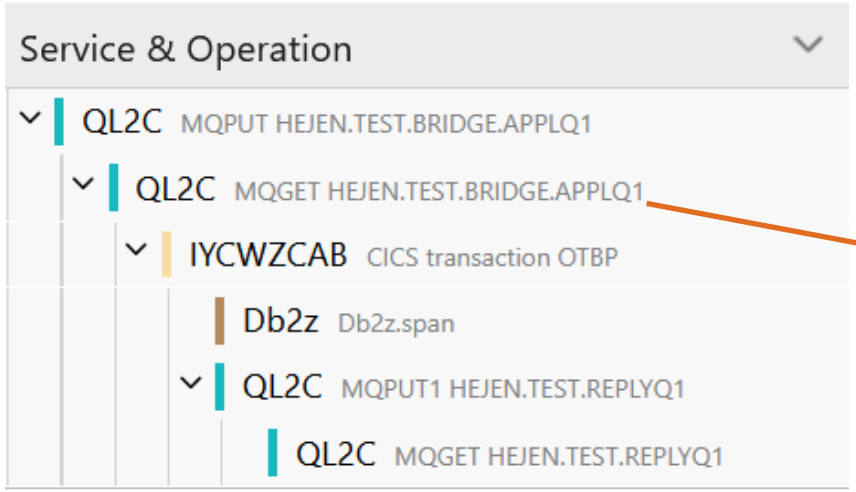
OTELSAMP beta

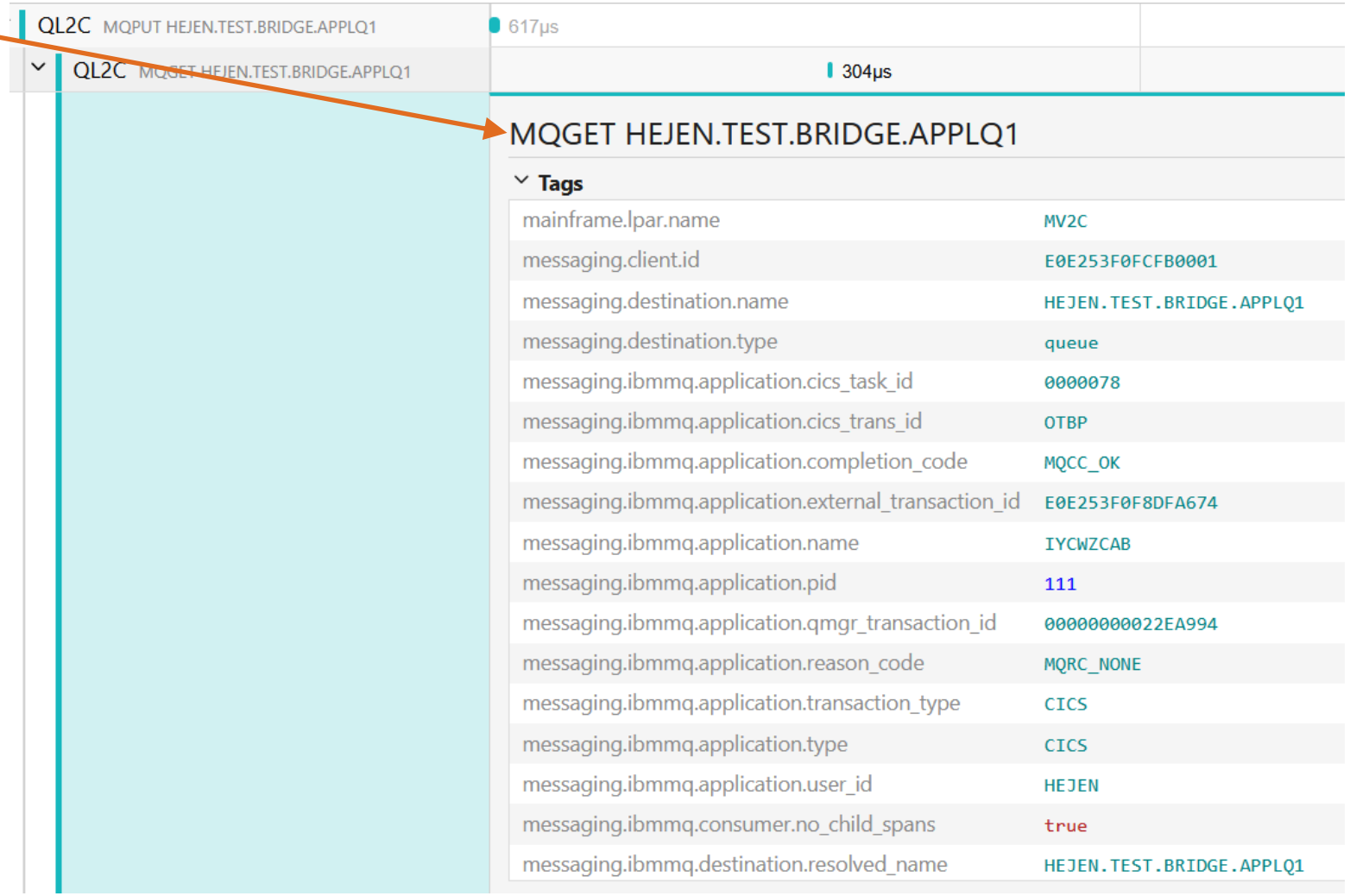
- Set the sampling probability percentage (0-100) for OTel trace context initiation when **OTELPROP** is set to **INITPROP**

