Trace Coordinator

Ahmad Faour

Polytechnique Montréal

DORSAL Laboratory

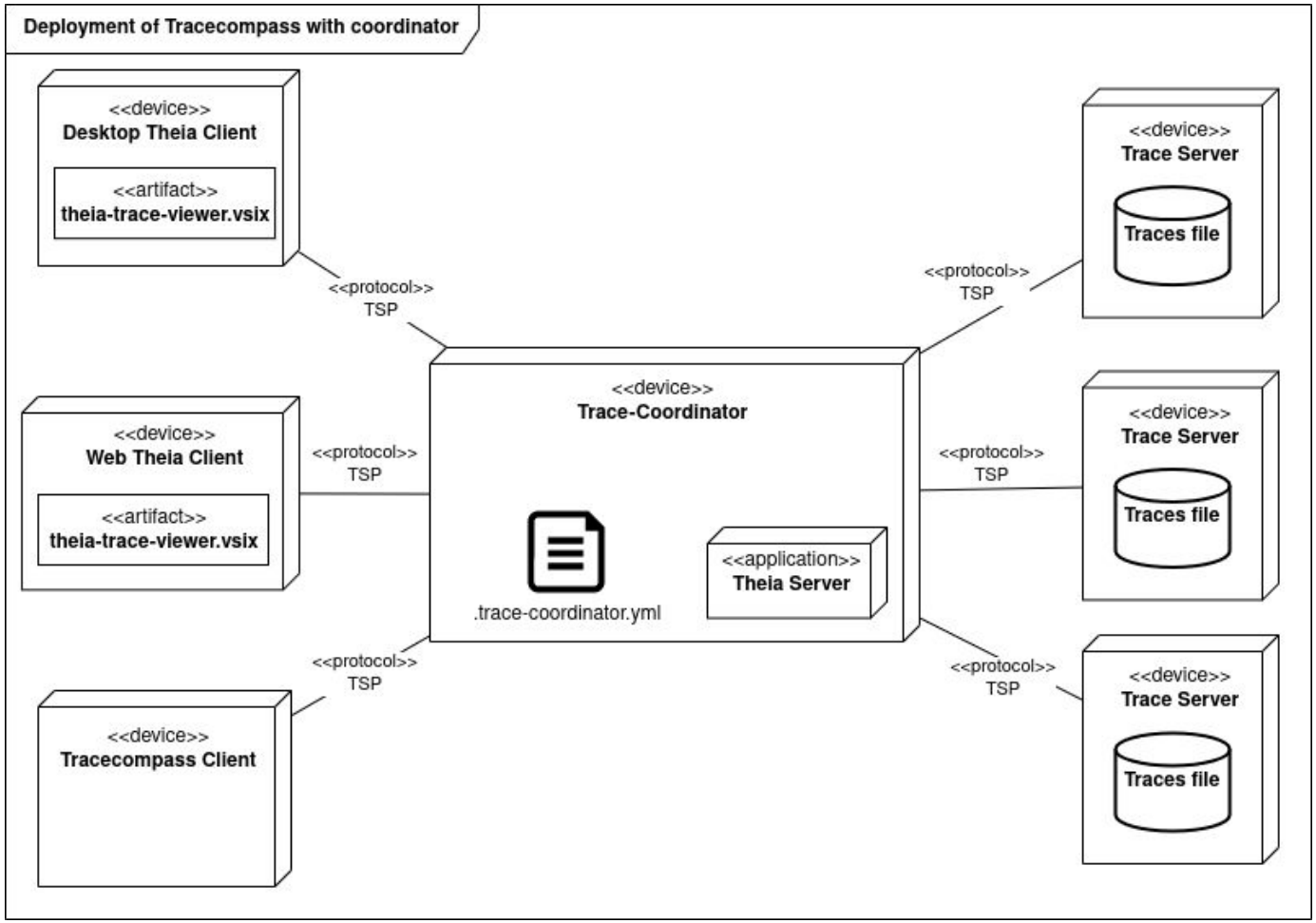
Agenda

- Introduction
- Trace Coordinator
- Benchmark
- TSP Extension
- What remains to be done?
- Conclusion

