Tracing in Theia

Hervé KABAMBA   Michel Dagenais

May 15, 2019

Polytechnique Montréal

Laboratoire **DORSAL**

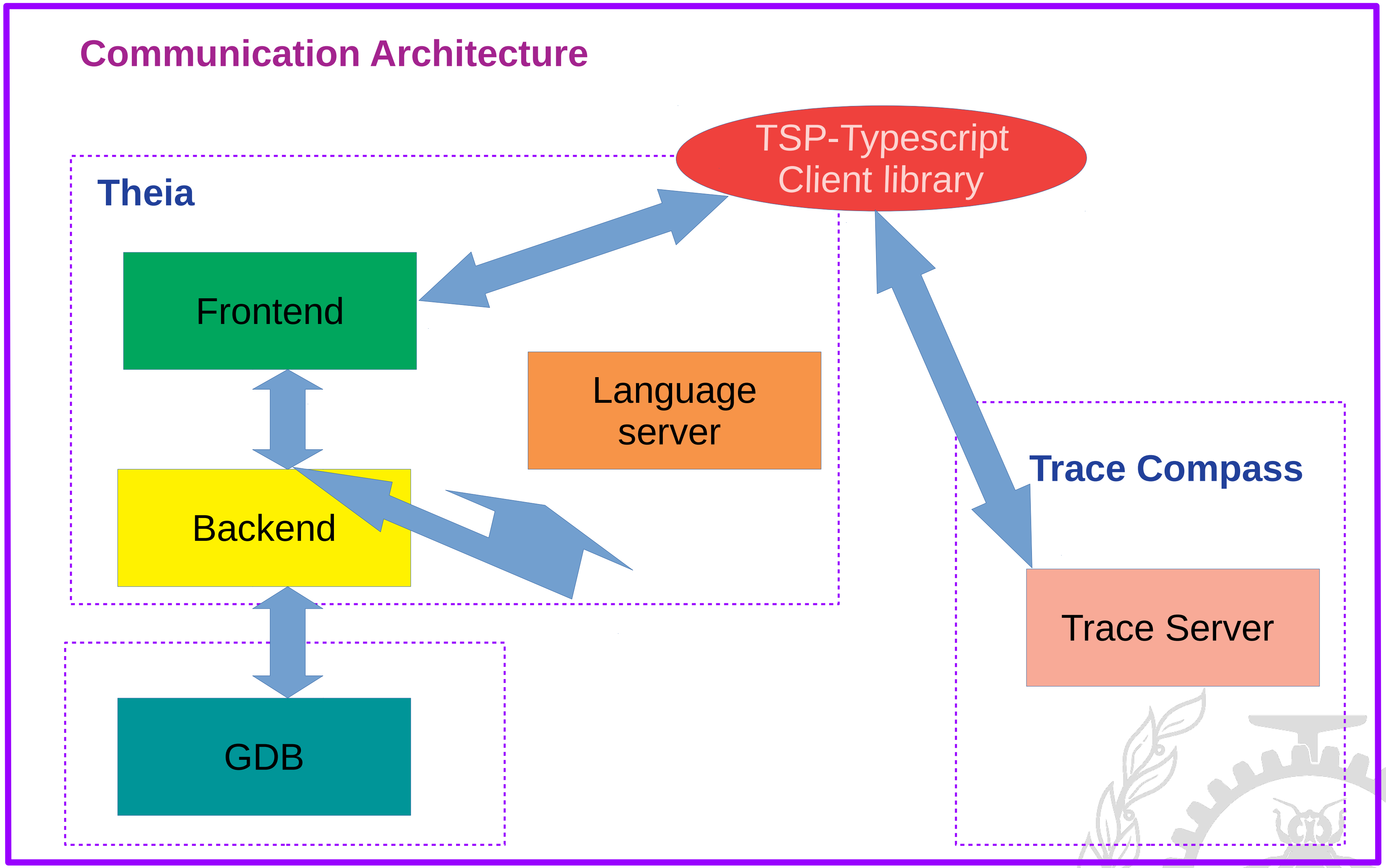
Agenda

- Introduction
- Tracing in Theia: How?
- Tracing Collection
- Ongoing work

Introduction (1)