OTel Tracing – CICS, MQ and Db2 scenario

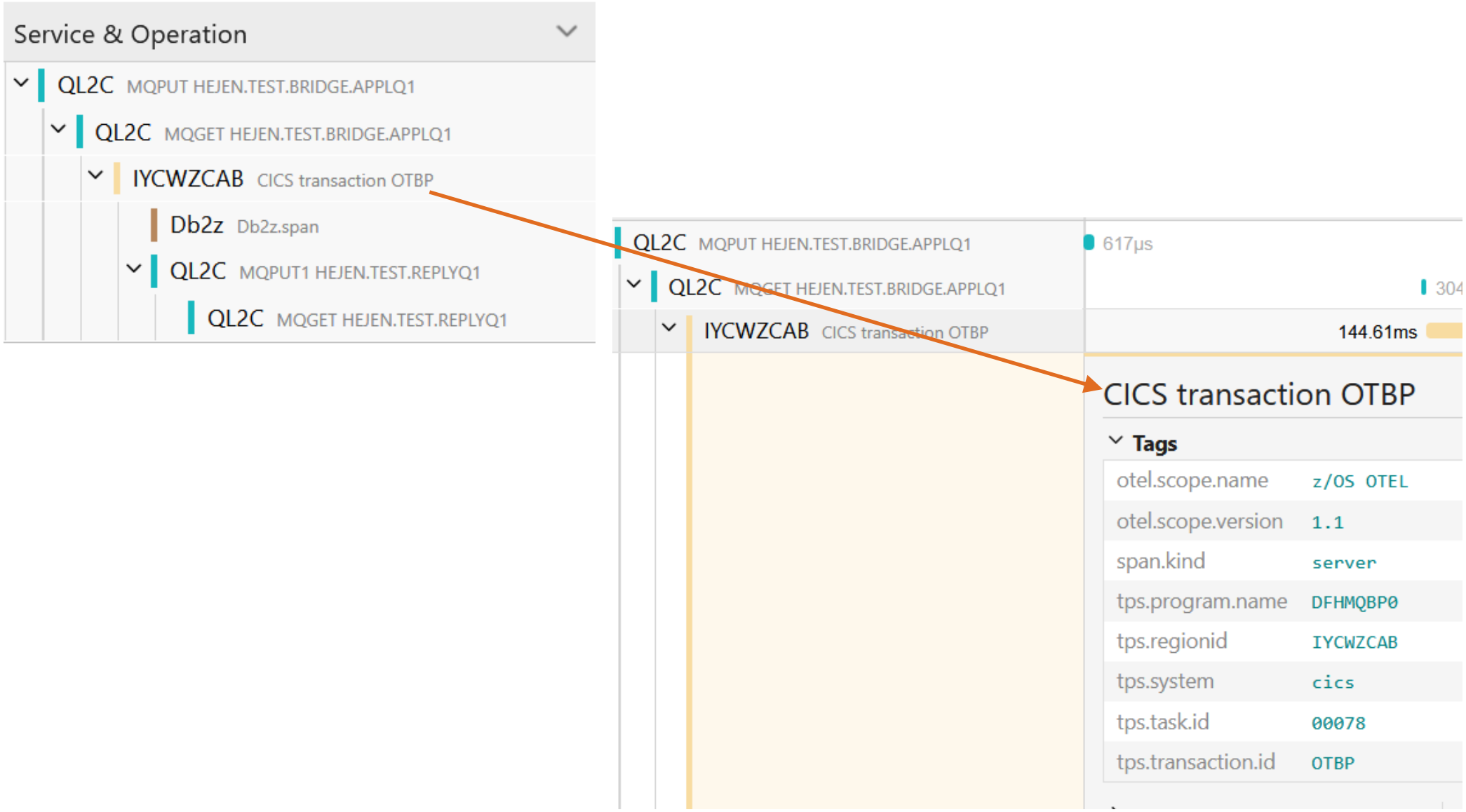


OTel Tracing – CICS, MQ and Db2 scenario

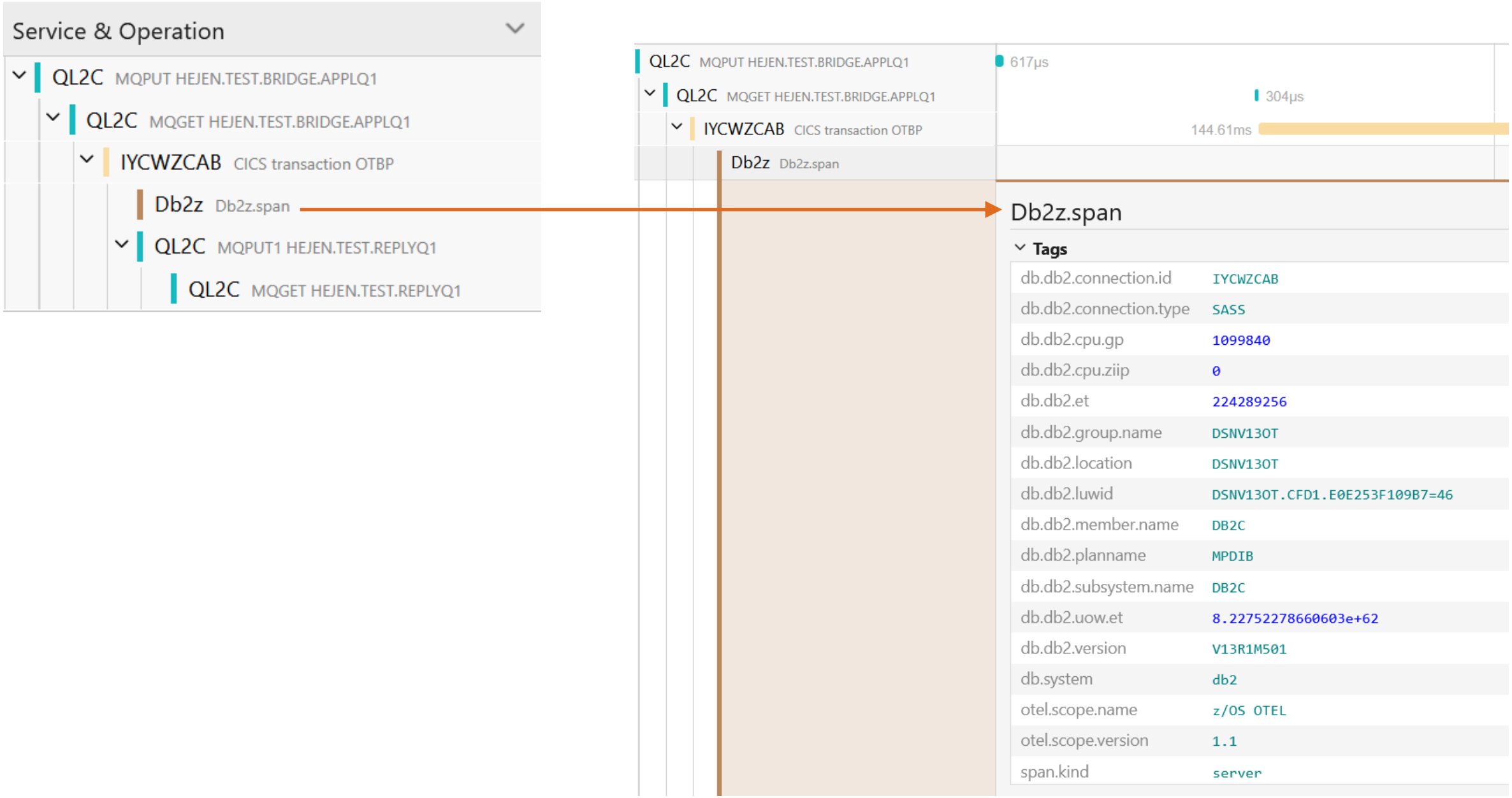




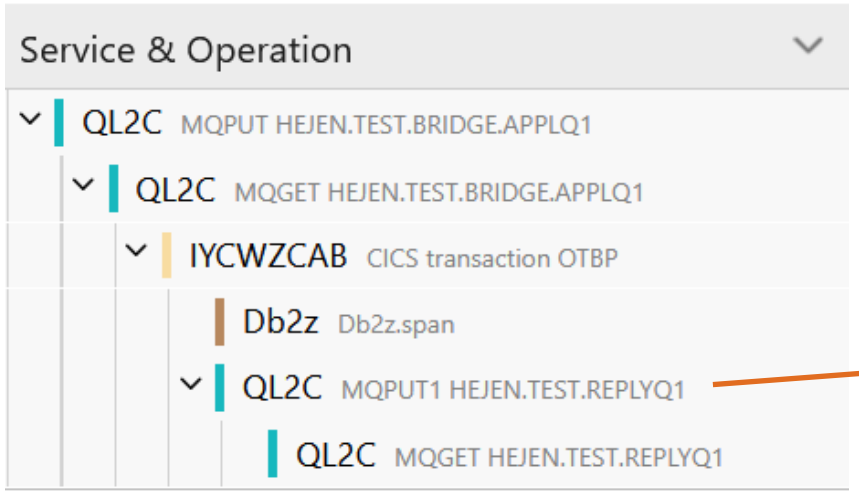
OTel Tracing – CICS, MQ and Db2 scenario

