



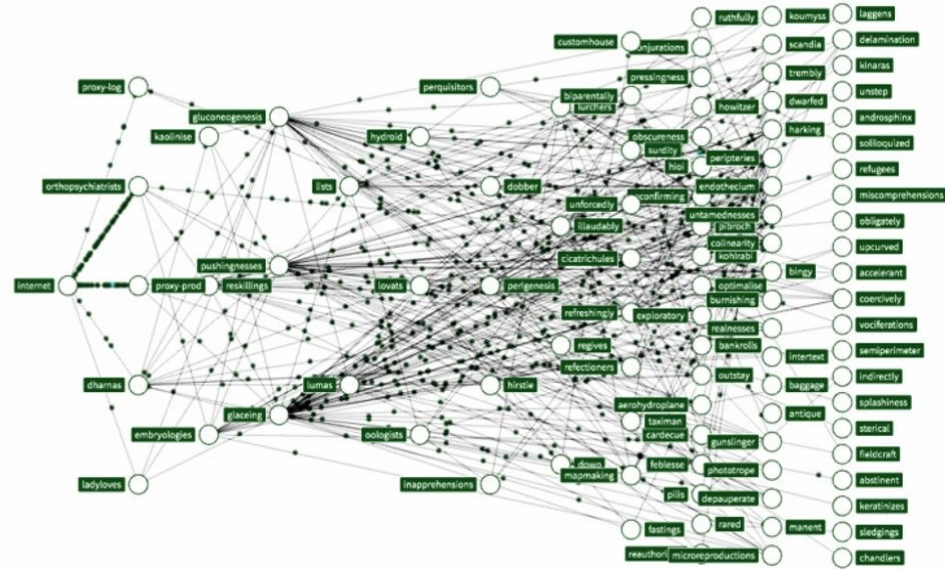
# DTraComp: Comparing distributed execution traces for understanding intermittent latency sources

*Maryam Ekhlesi*  
Dec. 8<sup>th</sup>, 2023

Polytechnique Montreal

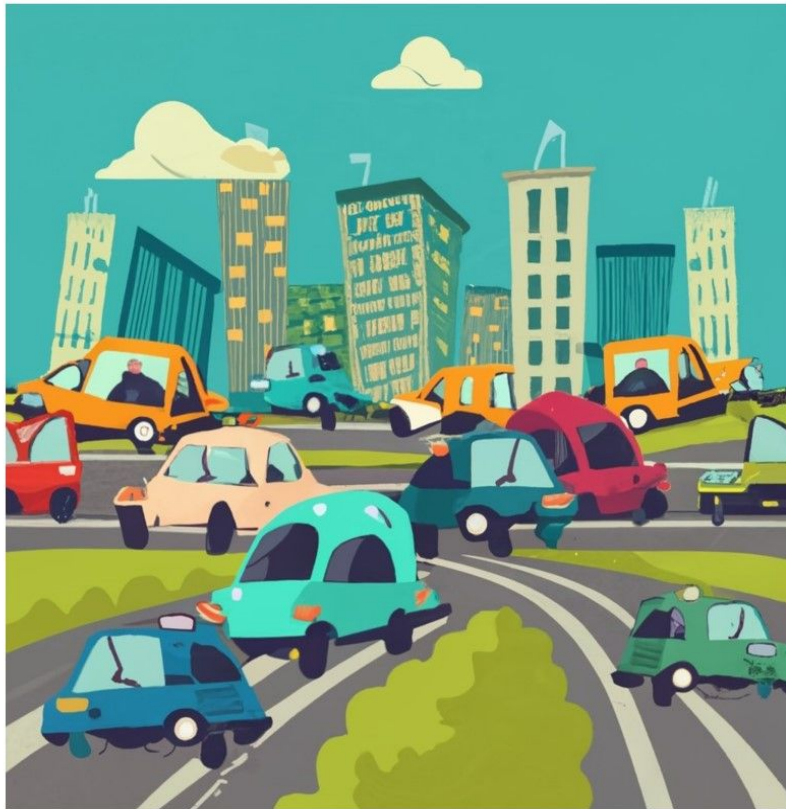
**DORSAL** Laboratory

# Welcome to Microservice City!




Reference: <https://www.honeycomb.io/microservices>

latency issue!