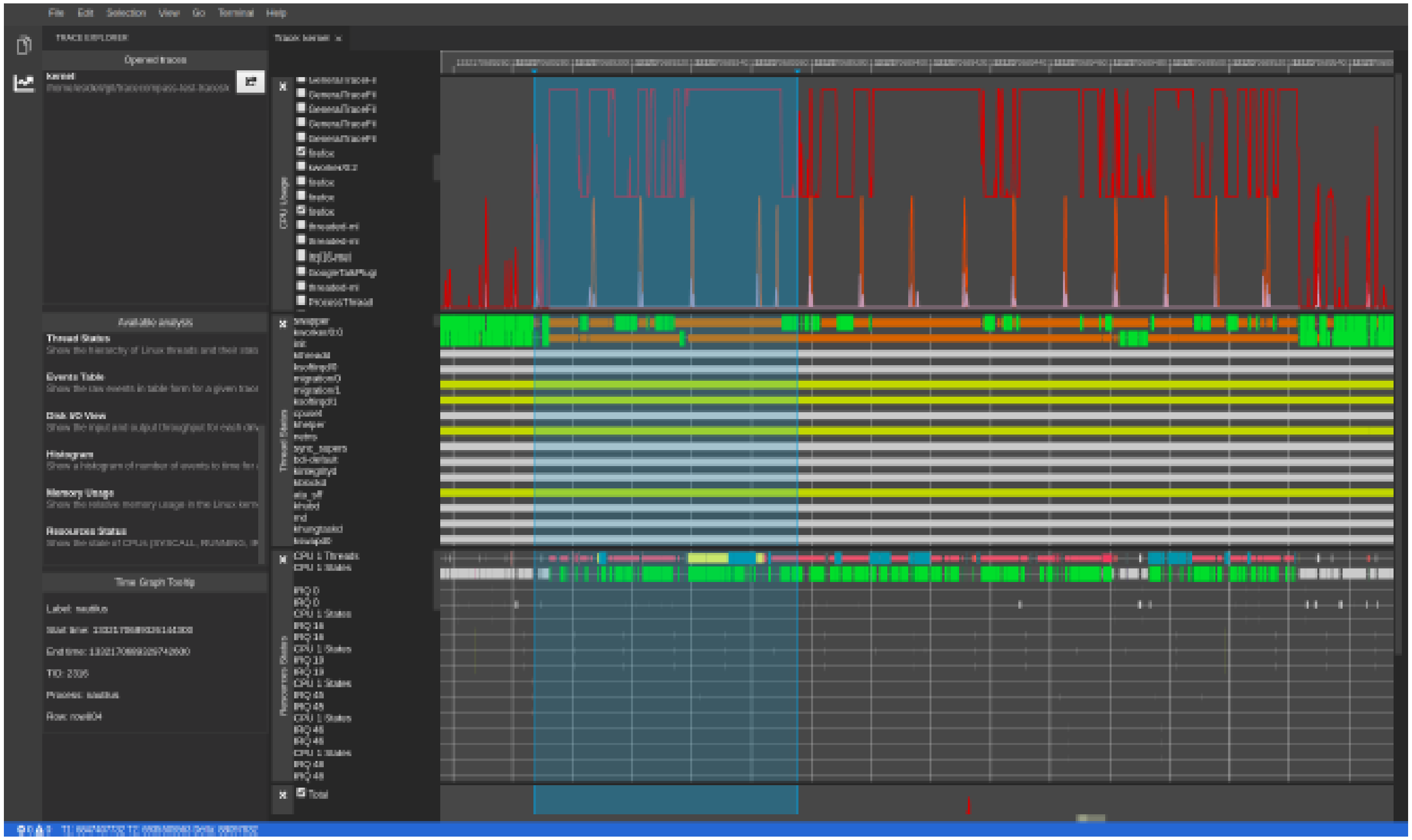
- **Theia Trace Compass Extension**
  - **Web Based trace Viewer**
  - **Complex Distributed application :**
    - **Frontend :** Run by the browser in JavaScript
    - **Backend :** communicates with the Frontend and other components
    - **Language Server Protocol :** Communicates with the Backend
    - **Trace Server:** Communicates with the Frontend
    - **GDB:** Communicates with the Backend