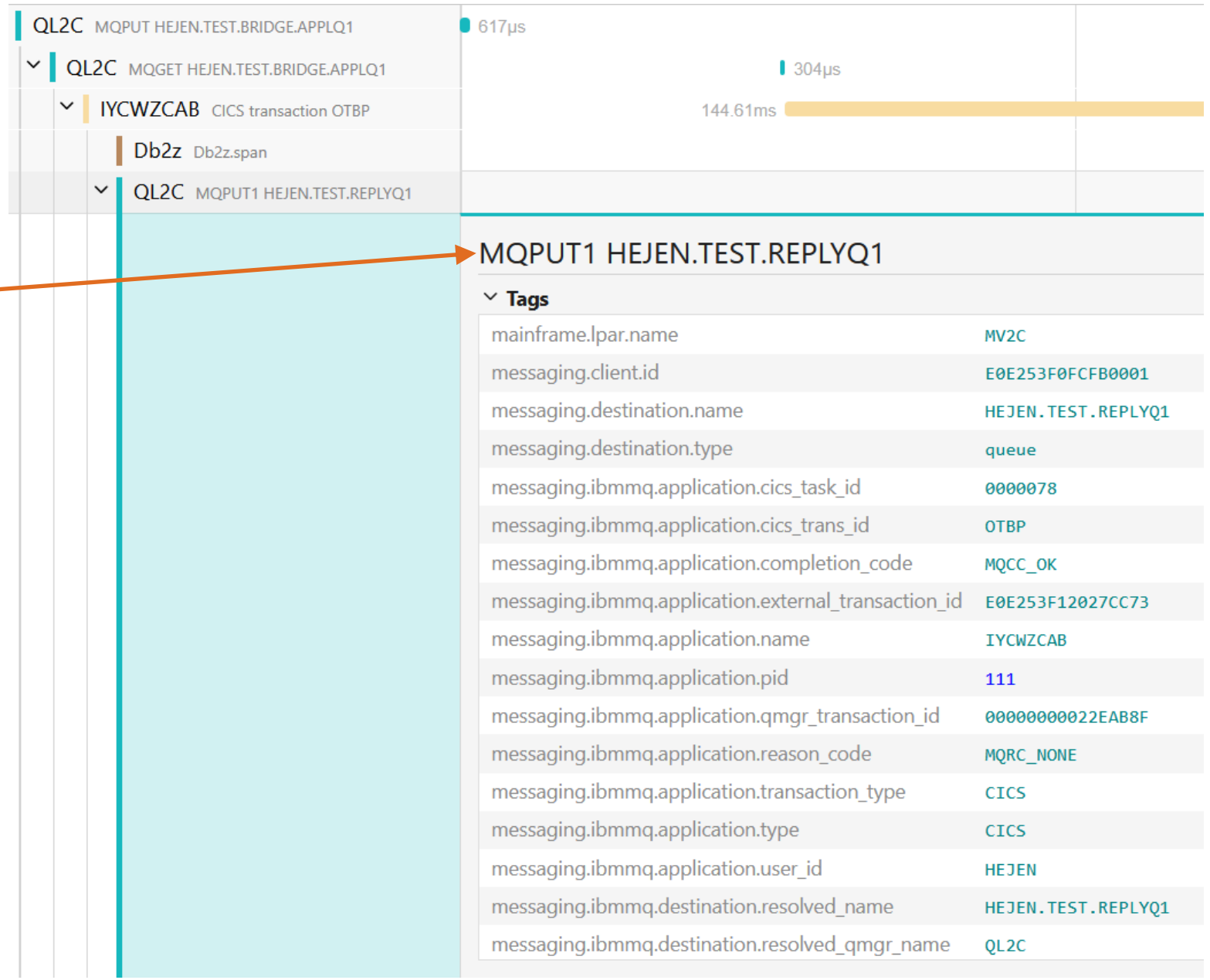


OTel Tracing – CICS, MQ and Db2 scenario

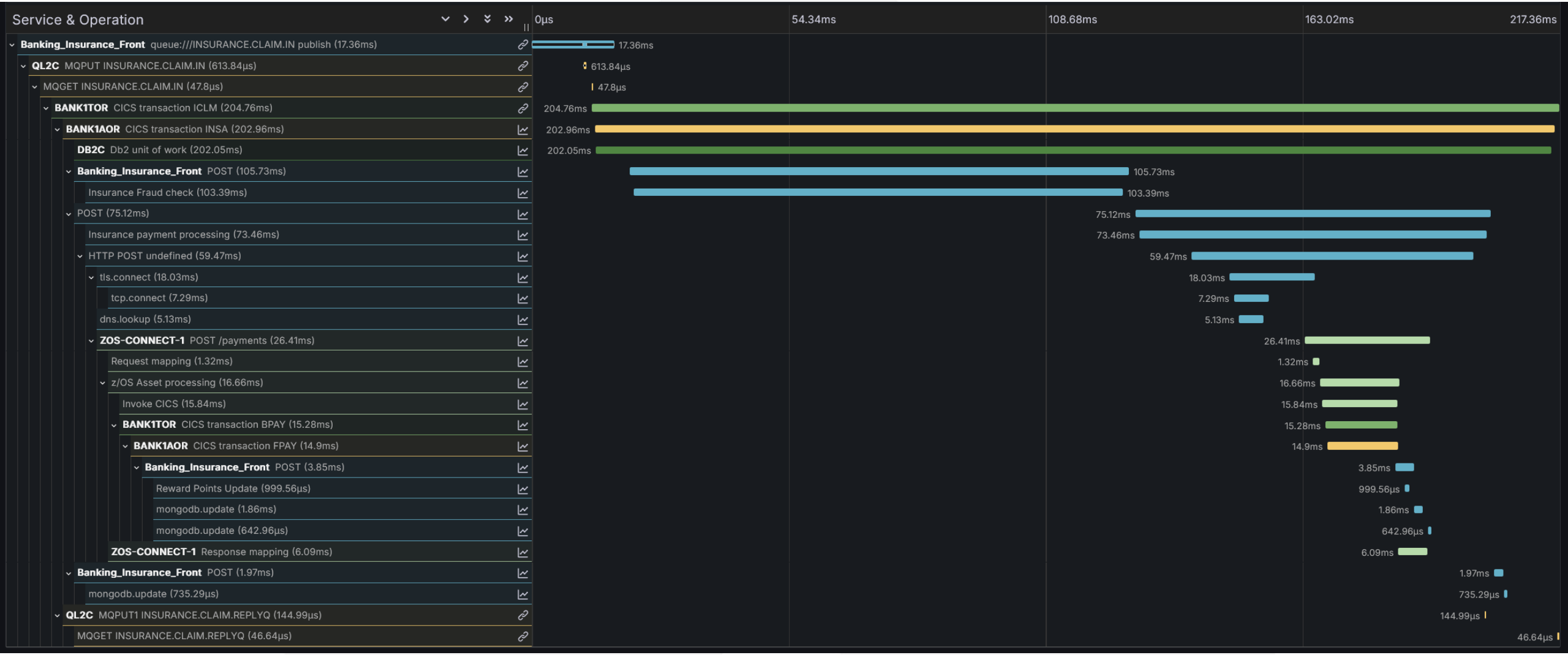


