

Enhancing Business Process Mining with Distributed Tracing Data in a Microservice Architecture

Jochen Graeff (B.Sc.) | 21.08.2017 | Master thesis final presentation
Advisor: Martin Kleehaus

Chair of Software Engineering for Business Information Systems (sebis)
Faculty of Informatics
Technische Universität München
www.matthes.in.tum.de

Agenda

Motivation

Research questions

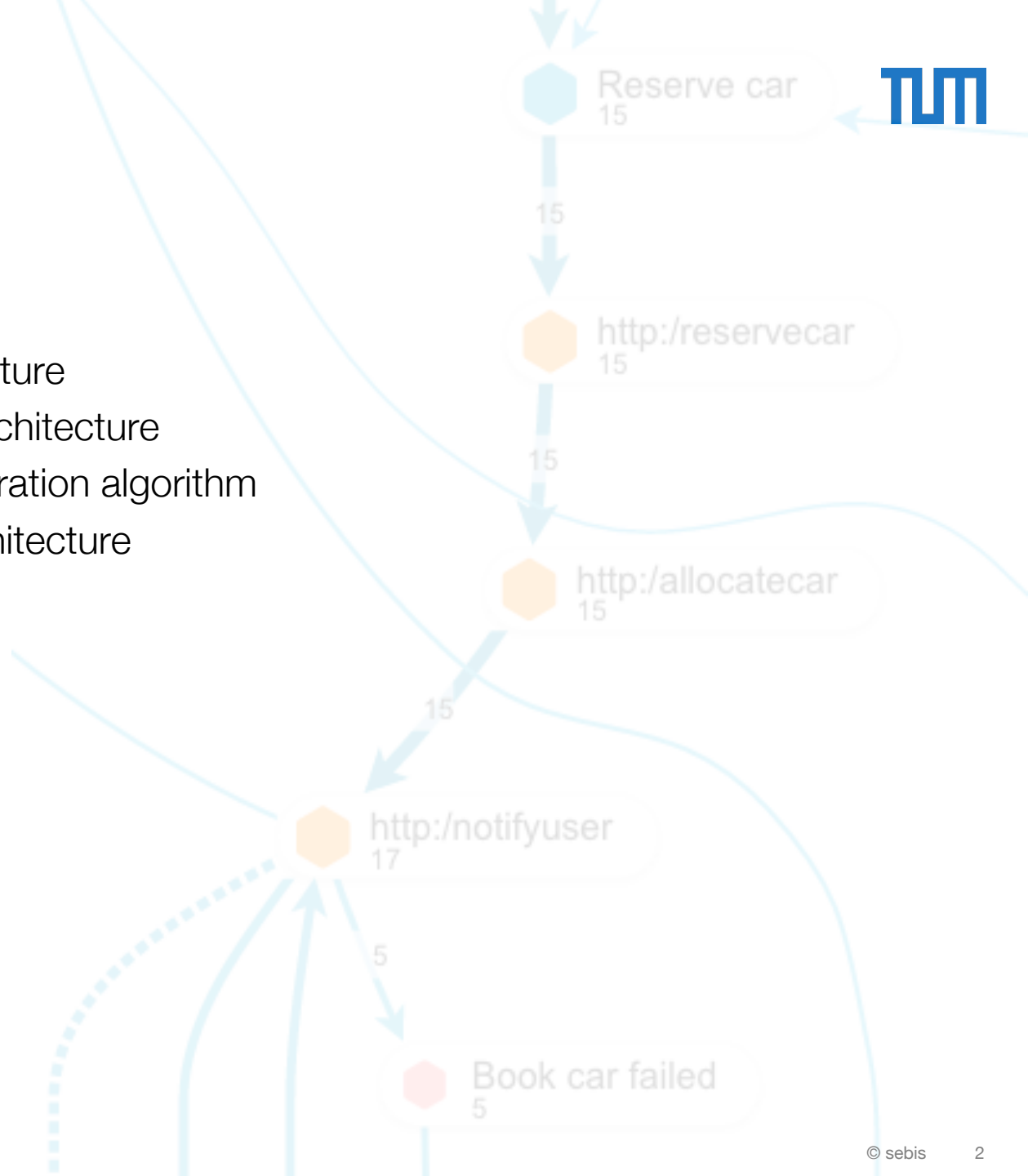
Approach

- Build sample architecture
- Instrument sample architecture
- Develop activity generation algorithm
- Set-up extended architecture
- Analysis creation

Live Demo

Evaluation

- Benefits
- Limitations



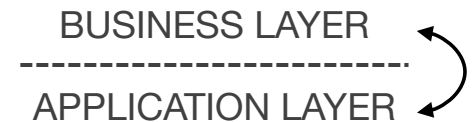
Amazon found every 100ms of latency cost them 1% in sales!¹



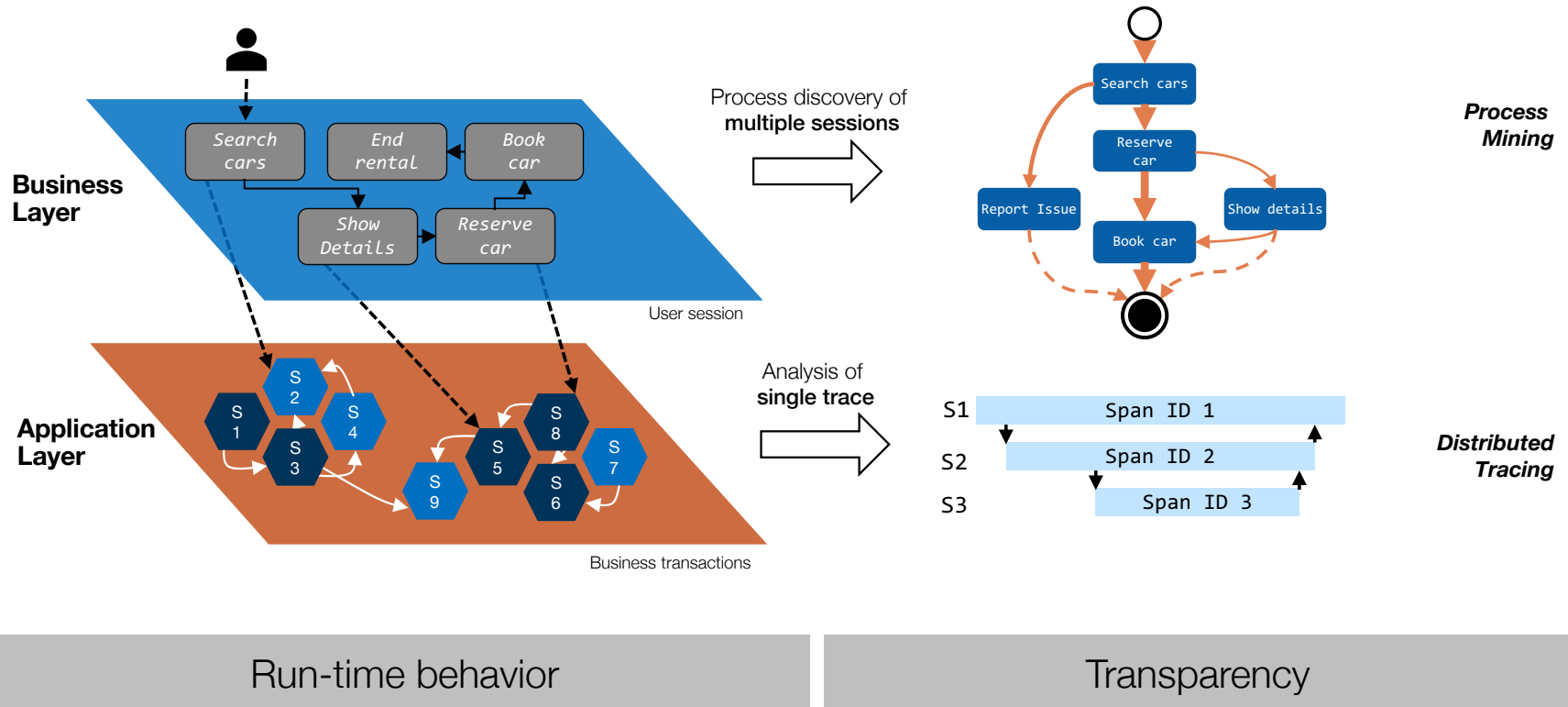
Walmart saw up to a 2% increase in conversions for every 1 second of improvement in load time. Every 100ms improvement also resulted in up to a 1% increase in revenue.²






