Università della Svizzera italiana

Faculty of Informatics

ESTIMATING THE COST FOR EXECUTING BUSINESS PROCESSES IN THE CLOUD

Vincenzo Ferme, Ana Ivanchikj, Cesare Pautasso

Faculty of Informatics

USI Lugano, Switzerland

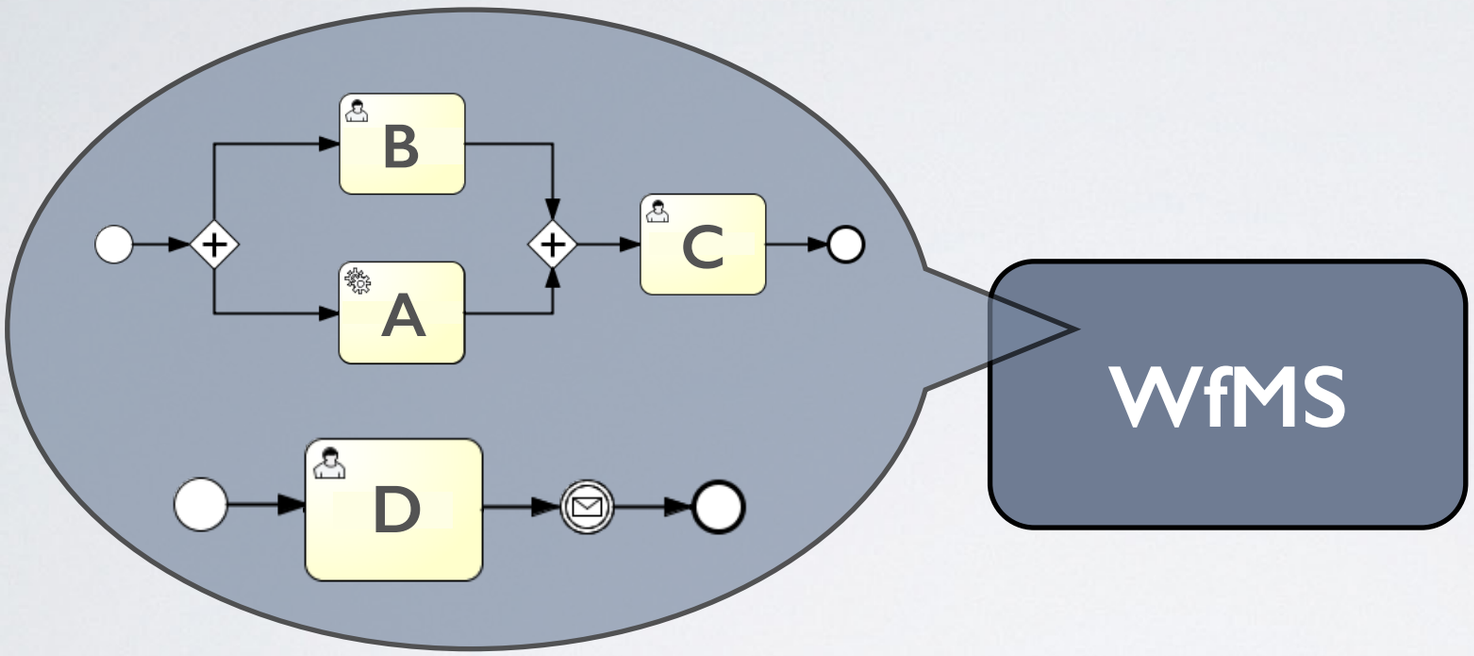
Deploying WfMSs to the Cloud

sweet spot in the performance vs. resource consumption trade-off

WfMS

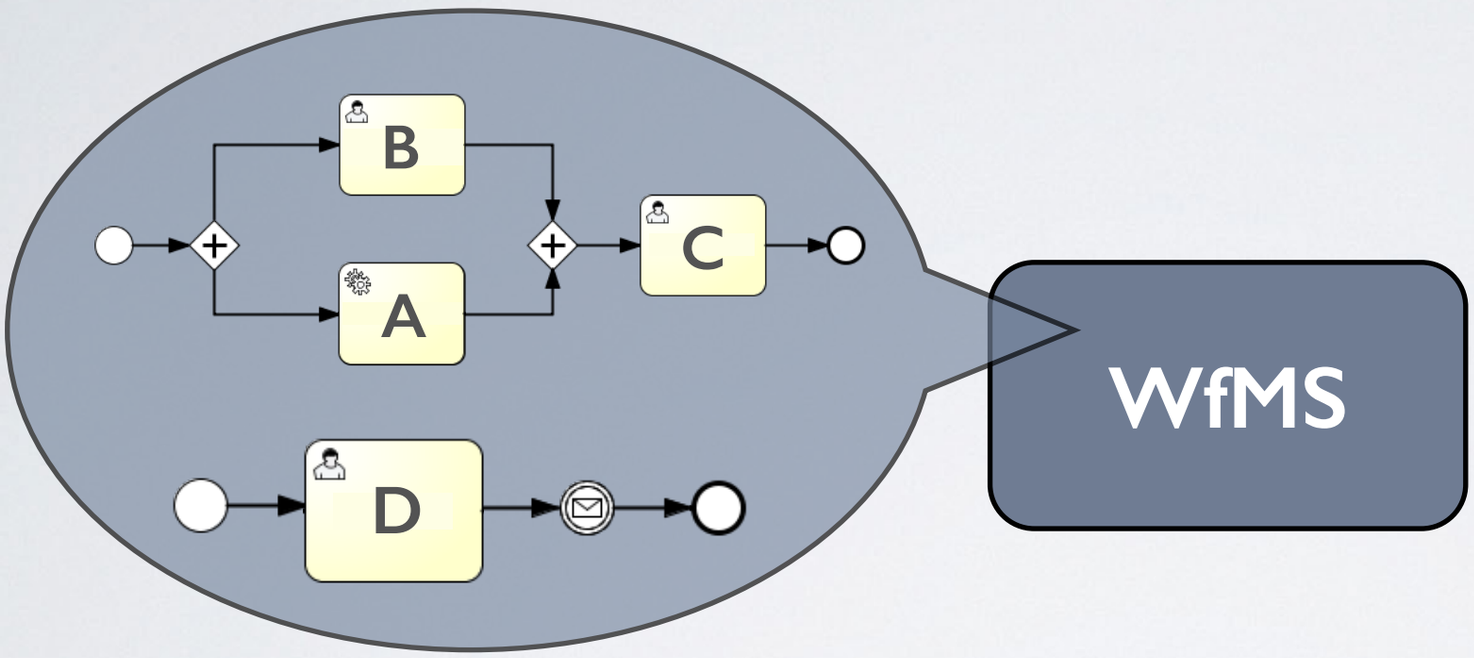
Deploying WfMSs to the Cloud

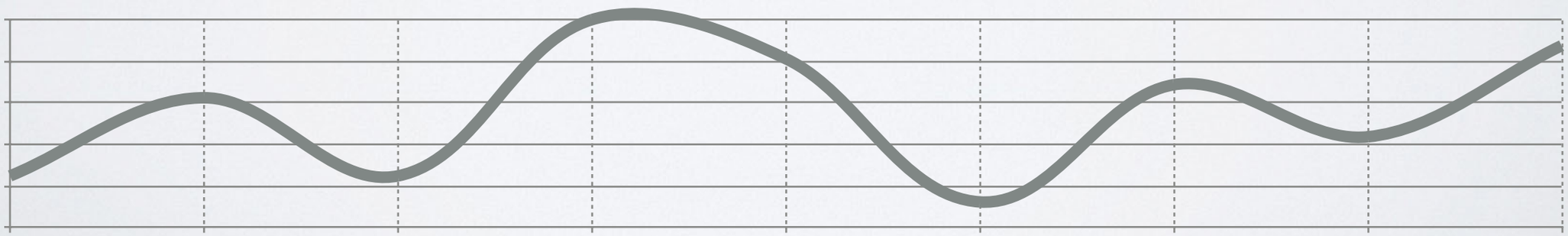
sweet spot in the performance vs. resource consumption trade-off