Introduction (2)



Introduction (3)

- **Viewing traces in Theia**



Label: nautilus

Start time: 1332170689328144300

End time: 1332170689328742600

TID: 2336

Process: nautilus

Row: row804

Introduction (4)

- **Problem addressed**

  – Theia is a complex modular and distributed application

  – Frontend is run by the browser in JavaScript

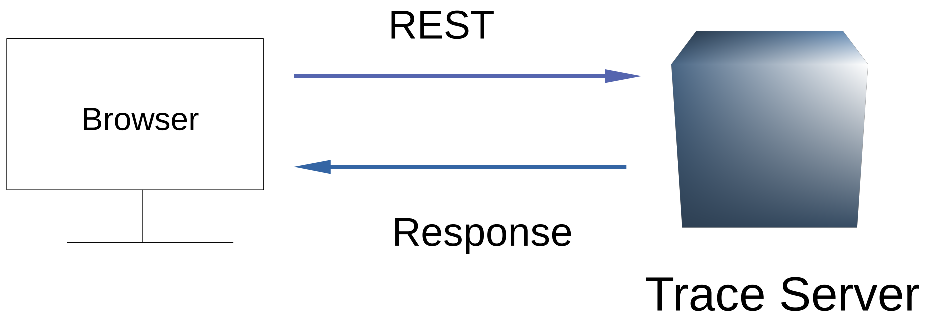
  – This modular architecture has many communicating components, we want information about the performance and interactions of each.

- **Question**

  - How to track communication in such a context, and assign performance information on the different interactions?

  - Tracing Theia distributed system should lead to a global understanding, of the performance in the whole system

Tracing in Theia: How?



1. User entry point

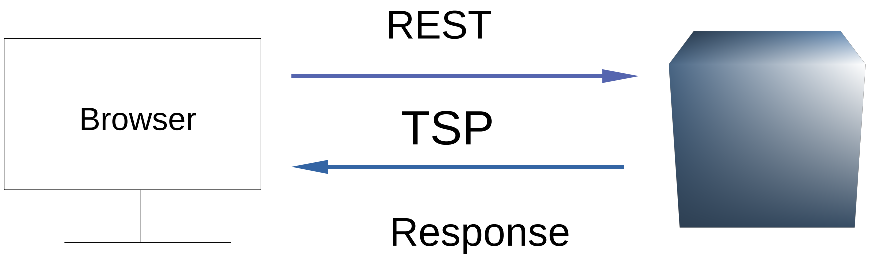
- REST calls are made from the browser to the Trace Server

-Payload encoded in Json

- The Trace Server processes the request and sends the reply

-And the process continues with more requests...

Tracing in Theia: How?



1. User entry point

All calls from the browser are managed by the TSP-Typescript Client

INSTRUMENTATION

- ZIPKIN trace points are inserted in the TSP typescript library

-All calls to the Trace Server are handled by the node Fetch API

- Wrap the Node-fetch module and add Zipkin headers

Tracing in Theia: How?