# Distributed Trace Compare (DTraComp) Enter!

[DOWNLOAD PDF](#)   80  1

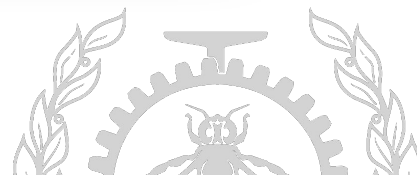
## DTraComp: Comparing distributed execution traces for understanding intermittent latency sources

[DISTRIBUTED SYSTEM](#) [OPERATING SYSTEMS](#) [PERFORMANCE ANALYSIS](#) [PERFORMANCE COMPARISON](#)  
[SOFTWARE VISUALIZATION](#) [TRACING](#)

 +3 **Maryam Ekhlesi** , **Fatemeh Faraji Daneshgar**, **Michel Dagenais**, **Maxime Lamothe**, **Naser Ezzati-Jivan**, **Matthew Khouzam**

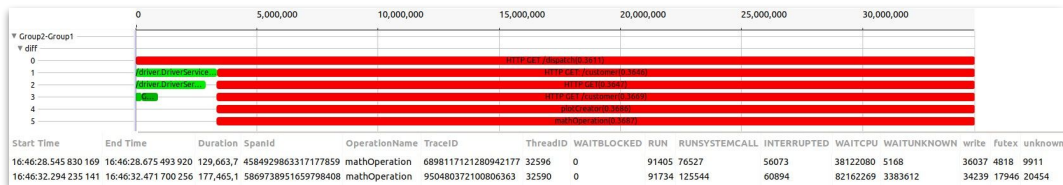


WILEY  Online Library



# Our Goal

1. Finding and locating the performance problem through a distributed system.
2. Comparing two sets of executions to evaluate the differences in terms of performance.
3. Providing sets of views to highlight the differences and speed up problem diagnosis.



Trace	Timestamp	Channel	CPU	Event type	Contents
kernel	13:40:43.717 126 771	kernelchannel_13	13	timer_brtimer_cancel	brtimer-d0fffb0a34dfc08, context.packet_seq_num=8, context.cpu_id=13, context_tid=37692, context_pid=37692
kernel	13:40:43.717 126 901	kernelchannel_5	5	sched_switch	prev_comm=code, prev_pid=37726, prev_prio=20, prev_state=1, next_comm=swapper/5, next_pid=0, next_prio=20, context.packet_seq_num=4, context.cpu_id=5, context_tid=37726, context_pid=37726
kernel	13:40:43.717 127 152	kernelchannel_10	10	sched_switch	prev_comm=swapper/70, prev_pid=0, prev_prio=20, prev_state=0, next_comm=worker/6642, next_tid=21275, next_prio=20, context.packet_seq_num=13, context.cpu_id=10, context_tid=0, context_pid=0
usr/uid/1000/64	13:40:43.717 127 392	userchannel_14	14	jaeger_user_start_span	trace_id_high=0, trace_id_low=6431679566420986618, span_id=313371220428027233, parent_span_id=7549595779124019993, op_name=SQL_SELECT, start_time=168175343717082902, context.packet_seq_num=0, context.cpu_id=14, context_tid=0, context_pid=0
kernel	13:40:43.717 127 412	kernelchannel_1	1	xmem_mm_page_alloc	page-d0fffb2c24f62800, pfn=1308832, order=5, gfp_flags=1387968, migrateype=0, context.packet_seq_num=7, context.cpu_id=1, context_tid=212777, context_pid=212777
kernel	13:40:43.717 127 512	kernelchannel_0	0	xmem_cache_free	call_site-d0fffb2c24f62800, page-d0fffb2c24f62800, karm=0, head, context.packet_seq_num=0, context.cpu_id=0, context_tid=0, context_pid=0
kernel	13:40:43.717 127 793	kernelchannel_15	15	power_cpu_idle	state=1, cpu_id=15, context.packet_seq_num=10, context.cpu_id=15, context_tid=0, context_pid=0
kernel	13:40:43.717 128 384	kernelchannel_1	1	xmem_kmalloc_node	call_site-d0fffb2c24f62800, bytes_req=75968, bytes_alloc=131072, gfp_flags=77248, node=1, context.packet_seq_num=7, context.cpu_id=1, context_tid=212777, context_pid=0
kernel	13:40:43.717 128 404	kernelchannel_8	8	xmem_mm_page_alloc	page-d0fffb2c24f62800, pfn=5811199, order=0, gfp_flags=1782966, migrateype=1, context.packet_seq_num=5, context.cpu_id=8, context_tid=6810, context_pid=6804