Deploying WfMSs to the Cloud

sweet spot in the performance vs. resource consumption trade-off

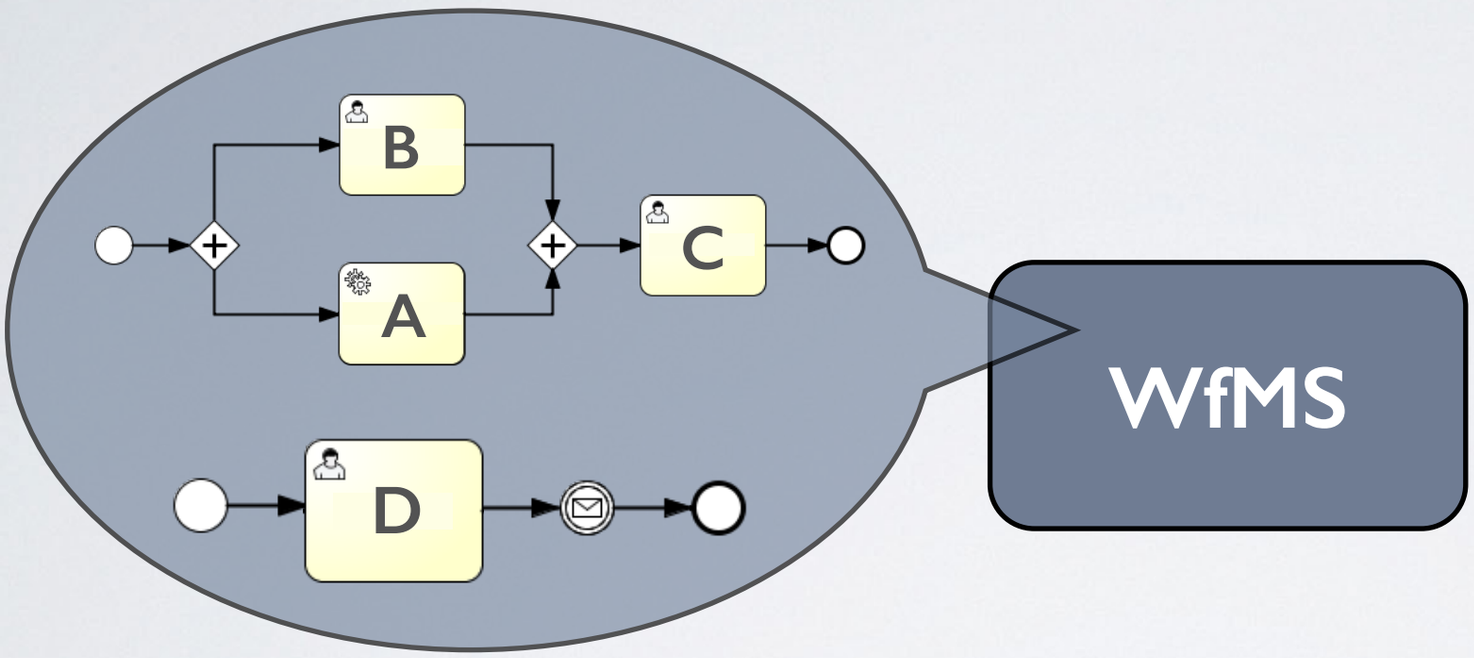


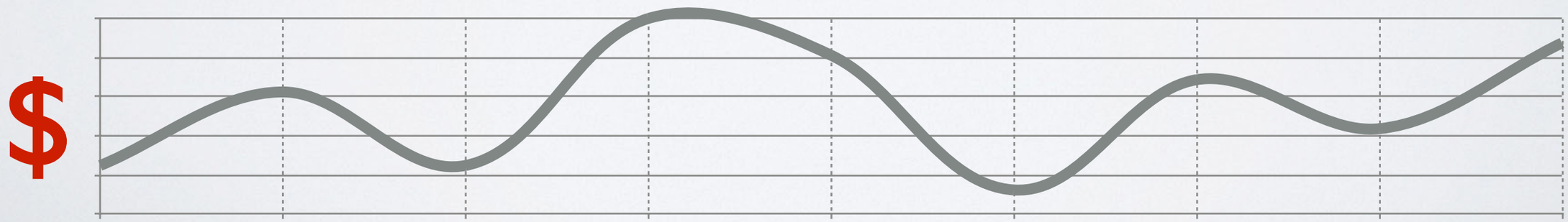


IT Resources

Deploying WfMSs to the Cloud

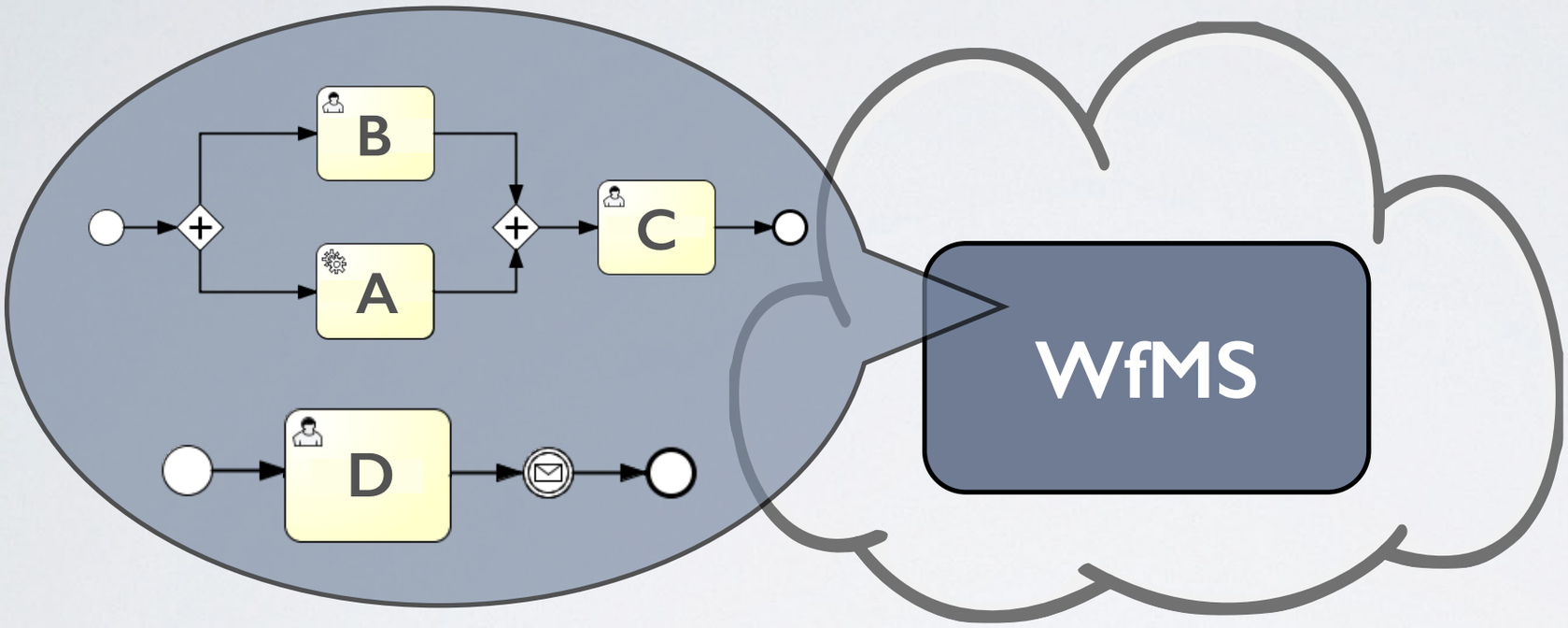
sweet spot in the performance vs. resource consumption trade-off