How does **user behaviour** and **system behaviour** influence each other?



Is there a gap between the layers?



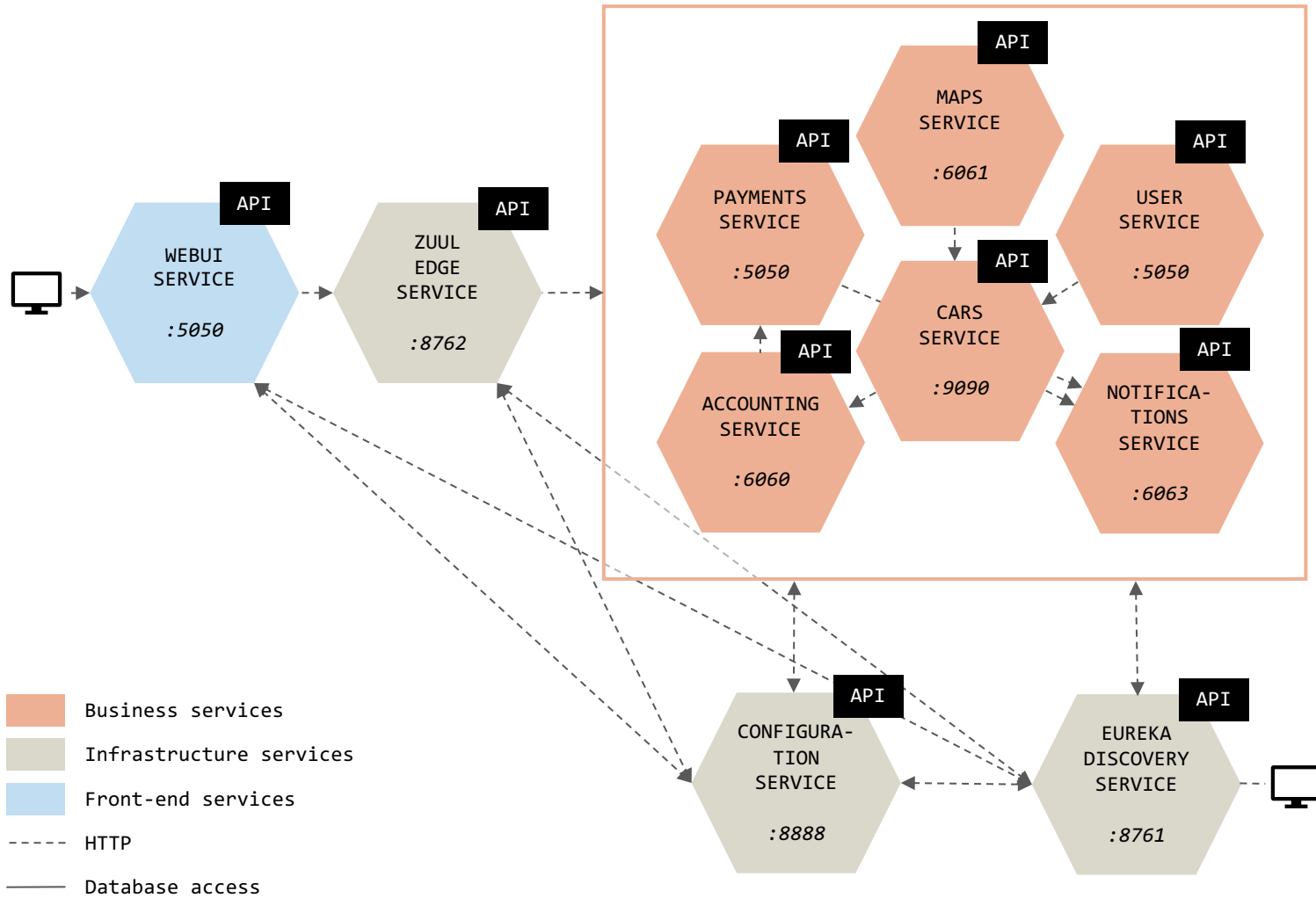
-  **RQ1.** How can a relationship between business activities and a distributed application architecture be established?
-  **RQ2.** What data has to be extracted and how has it to be mapped to enable and store the relationship knowledge?
-  **RQ3.** How can business process mining be extended with technical aspects in order to uncover
 - a) user and system throughput times for business activity executions and,
 - b) correlations between business process performance and system behaviour?

Build sample architecture

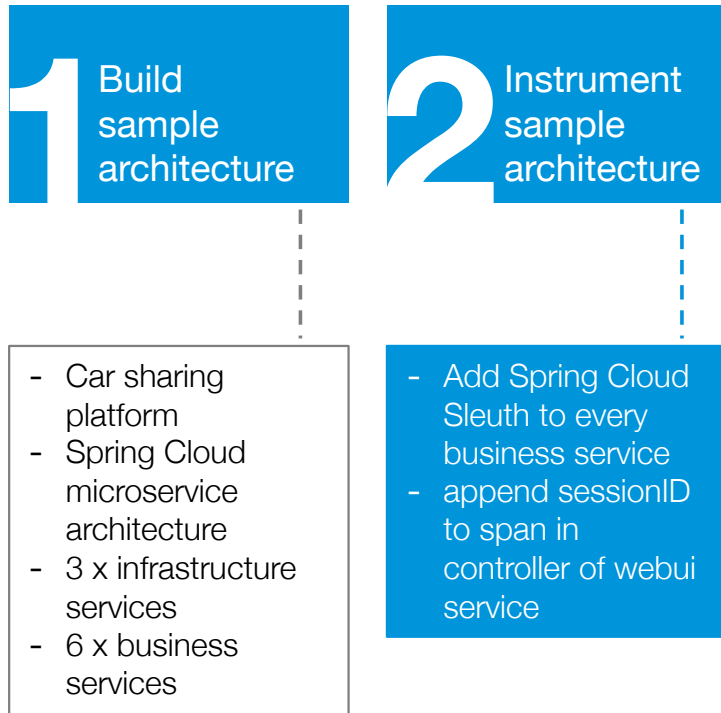
1 Build sample architecture

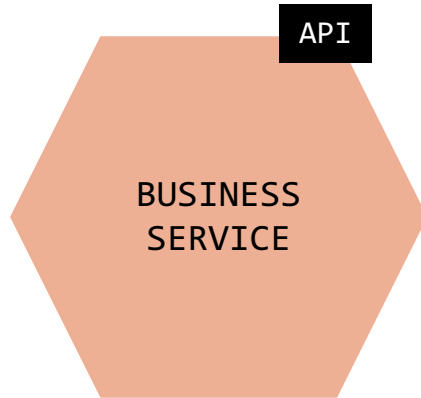
- Car sharing platform
- Spring Cloud microservice architecture
- 3 x infrastructure services
- 6 x business services