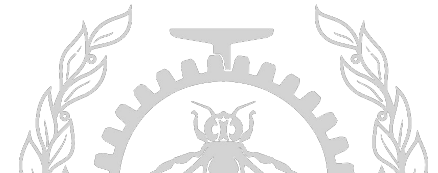
Span Metrics Summary	Start Time	End Time	Duration	Spanid	OperationName	TraceID	ThreadID	WAITBLOCKED	RUN	RUNSYSTEMCALL	INTERRUPTED	WAITCPU	WAITUNKNOWN	WAITFORK	UNKNOWN	write	futex	unknown	getrlimit
	13:40:41.168 876 485	13:40:41.171 187 315	2,310,830	2504674136162342969	HTTP GET /	2504674136162342969	216343	7549813024	691080 51370	0	3686	0	0	0	0	0	0	5409	0
	13:40:41.176 018 770	13:40:41.176 217 423	188,653	58019195889990802	HTTP GET /	58019195889990802	216344	0	148632 55055	14955	0	0	0	0	0	0	6386 1862	0	0
	13:40:43.715 837 710	13:40:43.716 105 152	267,462	21148072264770919	HTTP GET /confp	21148072264770919	216469	0	282347 7552	0	0	0	0	0	0	0	4377	0	0
	13:40:43.717 127 392	13:40:48.718 272 5001, 144,494		313371220428027233	SQL_SELECT	6431679566420986618	216470	8999907384	601351 441102	0	313548	0	0	0	0	0	22870 1411721310	0	0
	13:40:43.716 961 761	13:40:48.718 093 038	5,001,731,877	7549595779124019993	HTTP GET /	7549595779124019993	216470	5000483764	740190 558837	0	323105	0	0	0	0	0	42837 1411799207 66214	0	0
	13:40:43.716 241 819	13:40:48.718 977 441	5,002,735,622	4197584092289167312	HTTP GET	6431679566420986618	216344	5002564971	258680 160314	43861	0	0	0	0	0	0	5002600936	0	0