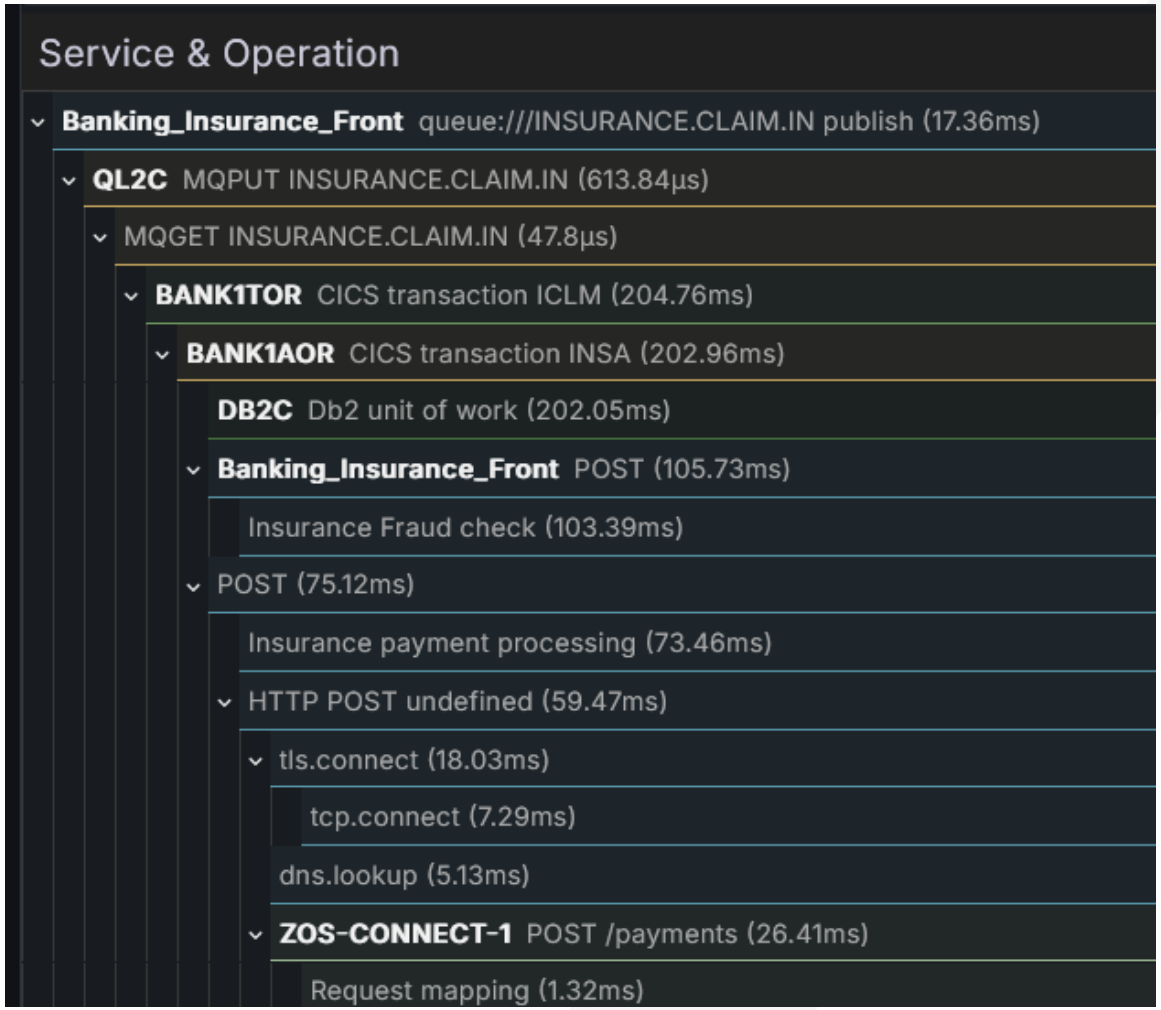
OTel Tracing – CICS, MQ and Db2 scenario