Build sample architecture

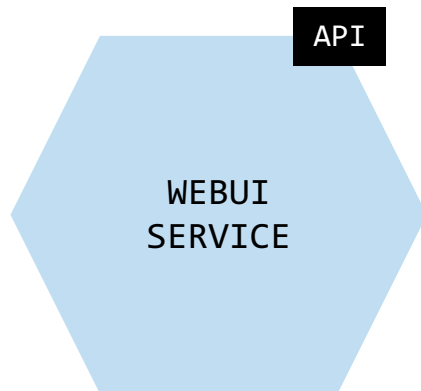


Instrument sample architecture



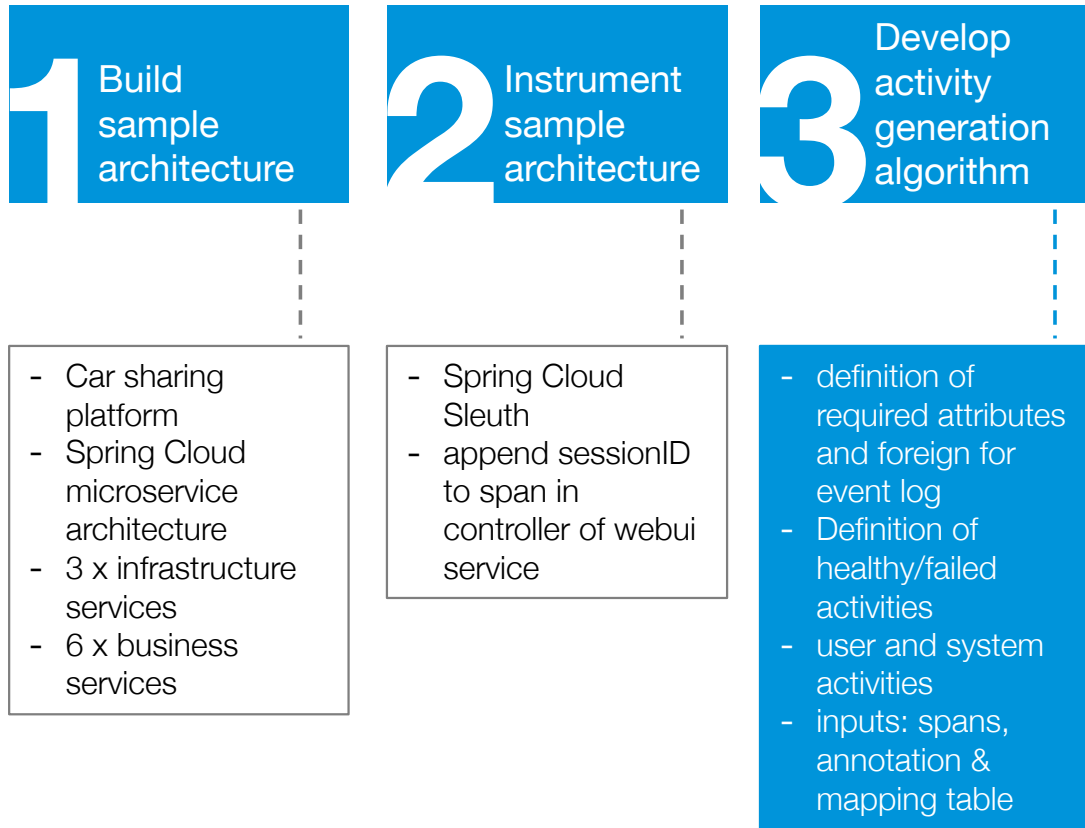


- Instrument every business service with spring cloud sleuth in order to generate span data in the applications
 - Add dependency `spring-cloud-starter-zipkin`
 - Set sampling rate



- Instrument service (as for business services)
- Append sessionID in every webui service endpoint definition: `tracer.addTag("sessionID", sessionID);`

Develop activity generation algorithm



Activity generation

spans table (zipkin)

TRACE ID	SPAN ID	PARENT ID	NAME	TIMESTAMP	DURATION
a	a	null	http://bookcar	07/06/17 12:45:32.000	800 ms
a	b	a	initialize	07/06/17 12:45:32.100	600 ms
a	c	b	http://initializebooking	07/06/17 12:45:32.200	500 ms
a	d	c	http://handlecarbooking	07/06/17 12:45:32.400	100 ms
b	b	null	http://opencar	07/06/17 12:37:32.400	3000 ms

annotations table (zipkin)

TRACE ID	SPAN ID	KEY	VALUE
a	a	http.method	GET
a	a	cs	
a	a	cr	
a	a	sessionID	12345
a	b	ss	