Introduction - Use Cases

- Target use cases
  - High Performance Computing with MPI Cluster
  - Microservice with Kubernetes Cluster
- Other interesting use cases
  - LTTng Log Rotation
  - Similarity of queries
  - Client-Server

Reminder - Trace Coordinator



- Different Clients
- Trace Server Protocol (TSP)
- Many Trace Servers
- Trace Coordinator
- Distributed trace files

Reminder - Aggregation Of Trace Data (Vertically)

- Generate new Id
  - XY (Series, Tree)
  - TimeGraph (Arrows, States, Tree)
- Each trace server is assigned a range of id (those id are transparent for the Trace Server)

$$step = \frac{INT32\_MAX}{NUMBER\_OF\_TRACE\_SERVER}$$

$$range = [step * traceServerId, step * (traceServerId + 1)[$$

Distributed Analysis - Aggregation Exemple

- TraceCompass Server #1: [0, 1000[
- TraceCompass Server #2: [1000, 2000[


```
{
    "statusMessage": "Completed",
    "model": {
        "headers": [],
        "entries": [
            {
                "id": 0,
                "parentId": -1,
                "style": null,
                "labels": [
                    "apt"
                ],
                "start": 1539814146277337159,
                "end": 1539814150494142355,
                "hasData": true
            }
        ]
    },
    "status": "COMPLETED"
}
```


Response of TraceCompass Server #1


```
{
    "statusMessage": "Running",
    "model": {
        "headers": [],
        "entries": [
            {
                "id": 1,
                "parentId": -1,
                "style": null,
                "labels": [
                    "pacman"
                ],
                "start": 1539786952342493550,
                "end": 1539786952794811137,
                "hasData": true
            }
        ]
    },
    "status": "RUNNING"
}
```


Response of TraceCompass Server #2

Distributed Analysis - Aggregation Exemple

Trace-Coordinator Change:

- Status
- headers
- Entries
- Id
- ParentId


```
{
    "statusMessage": "Running",
    "model": {
        "headers": [],
        "entries": [
            {
                "id": 0,
                "parentId": -1,
                "style": null,
                "labels": [
                    "apt"
                ],
                "start": 1539814146277337159,
                "end": 1539814150494142355,
                "hasData": true
            },
            {
                "id": 1001,
                "parentId": -1,
                "style": null,
                "labels": [
                    "pacman"
                ],
                "start": 1539786952342493550,
                "end": 1539786952794811137,
                "hasData": true
            }
        ]
    },
    "status": "RUNNING"
}
```


Response of Trace-Coordinator

Benchmark - MPI Benchmark

Soma

Soma is a High-Performance Computing (HPC) Monte-Carlo simulation for soft coarse-grained polymers

	Trace File Size (Mb)	Mean Processing Time (s)					
		Open Trace	Create Experiment	Get Output Descriptors	Get Time Graph (Thread Status/Control Flow)		
					Tree	States	Arrows
20 Workers	20 x (~100 to ~300)	0,82	37,21	0,08	2,61	64,46	25,49
Gain (%)		*236%*	*1247%*	*105%*	*623%*	*37%*	*40%*
Returned Payload (MB)				0,0124	3,7	526,3	38,4
10 Workers	20 x (~100 to ~300)	0,78	58,35	0,08	2,77	60,33	20,68
Gain (%)		*248%*	*795%*	*108%*	*587%*	*40%*	*49%*
Returned Payload (MB)				0,0124	3,7	526,3	38,2
5 Workers	20 x (~100 to ~300)	0,90	93,41	0,10	4,89	50,79	19,21
Gain (%)		*214%*	*497%*	*88%*	*333%*	*47%*	*53%*
Returned Payload (MB)				0,0124	3,6	526,3	37,8
1 Worker	20 x (~100 to ~300)	1,93	463,81	0,09	16,27	23,86	10,16
Returned Payload (MB)				0,0124	3,7	526,3	38,4