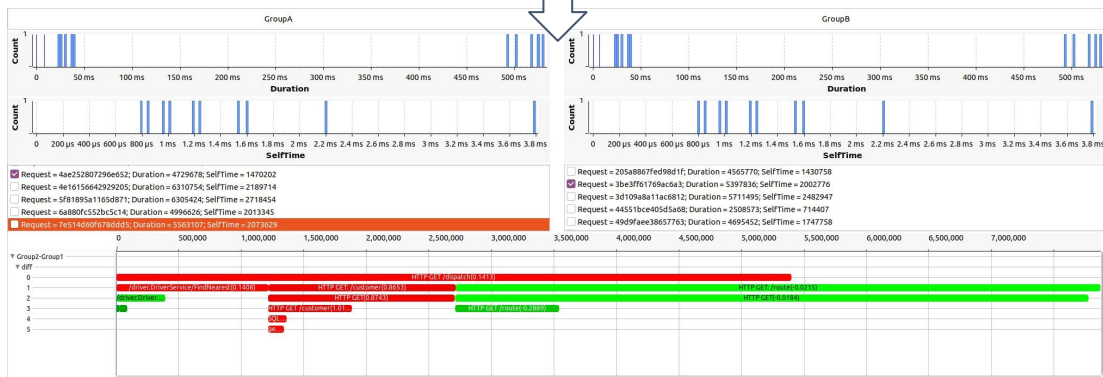
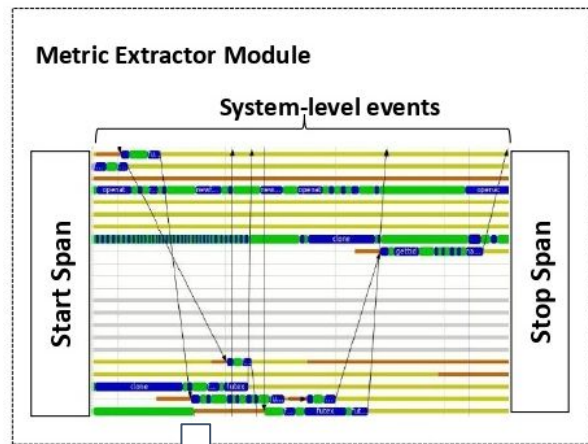
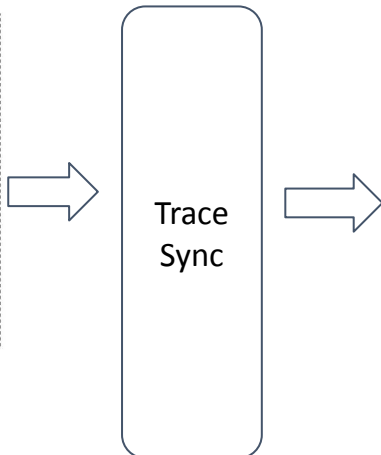
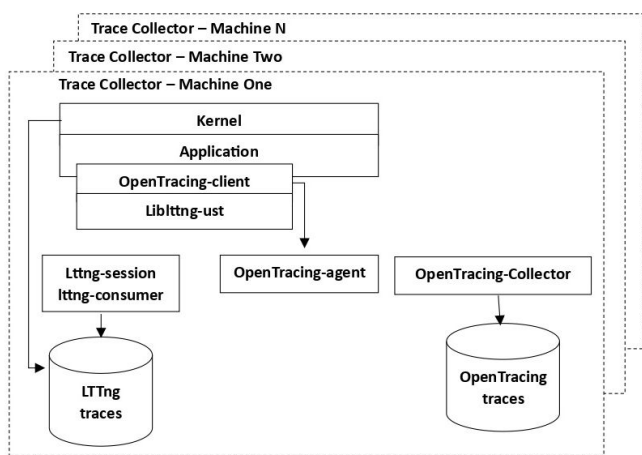
# Our Goal