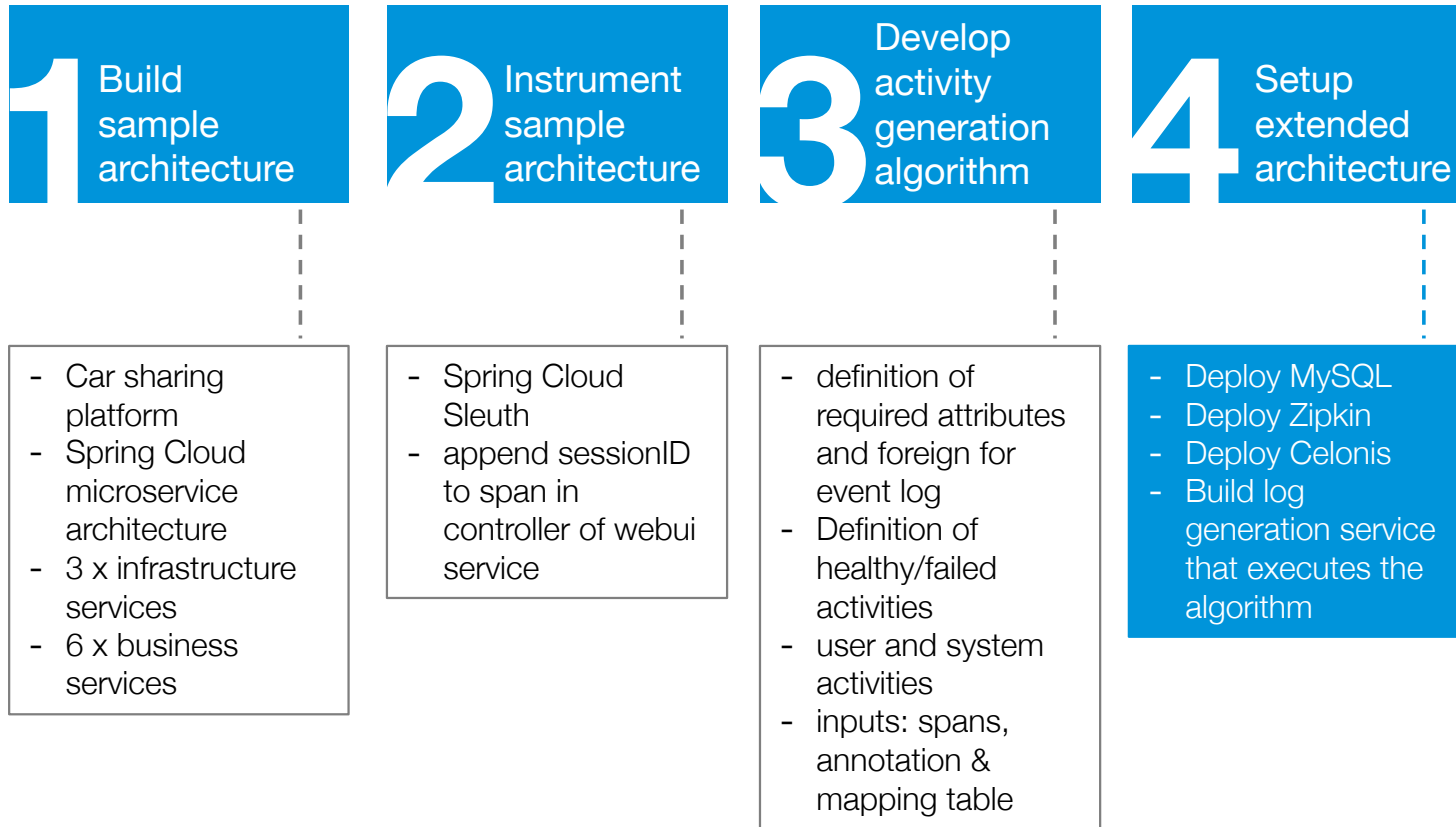
mapping table

technical_activity	pretty_name	is_activity	calls_service
http://bookcar	Book car	1	webui-service
http://initializebooking	http://initializebooking	1	accounting-service
http://handlecarbooking	http://handlecarbooking	1	cars-service
http://opencar	Unlock car	1	webui-service
http://findroute	Find route	1	webui-service