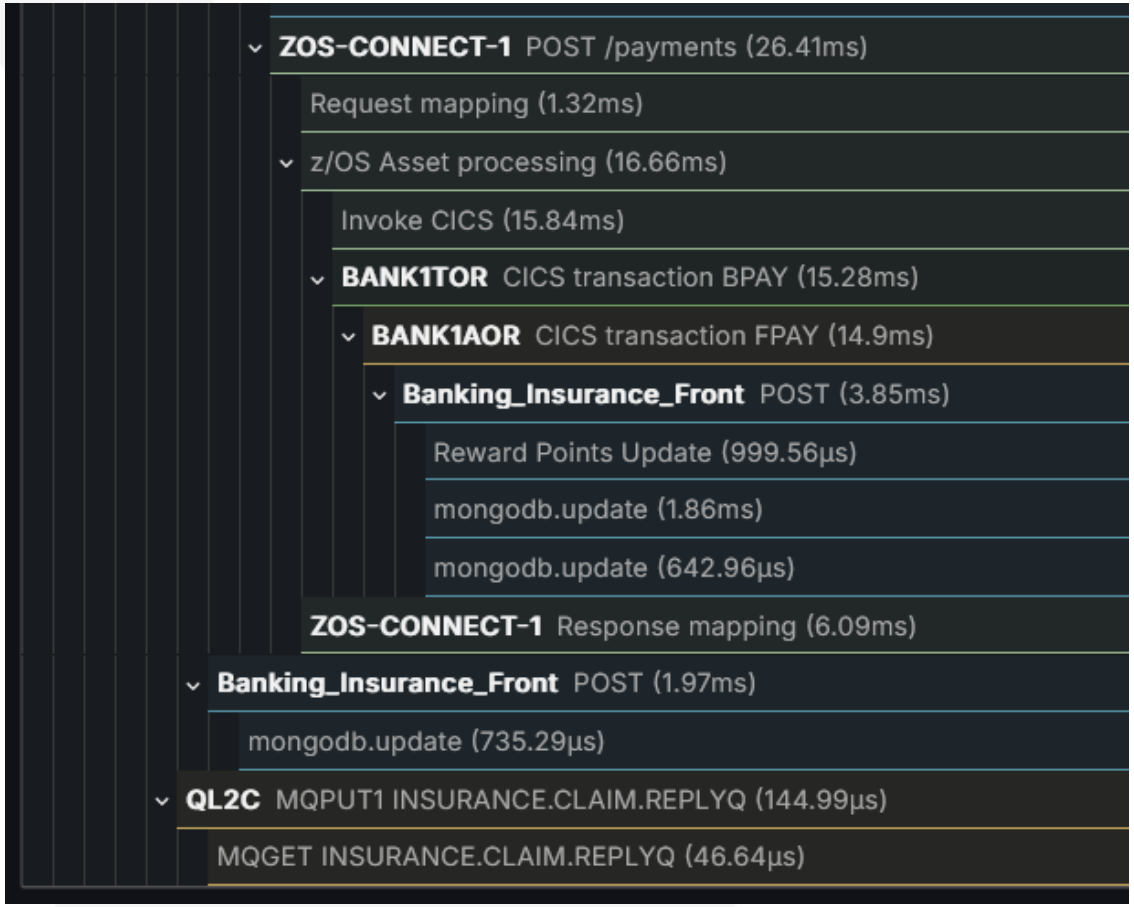




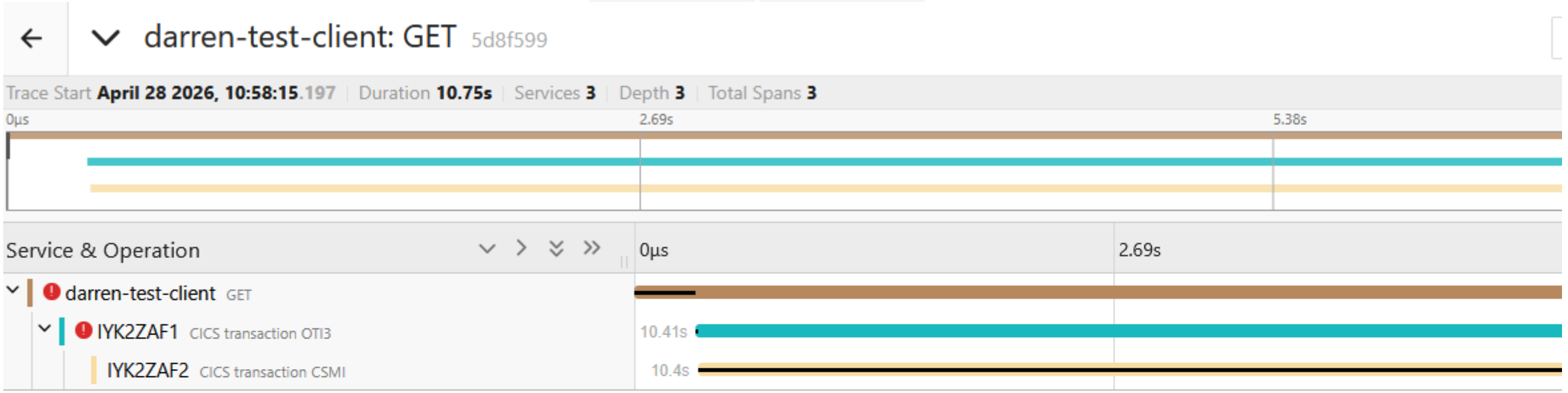
OTel Tracing as seen from Grafana



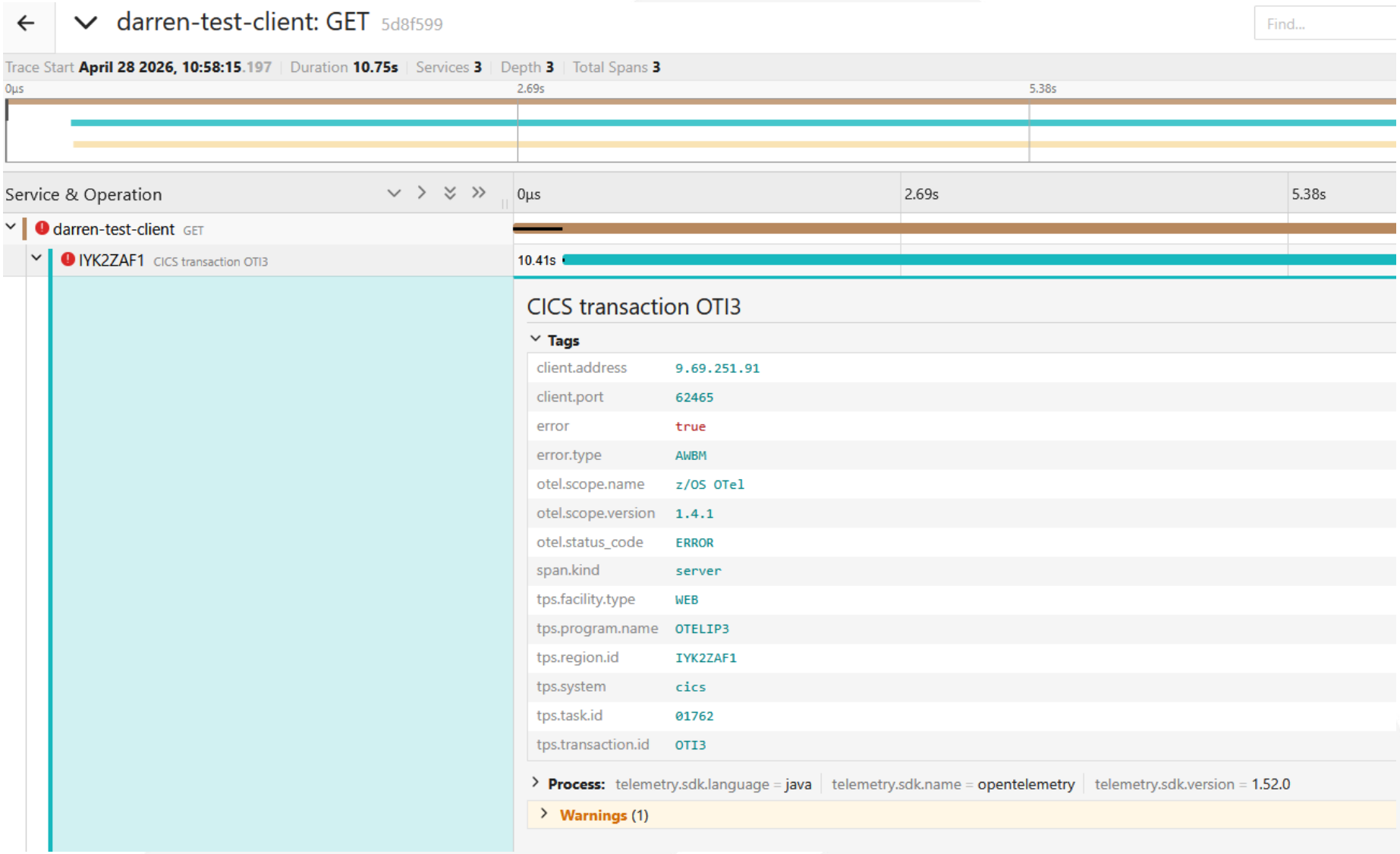




OTel Tracing showing an error in the application



OTel Tracing showing an error in the application



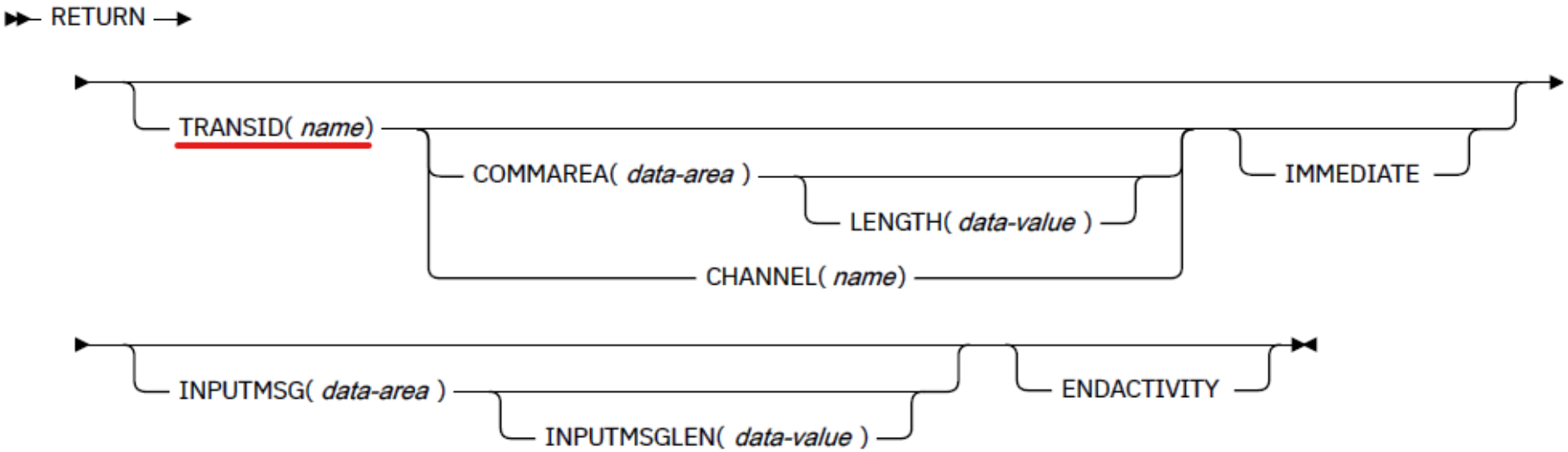
Beta enhancements

- Support for EXEC CICS RETURN TRANSID
- Support for DTP over MRO
- Long-running transactions that accept different requests with different OpenTelemetry contexts can now manage trace spans programmatically.
  - Exit programs use new XPI functions START_SPAN and END_SPAN.
  - Application programs use new API commands OTEL STARTSPAN, OTEL EXTRACT, and OTEL ENDSPAN.

EXEC CICS RETURN TRANSID

- EXEC CICS RETURN.
  - Most common usage, return control to the next higher logical level.
  - Within same task boundary, it generally has minimal impact on OTel.
  - But, this API provides many variations and options beyond this basic usage.

RETURN



EXEC CICS RETURN TRANSID

- EXEC CICS RETURN **TRANSID** ….
  - The task returns control to CICS.
  - CICS starts a **new task** – TRANSID on the terminal (of calling task)
  - Pseudo conversation between tasks – passing data from one task to the EC RETURN task(inputmsg, commarea, channel)

No behavioural changes, OTel context propagation has been added.

- From calling task to newly started EC RETURN TRANSID task
- Propagated to remote region if connection is MRO or IPIC

Distributed Transaction Processing (DTP) Optimizations

- OTel is supported for DTP over MRO
  - A check is made that both CICS regions contain the DTP support before the OTel data are sent
  - When the OTel trace context is sent, it is only included on the first communication between the connected CICS regions