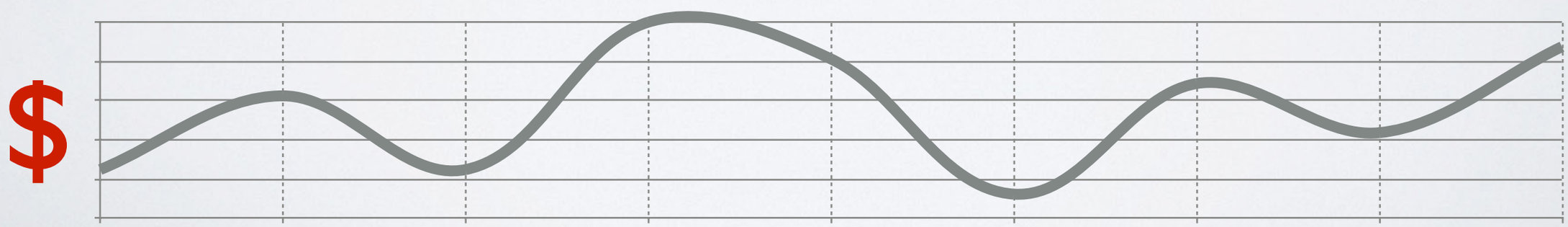


Deploying WfMSs to the Cloud

sweet spot in the performance vs. resource consumption trade-off

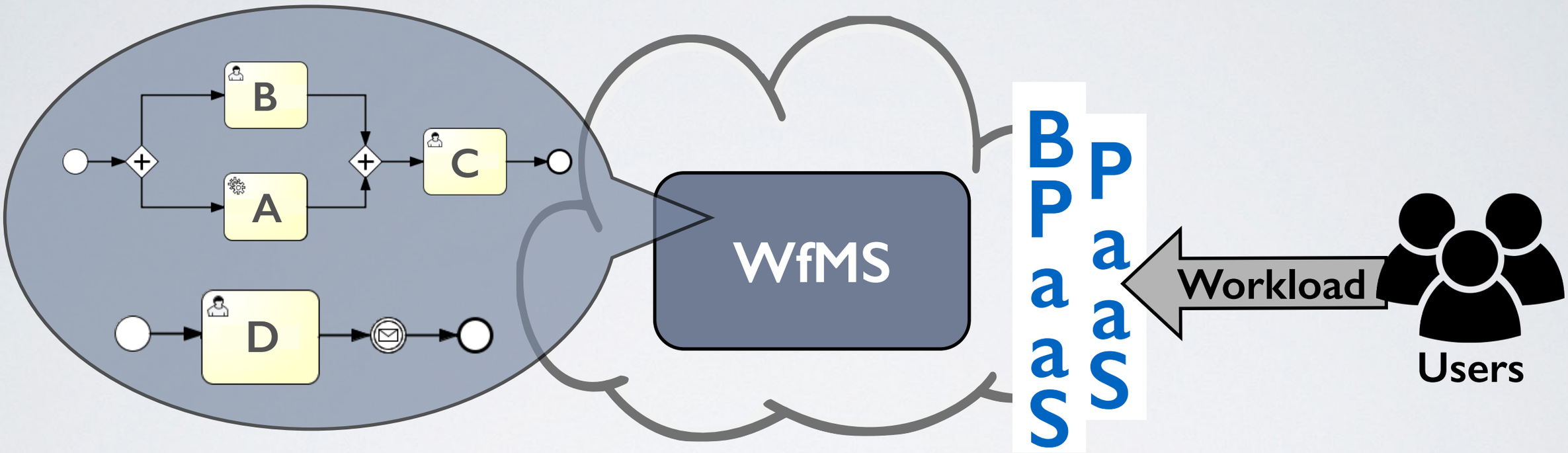


IaaS

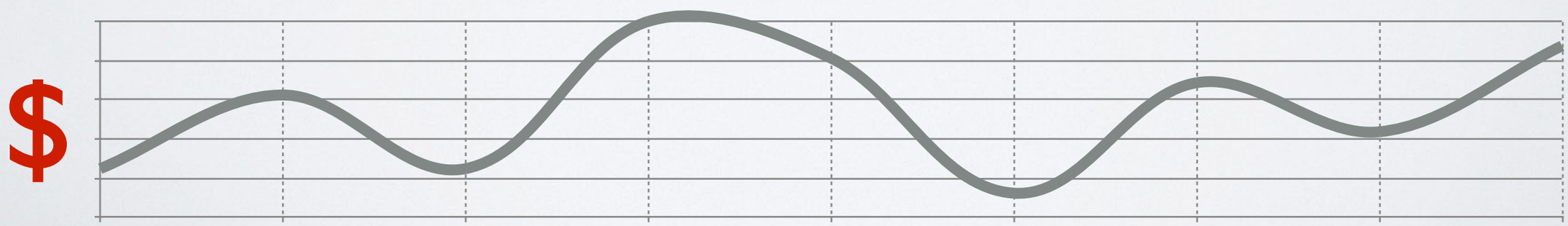


Deploying WfMSs to the Cloud

sweet spot in the performance vs. resource consumption trade-off

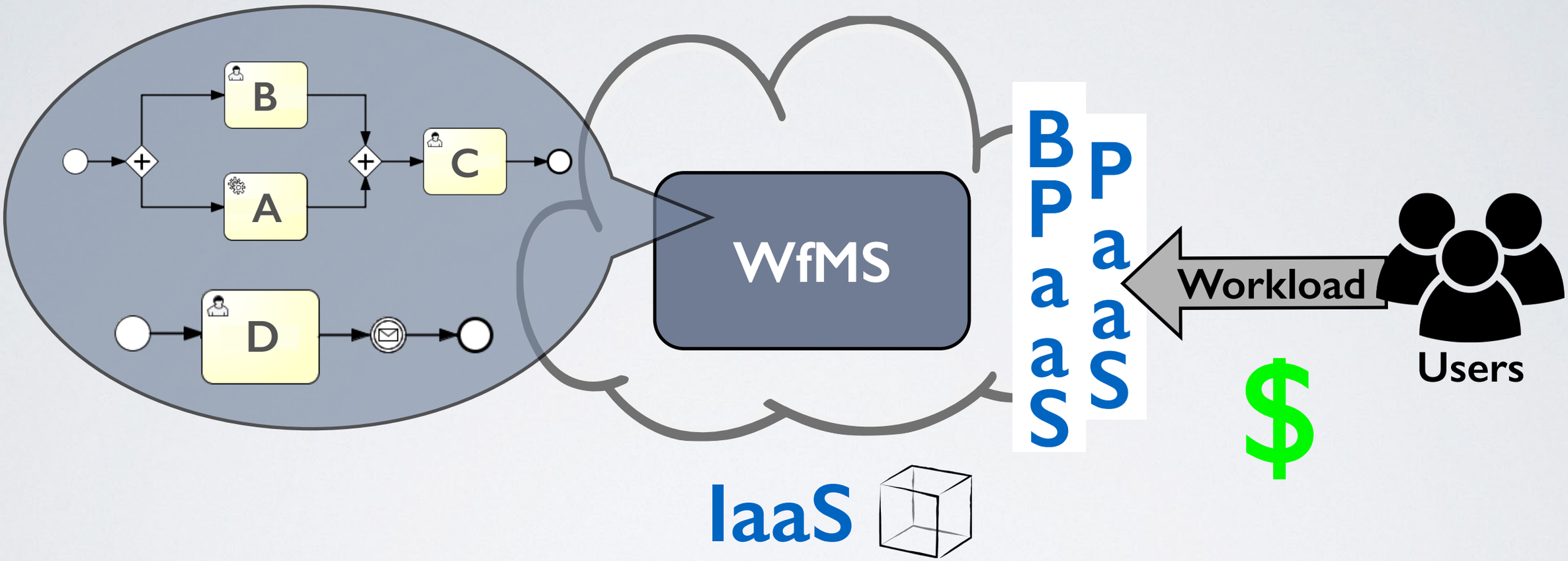


IaaS



Deploying WfMSs to the Cloud

sweet spot in the performance vs. resource consumption trade-off



Challenges in Deploying WfMSs in the Cloud

evaluate the Cloud cost models, determine the best suitable provider

Evaluate the Cloud Offers

Challenges in Deploying WfMSs in the Cloud