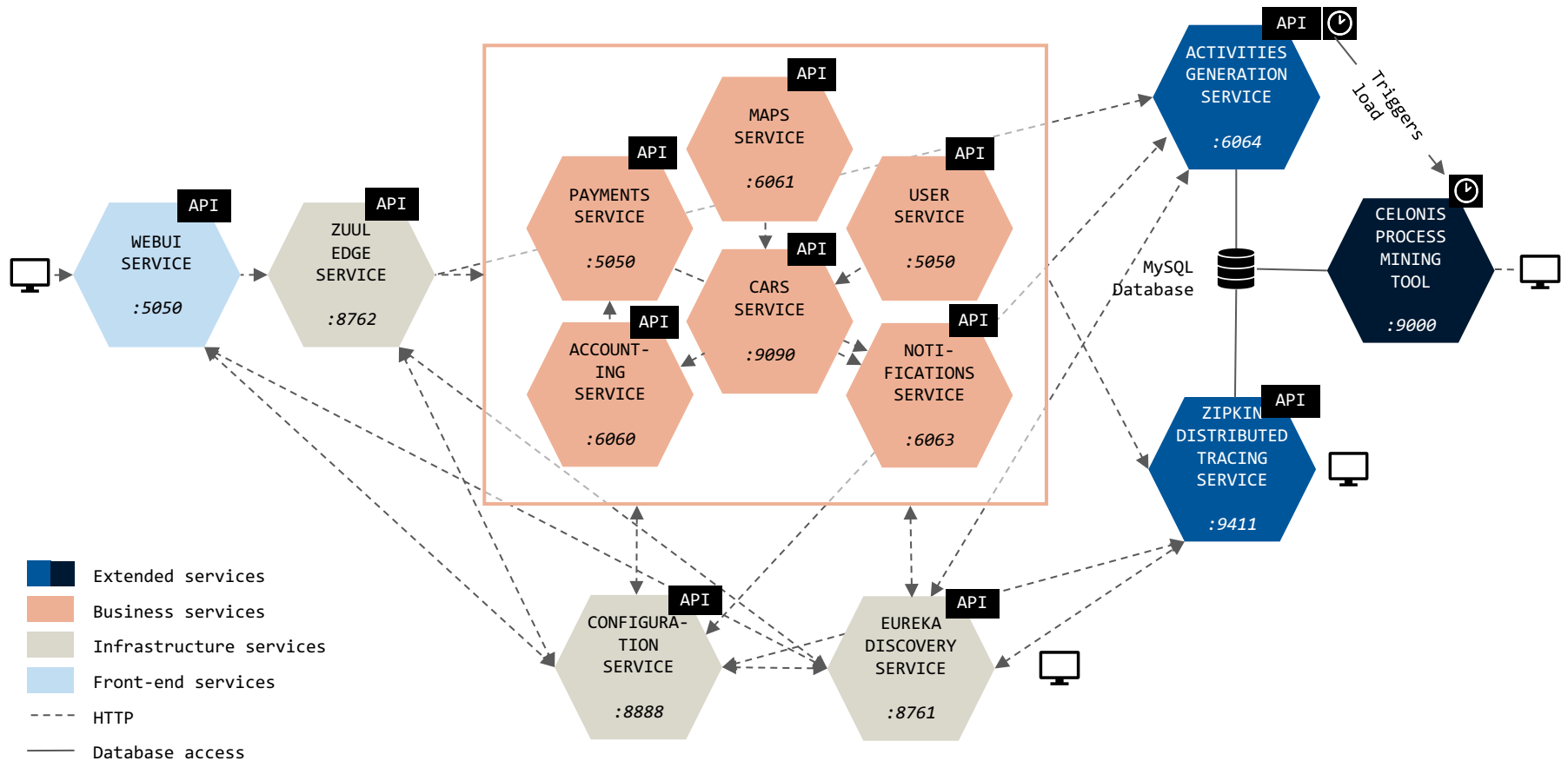
activities table

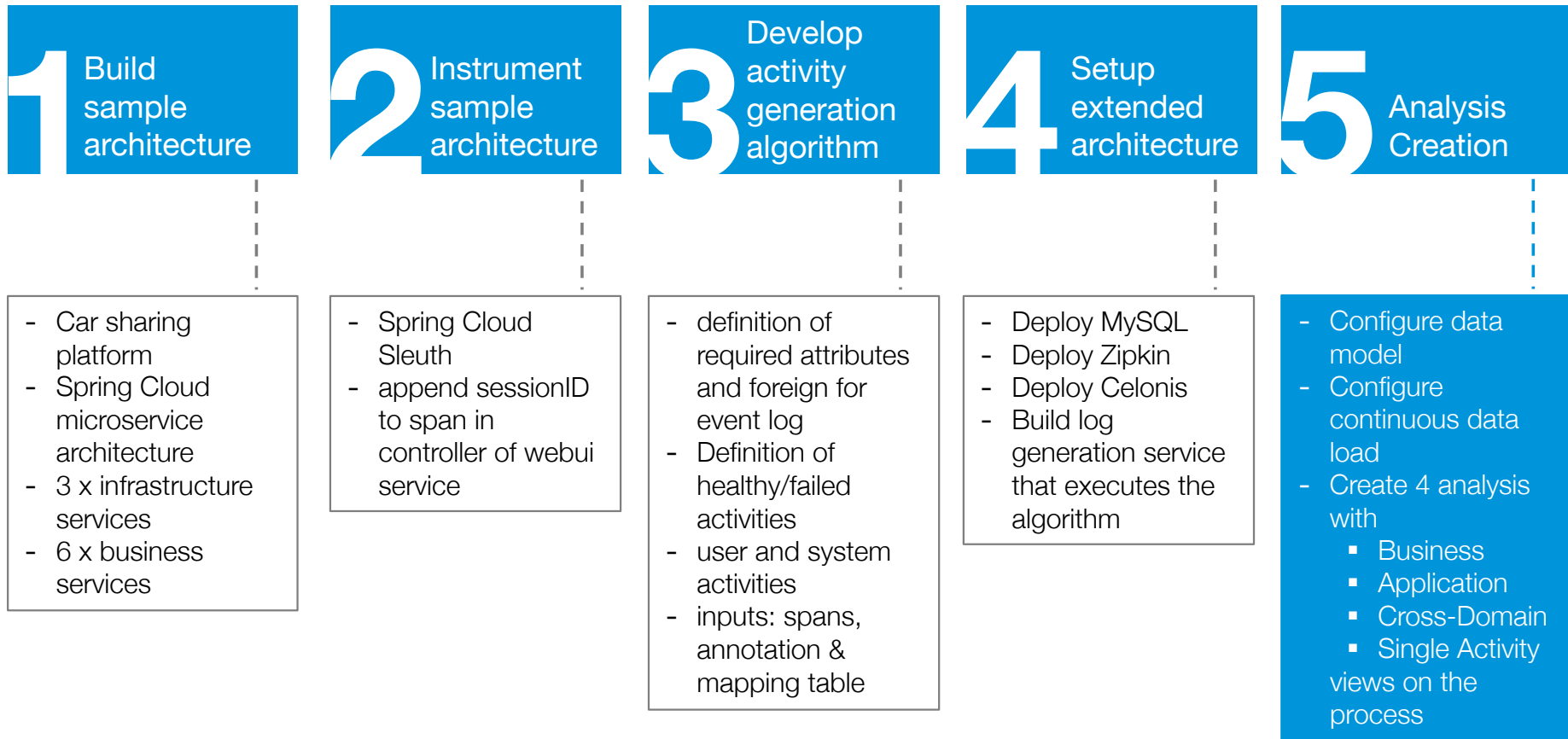
CASE ID	ACTIVITY	START_TS	END_TS	DURATION	TYPE	SERVICE_NAME
12345	Book car	07/06/17 12:45:32.000	07/06/17 12:45:32.800	800 ms	user	webui-service
12345	http://initializebooking	07/06/17 12:45:32.200	07/06/17 12:45:32.700	500 ms	system	accounting-service
12345	http://handlecarbooking	07/06/17 12:45:32.400	07/06/17 12:45:32.500	100 ms	system	cars-service
12345	Unlock car	07/06/17 12:37:32.400	07/06/17 12:37:35.400	3000 ms	user	webui-service
...

Setup extended architecture



Setup extended architecture

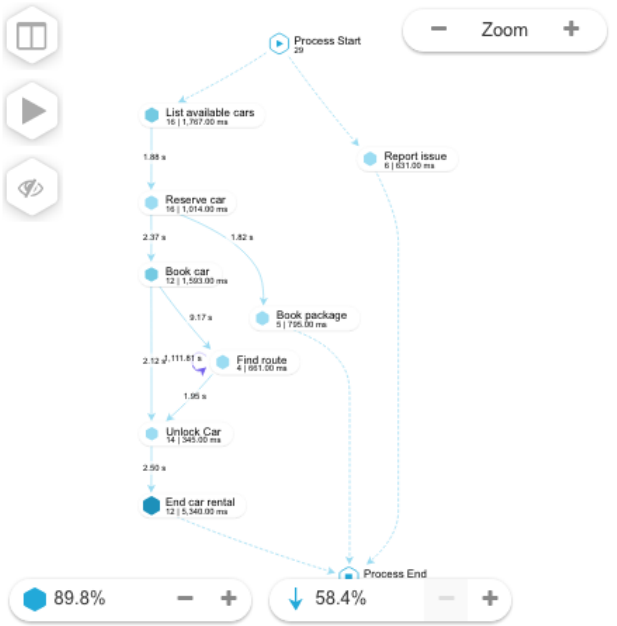




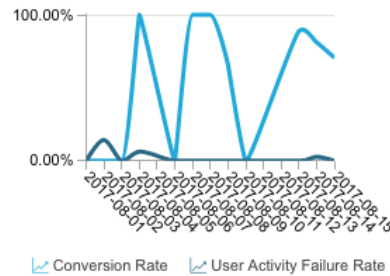
Cross-Domain Analysis

Conversion Rate **41.38%** User Activity Failure Rate **0.95%** AVG clicks to conversion **6** Session Duration w/ c **140 s** Session Duration w/o c **2916 s**

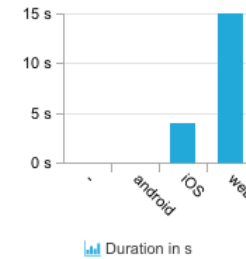
User Click Path



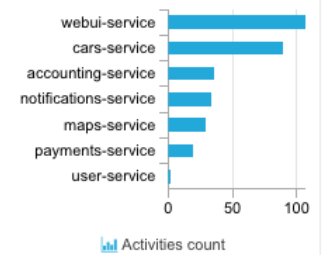
Conversion vs. Activity Failure



Duration: Session S...