  - Subsequent communications are unchanged to minimize performance impact

New OTEL APIs/XPIs

For some long running tasks or CSKL listeners, we may want to propagate OTel context into CICS task and trace them.

- Long running tasks may need to change OTel context, potentially multiple times during the task lifetime
- The rest of propagation will happen seamlessly using the existing CICS OTel propagation.
- APIs and XPIs are standard ways of providing functions to user programs.

APIs:

- **STARTSPAN:** Start a new OTel context on the issuing task.
- **ENDSPAN:** Explicitly end the existing OTel context for the issuing task.

XPIs:

- **START_SPAN**: Sets the trace parent information as OTel context to the current task.
- **END_SPAN**: Explicitly terminates the current OpenTelemetry trace.

Program logic using OTEL APIs

1. Accept a request;
2. Process the request, find out what to do next.
3. * (New logic) extract OTel context from the request, start a new OTel trace on the current task

```
exec cics otel startspan
     traceid(newoteltrid) parentid(newotelpaid)
     traceflags(newflag) version(newversion)
     resp(response) resp2(respons2)
end-exec.
```

4. Start a new CICS task or link to a CICS program;
5. Send response back to client.
6. * (New logic) End current task's OTel trace

```
exec cics otel endspan
     resp(response) resp2(respons2)
end-exec.
```

7. Wait for next request.

Program logic using OTel XPIs

```
DFHUEXIT TYPE=EP,ID=XEIIN       Generate XEIIN fields, in addition to standard GLUE fields
DFHUEXIT TYPE=XPIENV            Set up exit programming environment

COPY DFHMNOTY                   MNOT Copy book for otel