evaluate the Cloud cost models, determine the best suitable provider

Microsoft Azure

Google Cloud Platform

amazon web services

Springs.io

Evaluate the Cloud Offers

Challenges in Deploying WfMSs in the Cloud

evaluate the Cloud cost models, determine the best suitable provider

Microsoft Azure

Google Cloud Platform

amazon web services

Springs.io

Evaluate the Cloud Offers **Best Suitable Provider?**

Challenges in Deploying WfMSs in the Cloud

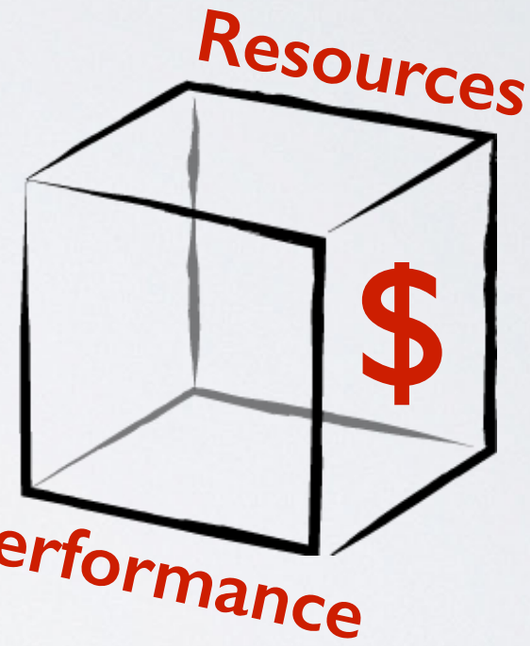
evaluate the Cloud cost models, determine the best suitable provider

Microsoft Azure

Google Cloud Platform

amazon web services

Springs.io



Evaluate the Cloud Offers

Best Suitable Provider?