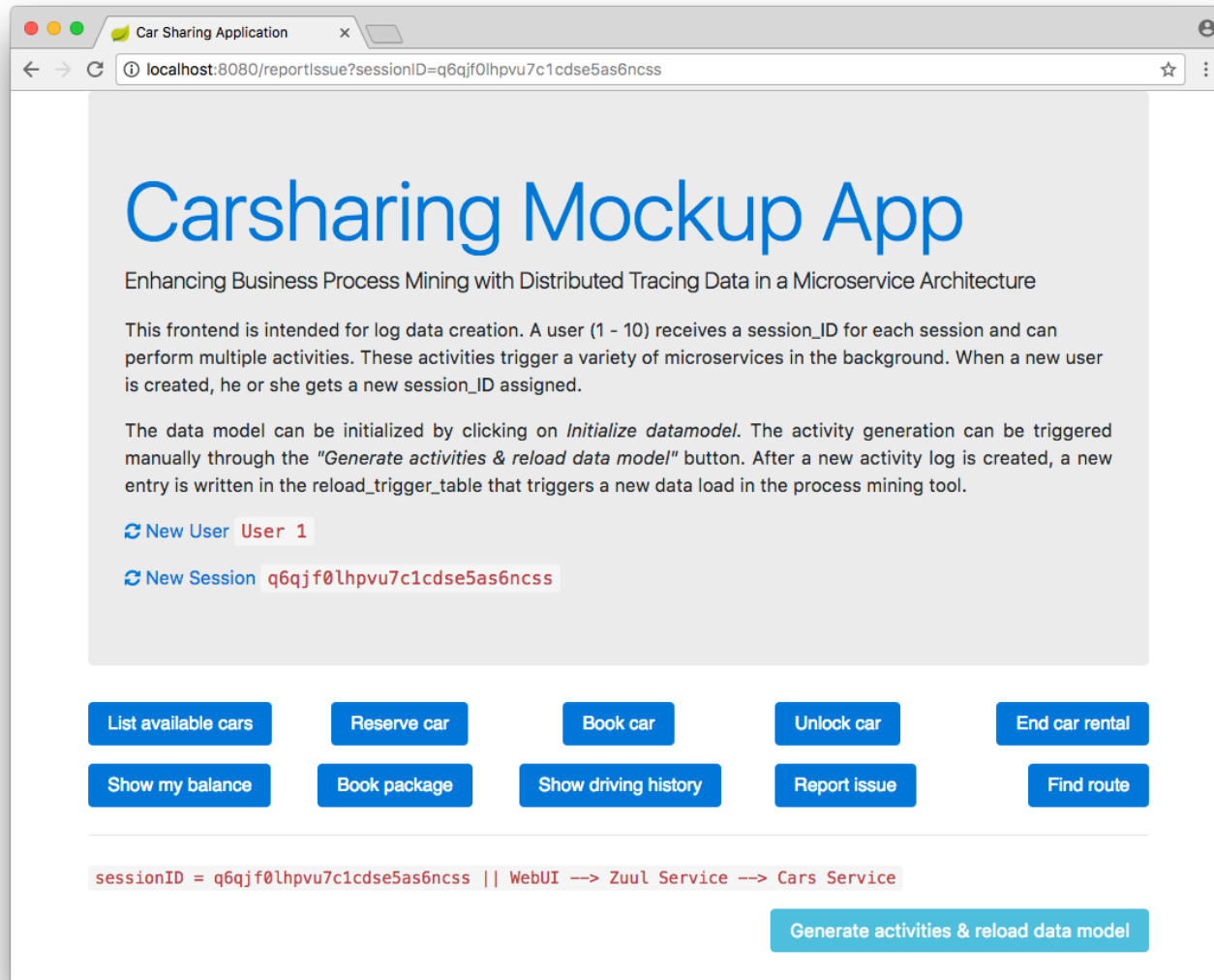


Most critical services



User Process Activity Performance

Activity	Activities count	Duration	Act. Conversion Impact
Book car	13	1593 ms	100.00%
Find route	6	661 ms	100.00%
End car rental failed	1	3512 ms	100.00%
End car rental	13	5340 ms	92.31%
Unlock Car	15	345 ms	86.67%
List available cars	18	1767 ms	61.11%
Reserve car	16	1014 ms	50.00%
Show balance	4	122 ms	50.00%



The screenshot shows a web browser window titled "Car Sharing Application" with the URL `localhost:8080/reportIssue?sessionID=q6qjf0lhpvu7c1cdse5as6ncss`. The main content area features a large blue heading "Carsharing Mockup App" and a subtitle "Enhancing Business Process Mining with Distributed Tracing Data in a Microservice Architecture". Below this, there are two paragraphs of text explaining the app's purpose and usage. The interface includes two status indicators: "New User User 1" and "New Session q6qjf0lhpvu7c1cdse5as6ncss". A grid of ten blue buttons provides various actions: "List available cars", "Reserve car", "Book car", "Unlock car", "End car rental", "Show my balance", "Book package", "Show driving history", "Report issue", and "Find route". At the bottom, a status bar displays `sessionID = q6qjf0lhpvu7c1cdse5as6ncss || WebUI --> Zuul Service --> Cars Service` and a prominent "Generate activities & reload data model" button.

Cross-domain analysis

Provides a more holistic view between the business and application layer

Resource-efficient data source

Resource-efficient (easy to implement) input source for process mining in microservice architectures

Portability

Approach transferable to different architectures with limited effort (i.a. due to OpenTracing standard)

Ubiquity

Ubiquitous through distributed tracing becoming a standard tool for microservice debugging

Flexibility on process perspectives

Process scope flexibility through appending *spans* with arbitrary IDs

Bottom up process discovery

Bottom up process discovery in legacy systems

System under survey (SUS)

Approach only tested on SUS

- Single architecture
- Data volume
- Data contents

Process visualisation of system activities

Petri nets not a suitable visualisation method for system activities

Performance overhead

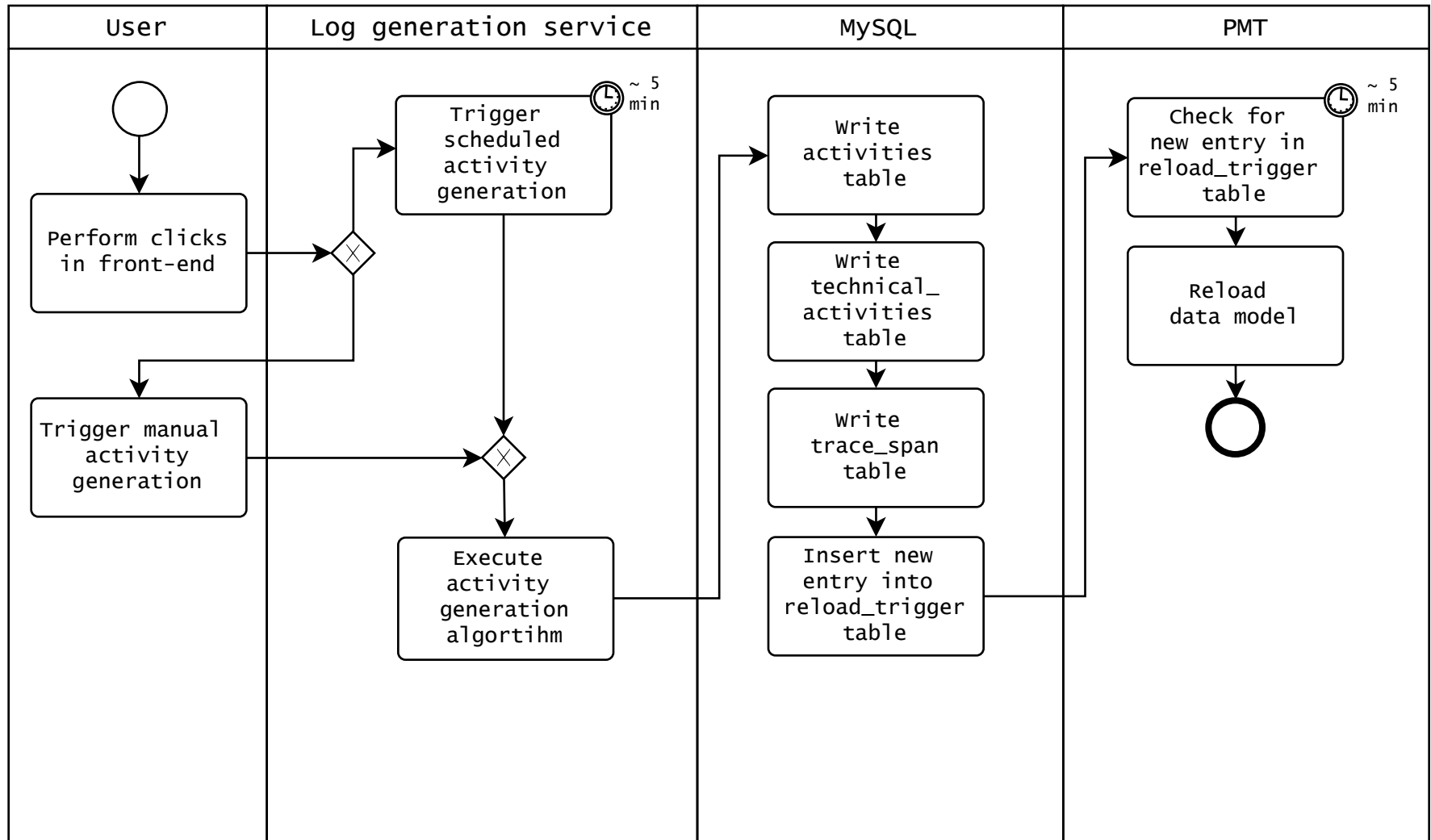
Necessity for SamplingRate=1.0 for capturing whole process instances leads to performance overhead