* Set TRACEID1 and PRNTID1 to the current task which will then be propagated by CICS automatically
USING DFHMNOT_ARG,R8
DFHMNOTX CALL,                                          x
               CLEAR,                                   x
               IN,                                      x
               FUNCTION(START_SPAN),                    x
               TRACEID(TRACEID1),                       x
               PARENTID(PRNTID1),                       x
               VERSION(VER1),                           x
               FLAG(FLAG1),                             x
               OUT,                                     x
               RESPONSE(*),                             x
               REASON(*)
         CLI   MNOT_RESPONSE,MNOT_OK
         BNE   CHECK002
PRT002OK WTO '----XPCREQ02 SET_TASKOTEL RESPONSE OK!----'
         B  PGMNEXT
```

```
DFHUEXIT TYPE=EP,ID=XEIIN       Generate XEIIN fields, in addition to standard GLUE fields
DFHUEXIT TYPE=XPIENV            Set up exit programming environment

COPY DFHMNOTY                   MNOT Copy book for otel

USING DFHMNOT_ARG,R8
DFHMNOTX CALL,                                          x
               CLEAR,                                   x
               IN,                                      x
               FUNCTION(END_SPAN),                      x
               OUT,                                     x
               RESPONSE(*),                             x
               REASON(*)
```

An Important Point

- It is acceptable for OTel data to be lost in some circumstances.
  - The SMF buffer could wrap leading to data overwrite
  - The SMF address space could fail completely. The real time interface means that all the data are in memory, so are lost if the address space fails.