Goal

select candidate VMs according to workload and wanted performance

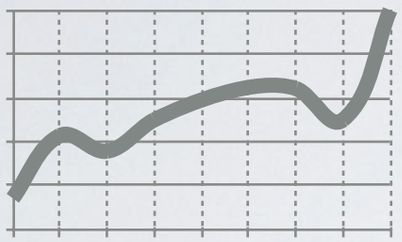
WfMS

Goal

select candidate VMs according to workload and wanted performance

Goal

select candidate VMs according to workload and wanted performance

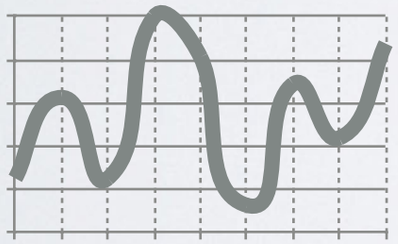


RAM USAGE

Resp. Time

WfMS

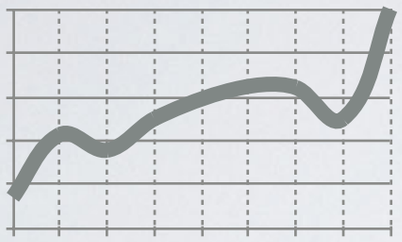
Throughput



CPU USAGE

Goal

select candidate VMs according to workload and wanted performance



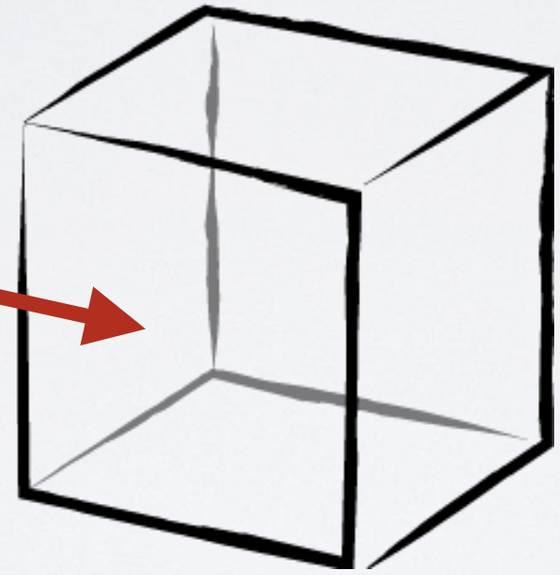
RAM USAGE

Resp. Time