---

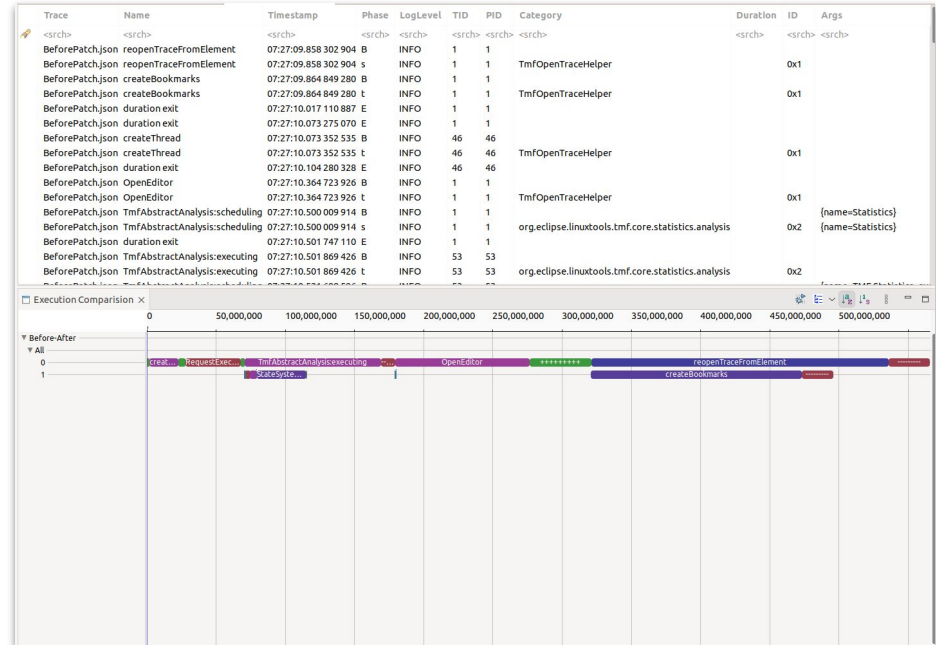
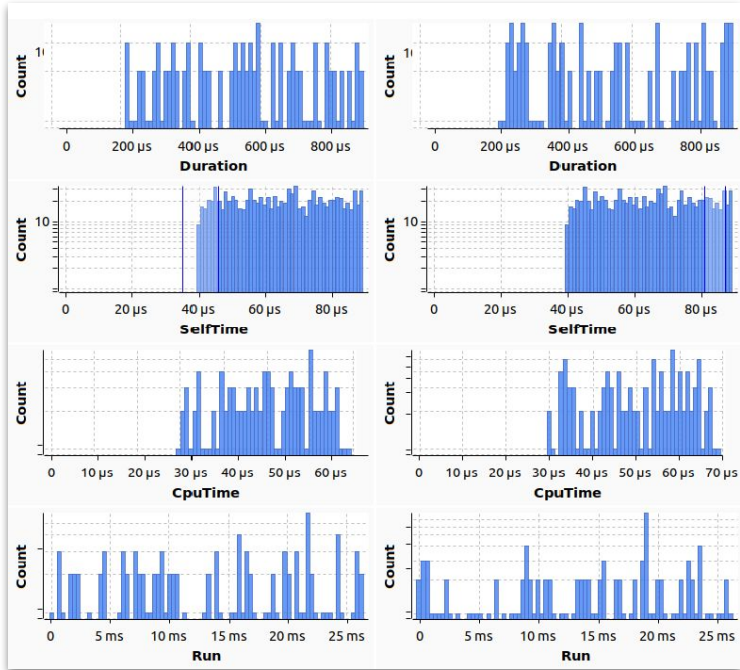
1. Locating the performance problem through a distributed system. (Accomplished)
2. Comparing two sets of executions to evaluate the differences in terms of performance. (Accomplished)
3. Providing sets of views to highlight the differences and speed up problem diagnosis. (Accomplished)
4. Grouping similar requests, with closely related but not identical structure. (Ongoing)
5. Finding the normal execution threshold for each group. (Ongoing)