2. Frontend-Backend interactions

- Data encoded in Json

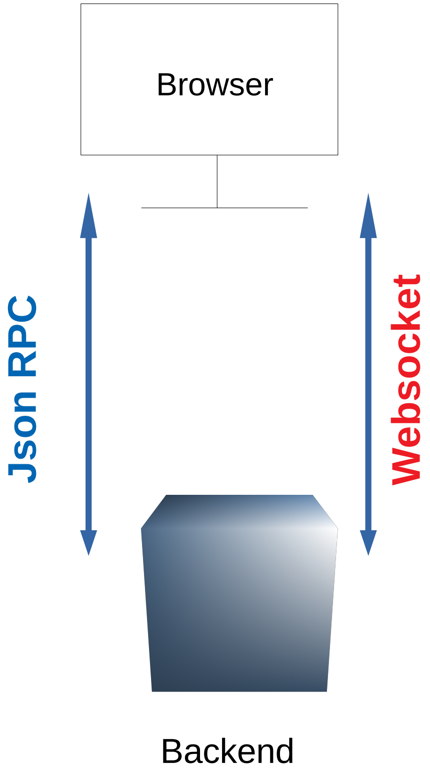
- Websockets are used for communication

- Communication is bi-directional and asynchronous

- Payload carries the remote service methods

- On the backend, data is read through the channel and the remote service is invoked

-Then the Frontend is notified of the response



Greater tracing complexity here

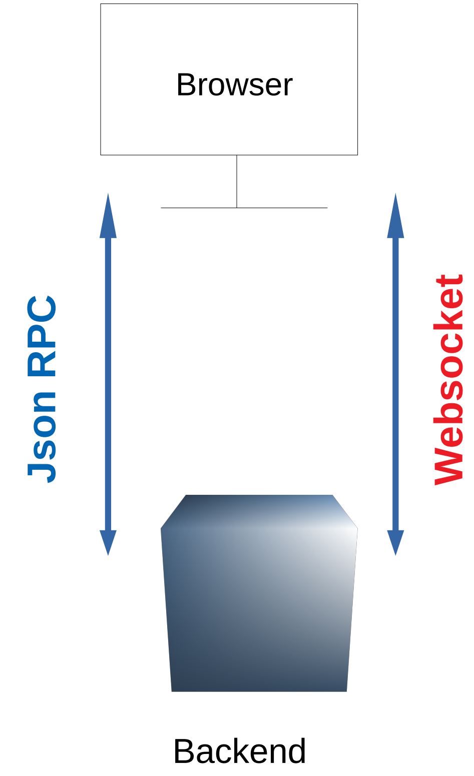
Tracing in Theia: How?

3. Targeted interaction Processes Objects (WS-Provider)

- Websocket provider service is called

- It provides all objects used to establish connection with the backend

- Communication is managed and channels are created to send and receive data.



Instrumentation

Goal: track the whole Json RPC span troughout the system

Approach:

-Place Place trace points here, when channels are created, a zipkin “cs” (Client send) annotation is added to the payload

-Add the unique “id” field from the encoded json data for span identification

- Send the data through the created channel

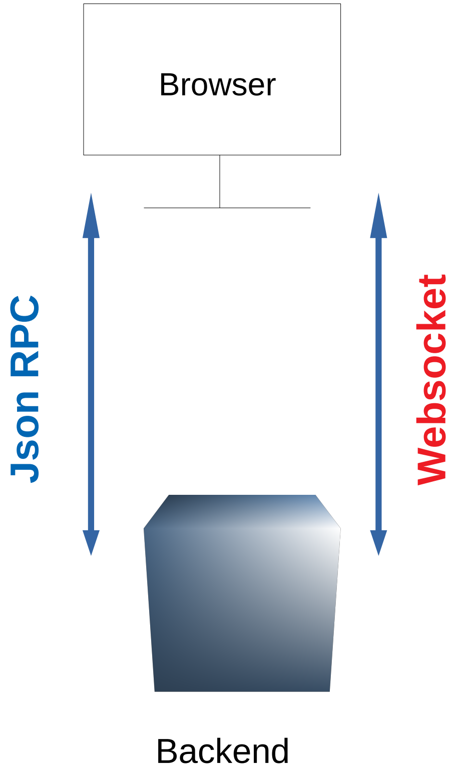
Tracing in

Theia: How?