Real-time event handling

Presented prototype only generates event log and reloads data model every 5 min

- Workflow of activity generation and data reloading
- Related work
- Process Mining
- Inputs for activity generation
- Distributed tracing
- User request and span/trace context
- *„End car rental“* user activity sequence diagram

Workflow of activity generation and data reloading



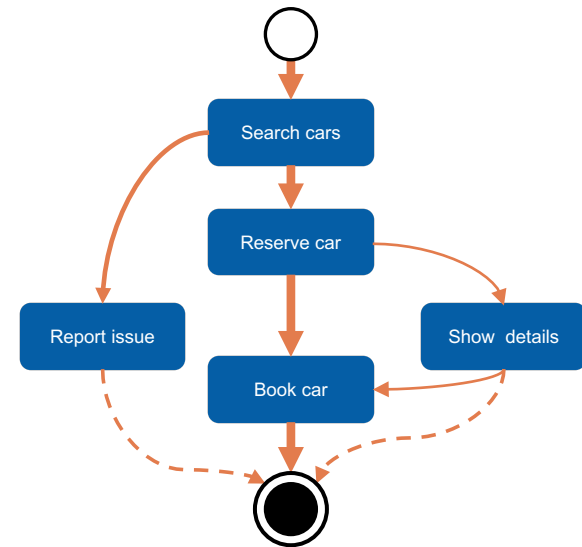
	Log origin	Activity types	Captured behaviour	Type of work	Evaluation environment	System architecture	Language independence	(Near) real-time
<i>Poggi et al. [53]</i>	Web logs	User Activities	Business	Algorithm evaluation	Real-life event logs	No	n/a	No
<i>Abe & Kudo [7]</i>	Web logs	User activities	Business	Framework	Real-life event logs	n/a	n/a	No
<i>Bruckmann et al. [13]</i>	n/a	User Activities, system activities	Business and system	Architecture proposal	n/a	n/a	n/a	Yes
<i>Leemans & van der Aalst [41]</i>	Joinpoint-pointcut model instrumentation	User activities	System	Instrumentation strategy, implementation	Real-life event logs	Yes	Yes	No
<i>Rubin et al. [59]</i>	Custom instrumentation	User activities, system activities	User and system	Experimental	Real-life event logs	No	n/a	No
<i>Proof-of-concept prototype of this work</i>	Distributed tracing instrumentation	User activities, system activities	Business, user and system	Instrumentation strategy, architecture description, implementation	Simulated user requests on testing system	Yes	Yes	Yes

“The idea of process mining is to **discover, monitor and improve real processes** (i.e., not assumed processes) by extracting **knowledge from event logs** readily available in today's (information) systems.”

IEEE CIS Task Force on Process Mining

Event Log

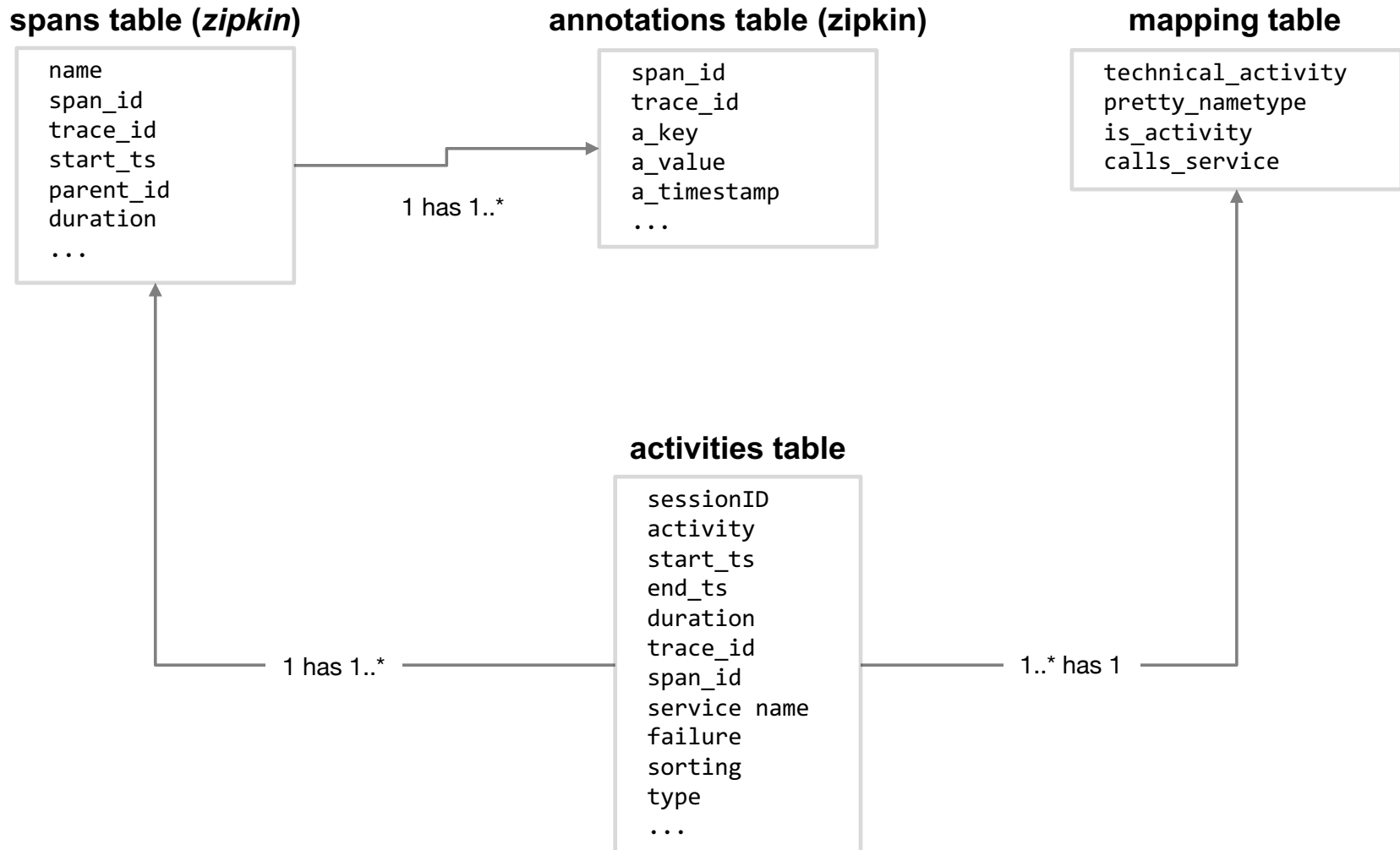
TIMESTAMP	ACTIVITY	CASE ID
2016-03-05 22:38:41.868	SEARCH CARS	#1234
2016-03-05 23:46:32.306	REPORT ISSUE	#5678
2016-03-05 23:47:42.321	RESERVE CAR	#1234
2016-03-05 23:53:12.354	SHOW DETAILS	#9012
...