# Architecture

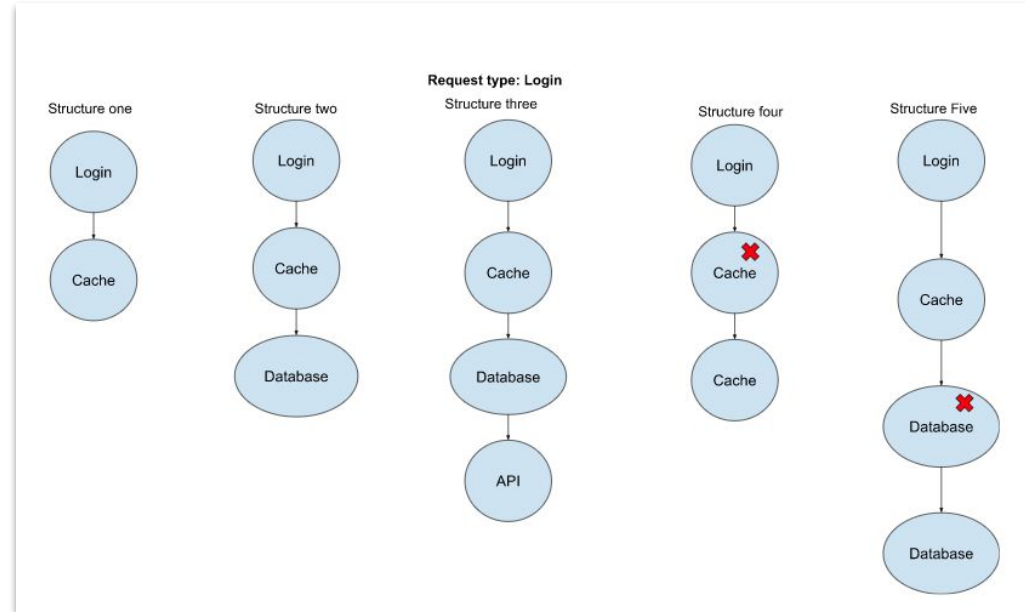
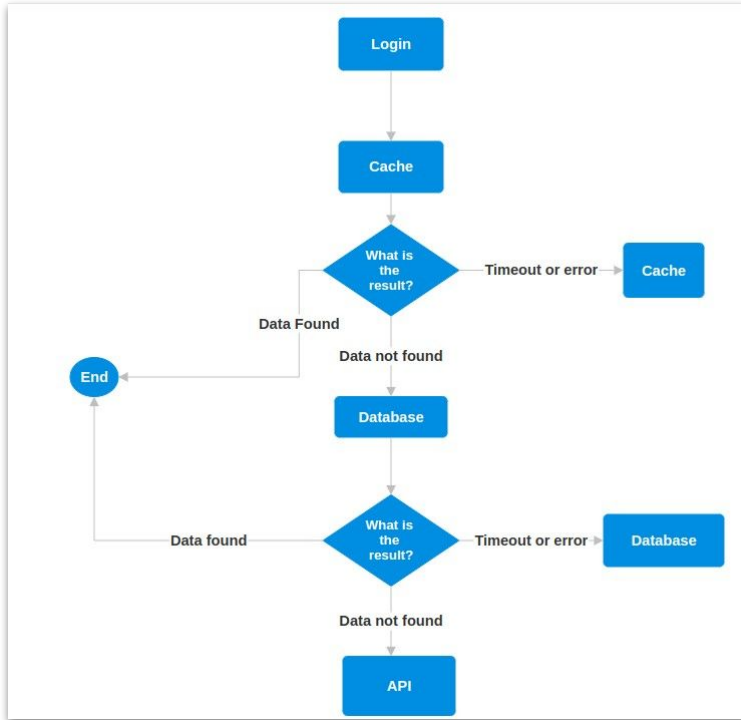