3. Targeted interaction Processes Objects (Json RPC Proxy Factory)

- Exposes the programmatic interface of Objects through Json-RPC requests

- Allows remote calls of methods on a bi-directional channel



Instrumentation

Goal: track the whole Json RPC span throughout the system

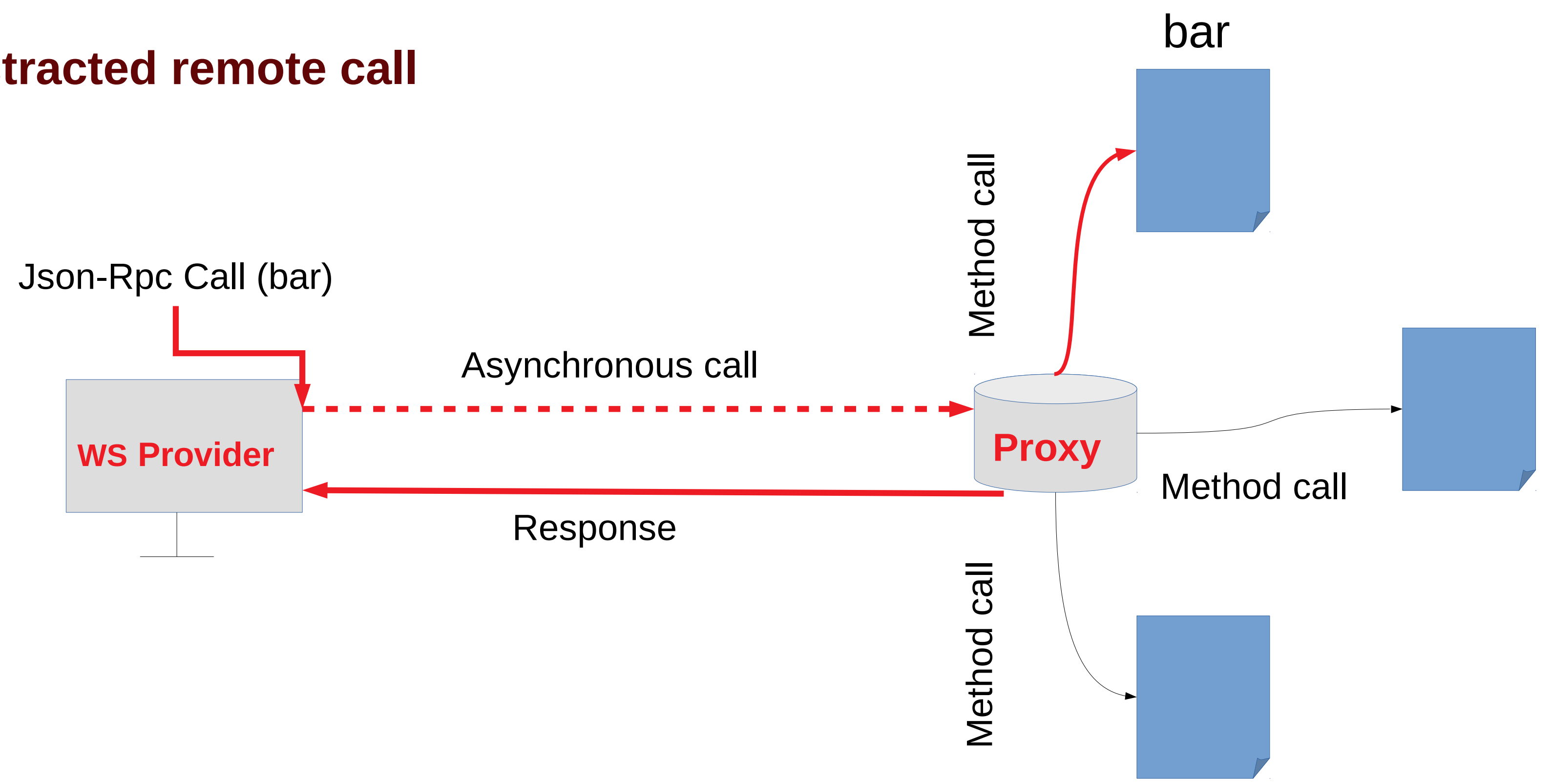
Approach:
-Decode the Json data, get the unique ID value and annotate the zipkin protocol with “sr” Server received

-Trace the call of the methods and get the response, annotate the zipkin protocol with “ss” Server sent back to the client

- On the client, get the response and look for the value of the ID in the Json data. Annotate the Zipkin protocol with “cs” client received

Tracing in Theia: How?