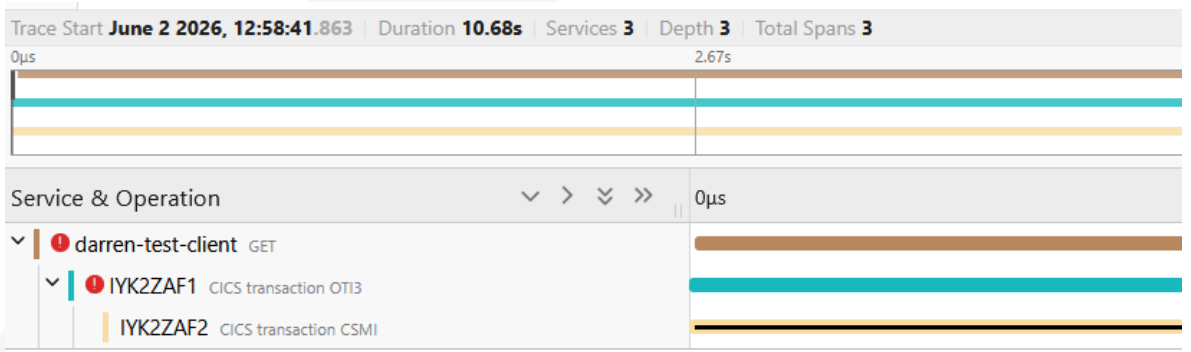
- OTel data are not business critical. There is no recovery process for lost data.

Other Important Points

- The work in CICS (CPU required) to generate spans is not large
  - Fractional change in application performance will depend on the complexity of the application
- Writing to SMF is similar to what is already done for CICS monitoring records
  - Span data records are buffered for better performance

Summary

- Open telemetry is useful in scenarios where you have distributed applications
  - It helps quickly to identify where a problem is occurring
    - An SRE or non-z/OS expert can easily extract valuable data from an OTel trace
  - Enables the right product experts to be assigned to find a resolution of the problem
  - It is an open standard
  - No application changes required



More information

- CICS TS 6.3 documentation
  - https://www.ibm.com/docs/en/cics-ts/6.x?topic=fundamentals-cics-opentelemetry
- Open Telemetry specifications
  https://opentelemetry.io/docs/specs/otel/
- IBM statement of direction concerning Open Telemetry (Published 3rd June 2025)
  https://www.ibm.com/docs/en/announcements/statement-direction-zos-middleware-products-support-opentelemetry
- MQ for z/OS 9.4.3 documentation
  https://www.ibm.com/docs/en/ibm-mq/9.4.x?topic=tracing-opentelemetry-mq-zos
- Db2 v13 details
  https://www.ibm.com/docs/en/db2-for-zos/13.0.0?topic=recovery-opentelemetry-tracing-in-db2

Experience more with IBM

SHARE

SHARE26 August 16-20, 2026 PITTSBURGH, PA

Visit us at the IBM Booth #113
This year at SHARE Pittsburgh, IBM will showcase the latest innovations in modernization, hybrid cloud, security, digital assets, and AI for IBM Z.

Connect with IBM experts, discover real-world client success stories, and experience firsthand how the latest IBM Z technologies are helping organizations accelerate transformation, strengthen resilience, and unlock new business value.

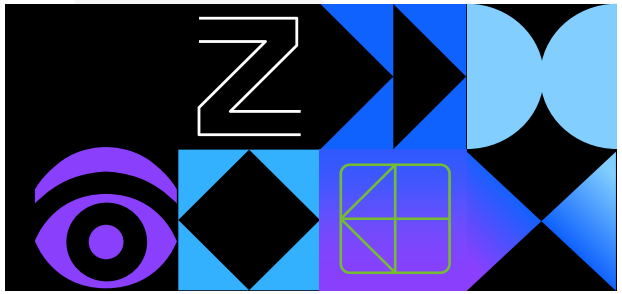
Don't miss this opportunity to learn, network, and explore how IBM Z continues to power the world's most critical applications and workloads.

Lunch and Learn sessions
Join IBM for our Lunch & Learn sessions focused on AI on IBM Z and security. These sessions are available exclusively to IBM customers and partners and are offered on a first-come, first-served basis. Seating and meals are limited.

The zSecure Transformation
17 August, 12:00–1:00 PM
Room 304/305
David L. Lawrence Convention Center

Changing the game for mainframe developers with IBM Bob
20 August, 12:00–1:00 PM
Room 304/305
David L. Lawrence Convention Center

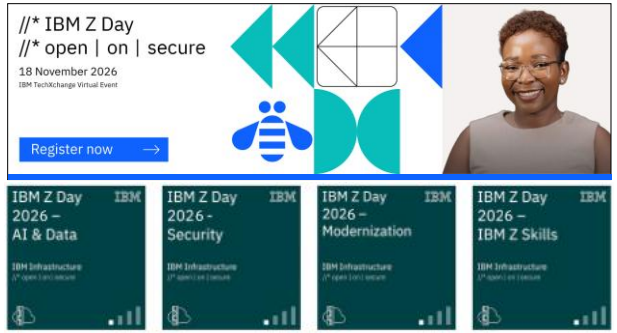
IBM TechXchange 2026
October 26-29, Atlanta



Build with purpose.
The hands-on conference where technical practitioners build real-world skills across AI, data, IT Ops, App dev, hybrid cloud & security. Experience 100+ IBM Z tech breakouts, tech talks, demos, labs, AMAs & more.

Register now →

IBM Z Day 2026
November 18



Go deeper at our largest tech event of the year, IBM Z Day - a free virtual conference featuring 6 tracks, 250+ expert speakers and participants from over 165 countries. **Earn up to 4 exclusive IBM digital skills badges**.

Register here →

Thank you for your attention

Any questions?