WfMS

Throughput

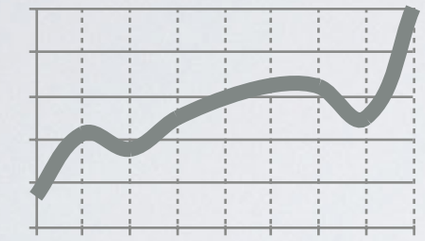
CPU USAGE



CPU + RAM Bound

Goal

select candidate VMs according to workload and wanted performance

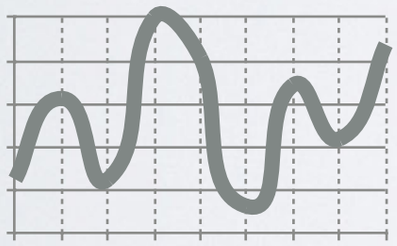


RAM USAGE

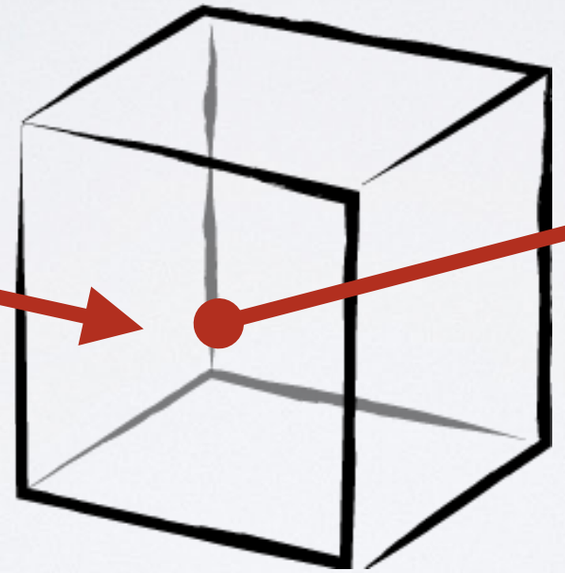
Resp. Time

WfMS

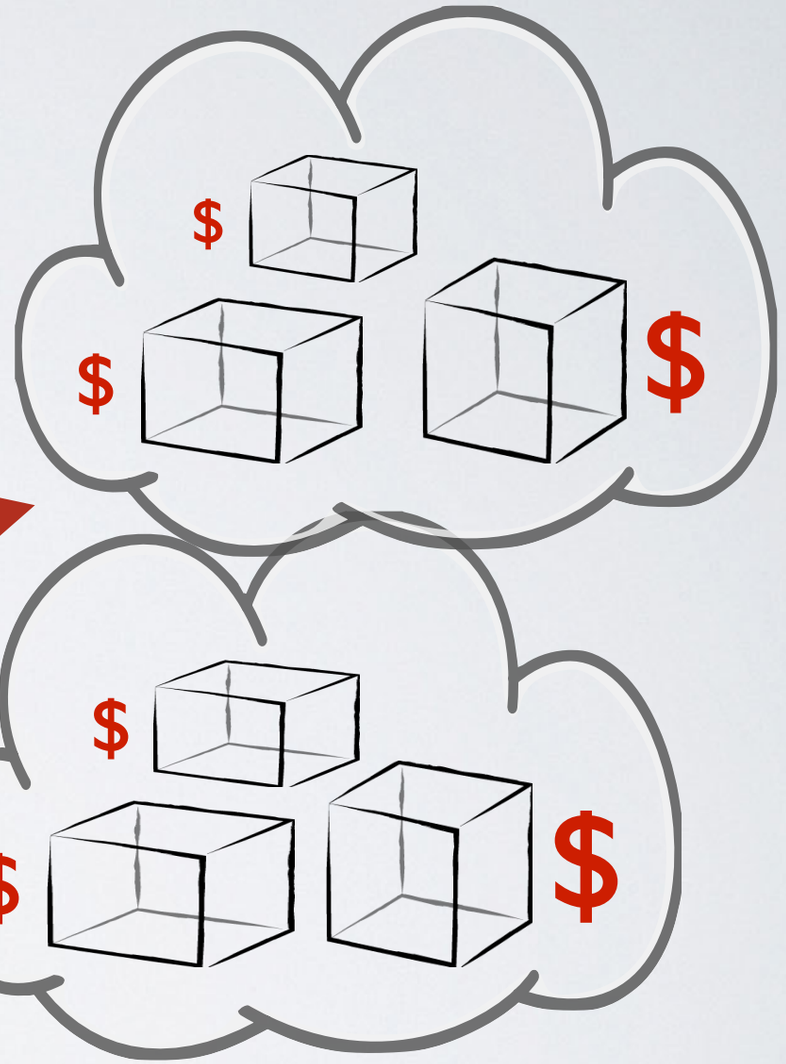
Throughput



CPU USAGE



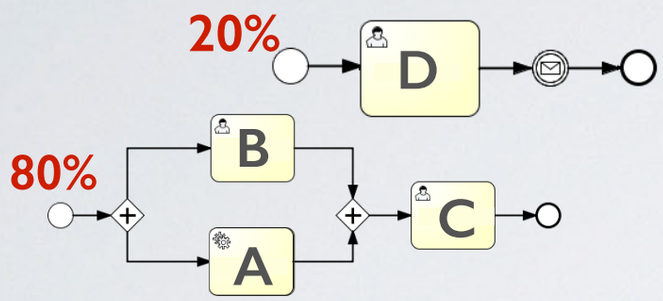
CPU + RAM Bound



Matching Clouds VMs

The Steps of the Method

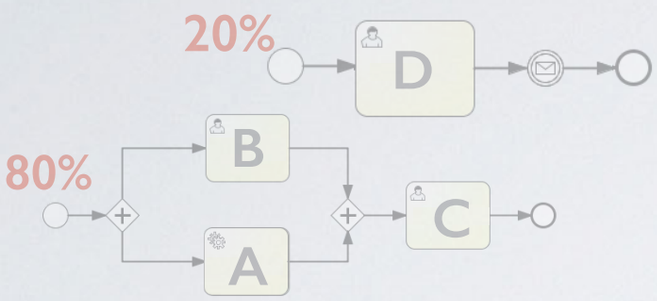
from performance requirements to cheapest VMs