4. Abstracted remote call



The Json-RPC span life cycle can be seen in **red** from the client request to the backend reply

Traces Collector

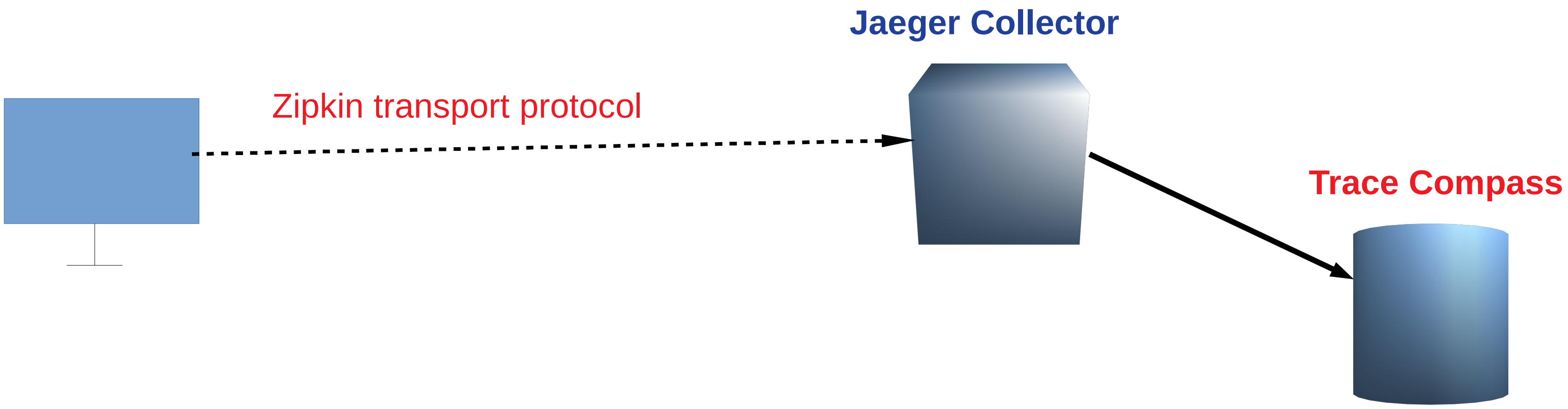
Zipkin collector :