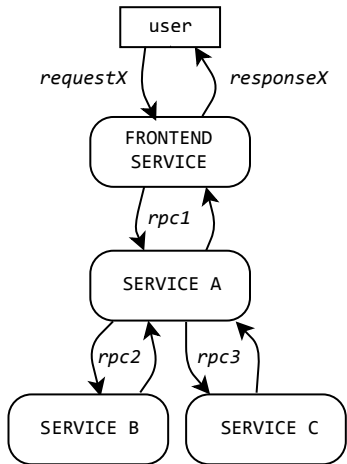
There are three Classes of Process Mining:

1. Process Discovery
2. Conformance Checking
3. Extension

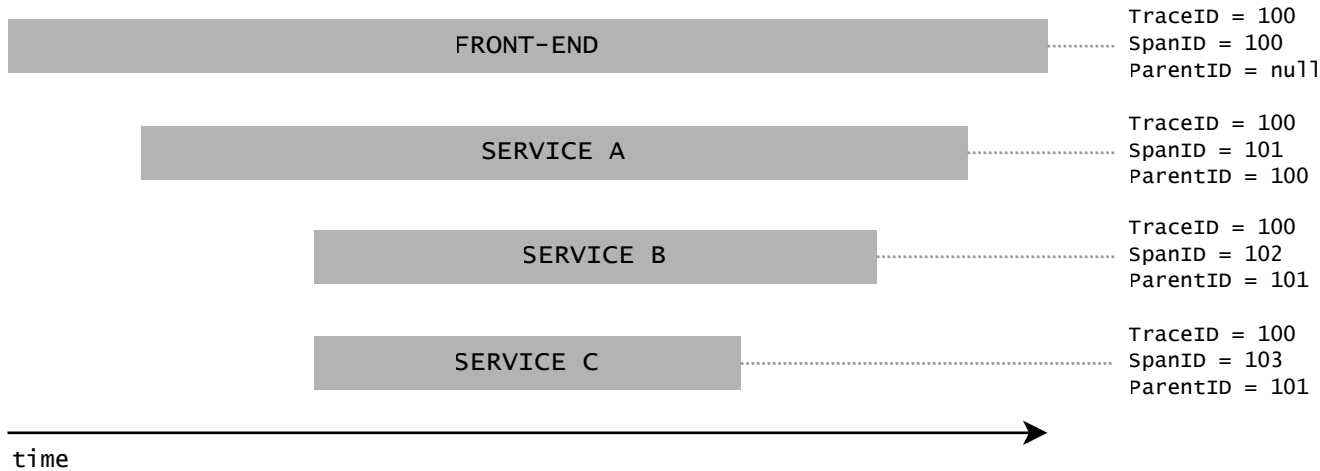
Develop activity generation algorithm



Distributed tracing

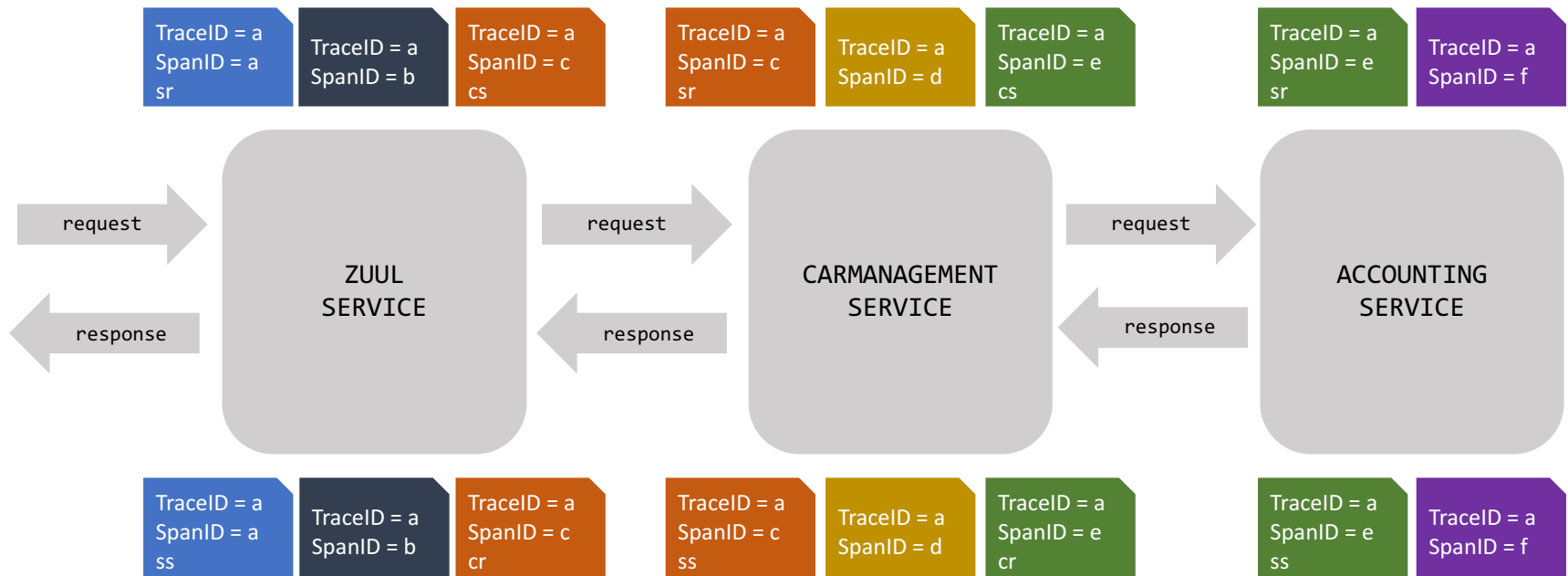


The path taken through a simple servicing system on behalf of user request X.



The causal and temporal relationship between four spans of a trace.

User request and span/trace context



,End car rental' user activity