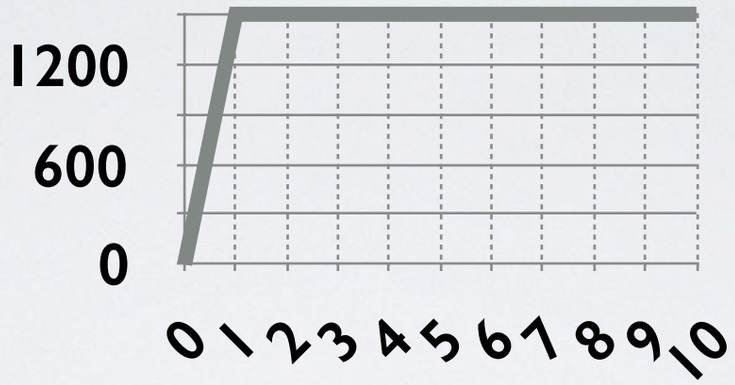
1. Workload Mix

The Steps of the Method

from performance requirements to cheapest VMs



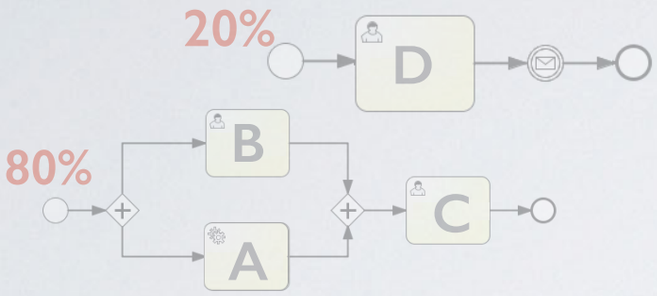
1. Workload Mix



2. Load Function

The Steps of the Method

from performance requirements to cheapest VMs

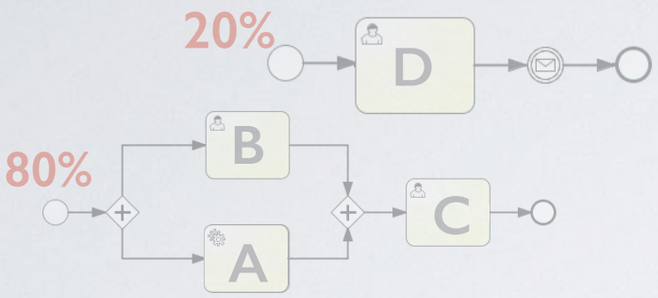


Metrics
KPIs

1. Workload Mix → 2. Load Function → 3. Performance Req.

The Steps of the Method

from performance requirements to cheapest VMs



1. Workload Mix

2. Load Function

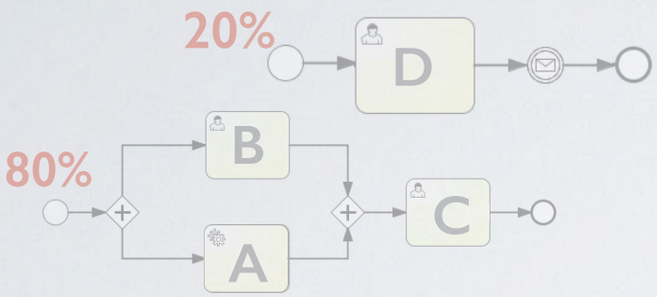
Metrics
KPIs

3. Performance Req.

4. Run Experiments

The Steps of the Method

from performance requirements to cheapest VMs



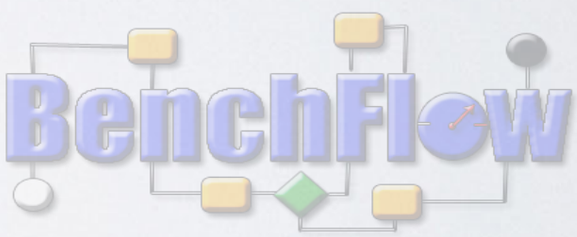
1. Workload Mix

2. Load Function

Metrics
KPIs

3. Performance Req.

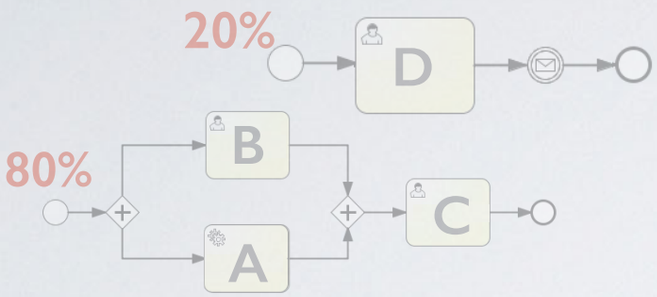
5. Analyse Results



4. Run Experiments

The Steps of the Method

from performance requirements to cheapest VMs

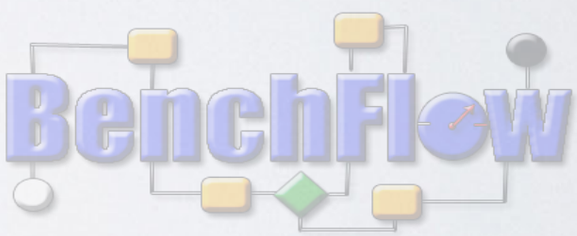


1. Workload Mix

2. Load Function

Metrics
KPIs

3. Performance Req.



4. Run Experiments

5. Analyse Results

6. Cloud Offers

The Steps of the Method

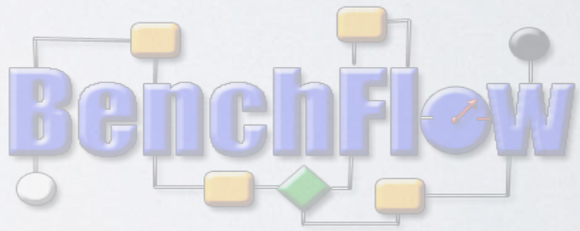
from performance requirements to cheapest VMs