Not supported by trace compass

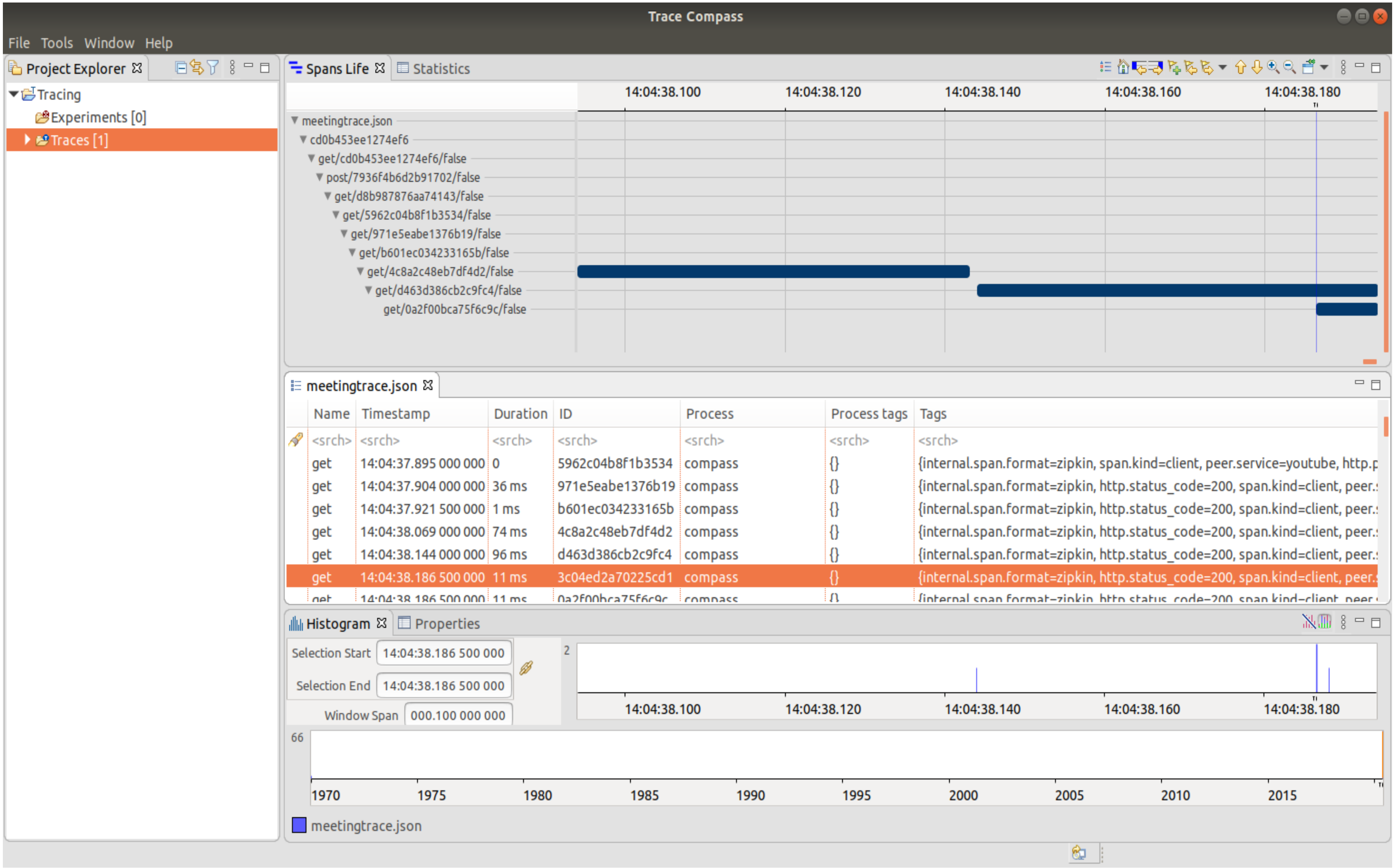
Solution:

We use the **Jaeger collector** for our Zipkin traces. Traces are sent automatically to the Jaeger collector.

The Jaeger trace encoding is supported by trace compass



In Trace compass:



Name	Timestamp	Duration	ID	Process	Process tags	Tags
<srch>	<srch>	<srch>	<srch>	<srch>	<srch>	<srch>
get	14:04:37.895 000 000	0	5962c04b8f1b3534	compass	{}	{internal.span.format=zipkin, span.kind=client, peer.service=youtube, http.p
get	14:04:37.904 000 000	36 ms	971e5eabe1376b19	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.
get	14:04:37.921 500 000	1 ms	b601ec034233165b	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.
get	14:04:38.069 000 000	74 ms	4c8a2c48eb7df4d2	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.
get	14:04:38.144 000 000	96 ms	d463d386cb2c9fc4	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.
get	14:04:38.186 500 000	11 ms	3c04ed2a70225cd1	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.
get	14:04:38.186 500 000	11 ms	0a2f00bca75f6c9c	compass	{}	{internal.span.format=zipkin, http.status_code=200, span.kind=client, peer.

Selection Start 14:04:38.186 500 000

Selection End 14:04:38.186 500 000

Window Span 000.100 000 000

Ongoing Work

- **Trace collection and analysis**

  Use cases:

  - **Zooming in Theia, exposing root causes**
  - **Websockets are sequential, can they sometimes contribute to any Theia performance issue?**

  Next

  - **We keep our eyes on GDB and language Server, if needed we should get their performance information.**

Questions?

herve.kabamba-mbikayi@polymtl.ca