*Each Worker: Intel Core i5-4570s 2.9GHz, 16Go RAM

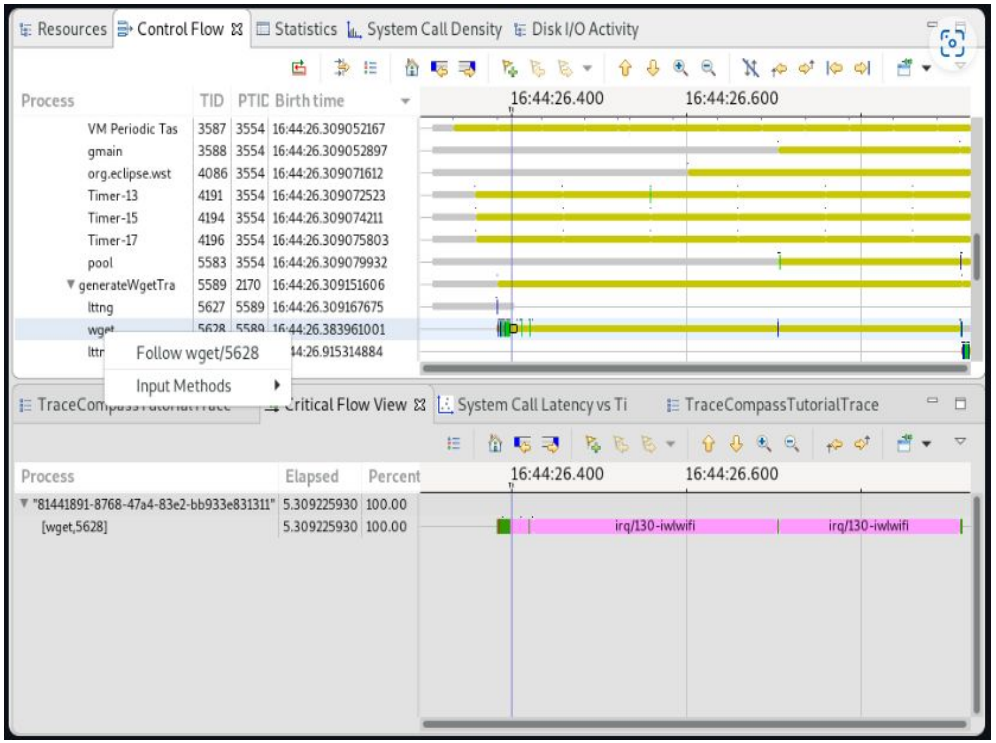
TSP Extension - Action API

Problem: Define input for each data provider for different view at server level.
Objectif: Enable feature (e.g. filter, search, critical path) for specific analysis

Verbs	Endpoints	Description
POST	/experiments/{expUUID}/outputs/timegraph/{outputId}/tooltip/actions	Get all tooltip action for an item
POST	/experiments/{expUUID}/outputs/timegraph/{outputId}/tooltip/actions/{actiondId}	Apply tooltip action

TSP Extension - Action API

- Action Tooltip Message and Input Parameters are build directly in the server
- Two type of actions: INPUT_PROVIDER, OUTPUT_LOCATION




```
{
    "model": [
        {
            "providerType": "TIME_GRAPH",
            "actionTooltipMessage": "Follow Thread 3",
            "inputParameters": {
                "hostThread": {
                    "tid": 3,
                    "host": "\"d7423748-2345-4a99-bc73-fce71c9681cf\""
                }
            },
            "targetProviderId": "org.eclipse.tracecompass.analysis.graph.core.dataprovider.
            CriticalPathDataProvider",
            "actionType": "INPUT_PROVIDER",
            "description": "Critical Path",
            "name": "Follow Thread",
            "id": "FollowThreadAction"
        }
    ],
    "statusMessage": "Completed",
    "status": "COMPLETED"
}
```

What remains to be done?

- Aggregation over time (Horizontally)
- TSP Extension - OS Execution Graph API (Critical Path*)
- Explore performance gain by integrating Streaming instead of Polling
- Reduce json Serialization/Deserialization (e.g. protobuf)

*Distributed computation of critical path (from Pierre-Frédérick Denys)

Conclusion

- Trace-Coordinator
- Serialization/Deserialization of JSON bottleneck
- Extends TSP to include more complex analyses

Github Repository:

- tsp-java-client: Client side implementation, in Java, of the Trace Server Protocol
- Trace Coordinator: Trace Coordinator Project implement in Java
- Trace Coordinator CLI: CLI for benchmark and help to generate config file

Q&A

Thank you for listening!