# Filtering Module

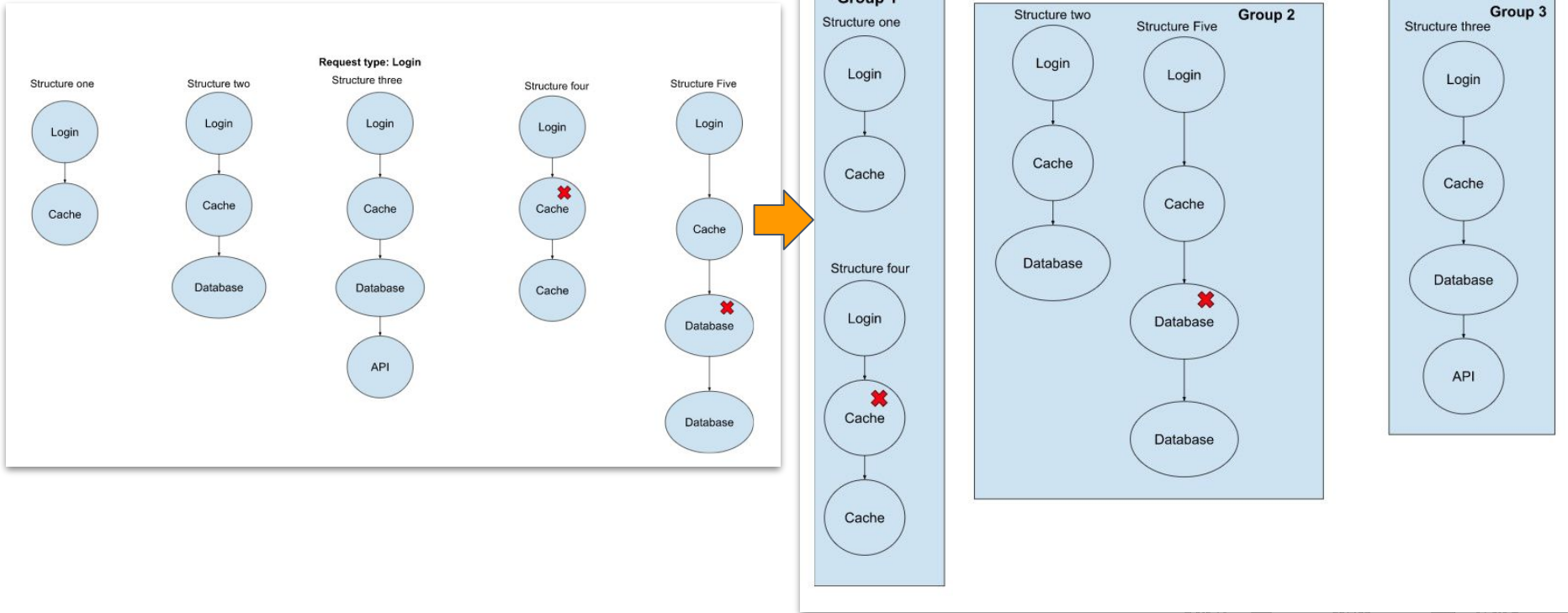




# Example



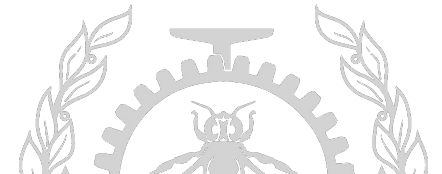
# Different Structures



# Concerns

---

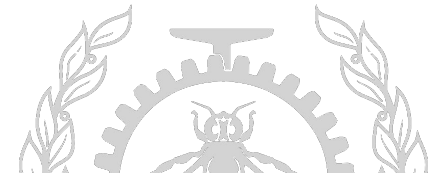
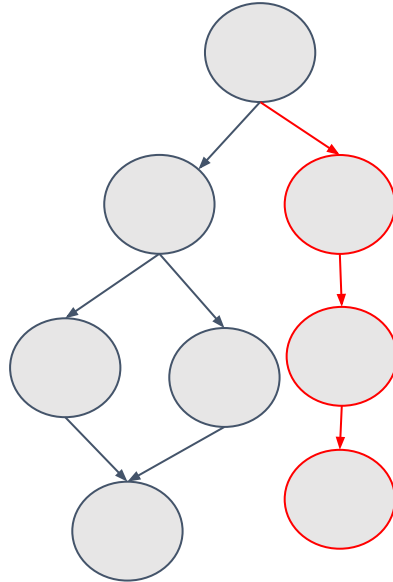
1. One simple request may have several different structures.
2. Graph/tree mining approaches are computationally expensive.
3. Parallel services (branches) can be challenging.



# Strategy

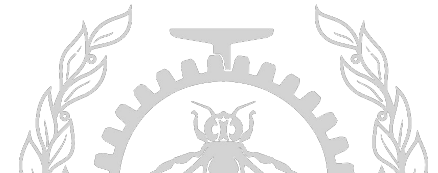
---

1. Critical Path may be a good criteria for performance related comparisons.
2. Using Statistical approaches for finding normal execution threshold for each group.



I need your **help!**

Real industrial requests with different structures but semantically similar.





---

# Thank you

**Email:** [maryam.ekhlasi@polymtl.ca](mailto:maryam.ekhlasi@polymtl.ca)