Metrics
KPIs

7. VMs Selection

2. Load Function

3. Performance Req.



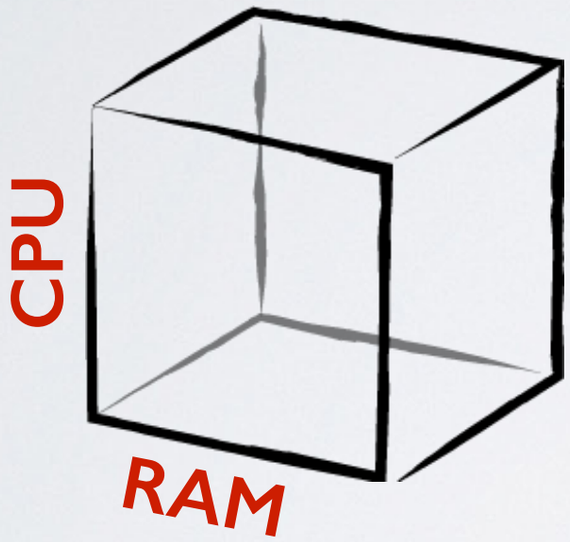
6. Cloud Offers

5. Analyse Results

4. Run Experiments

Assumptions

RAM and CPU, only WfMS no DBMS, workload dependent

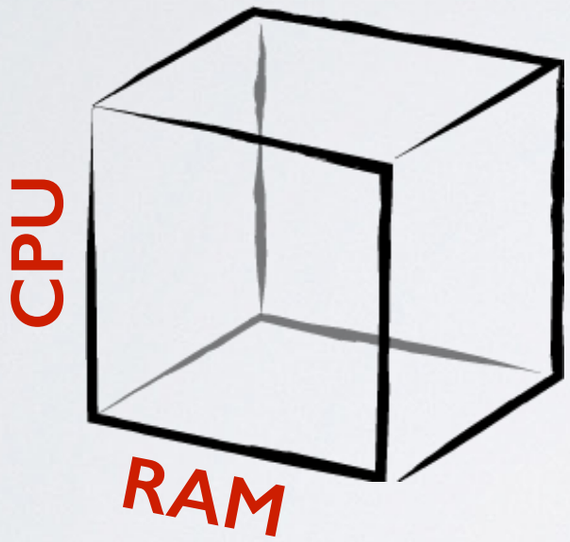


Observed Resources

Assumptions

RAM and CPU, only WfMS no DBMS, workload dependent

No Network, No I/O

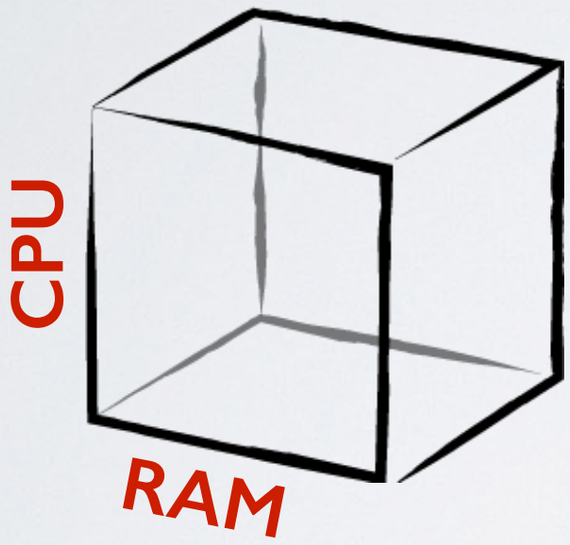


Observed Resources

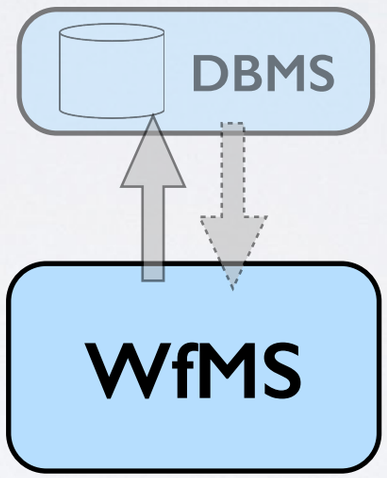
Assumptions

RAM and CPU, only WfMS no DBMS, workload dependent

No Network, No I/O



Observed Resources

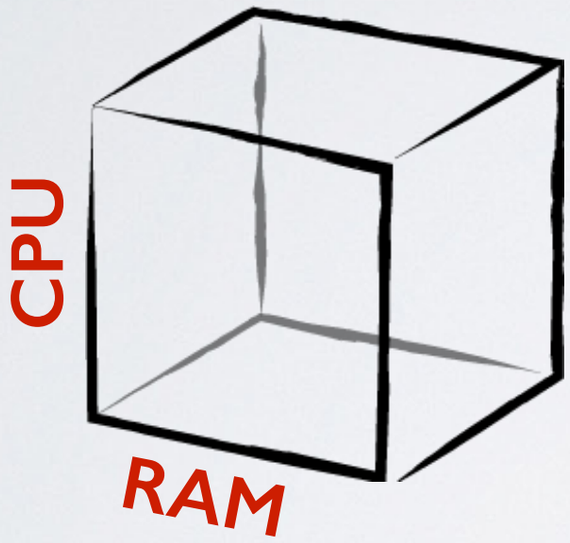


Only WfMS

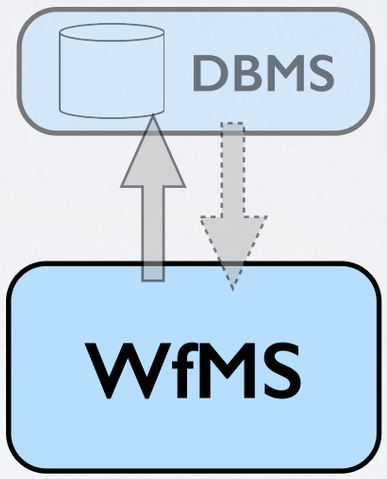
Assumptions

RAM and CPU, only WfMS no DBMS, workload dependent

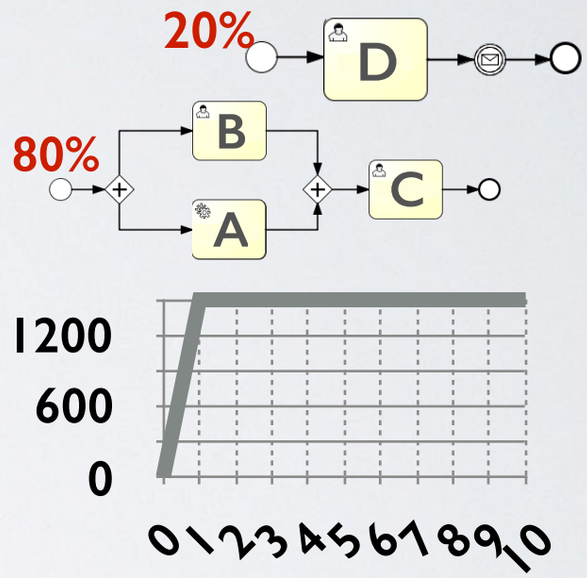
No Network, No I/O



Observed Resources



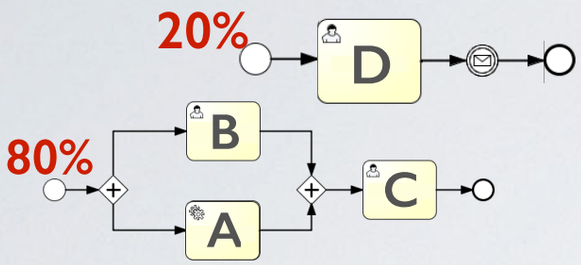
Only WfMS

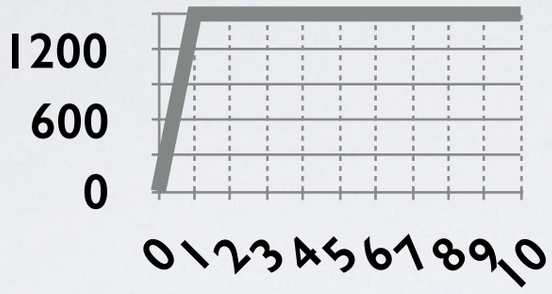


Workload Dependent

An Example of Application

applying the method for the Camunda WfMS





Metrics KPIs

BenchFlow

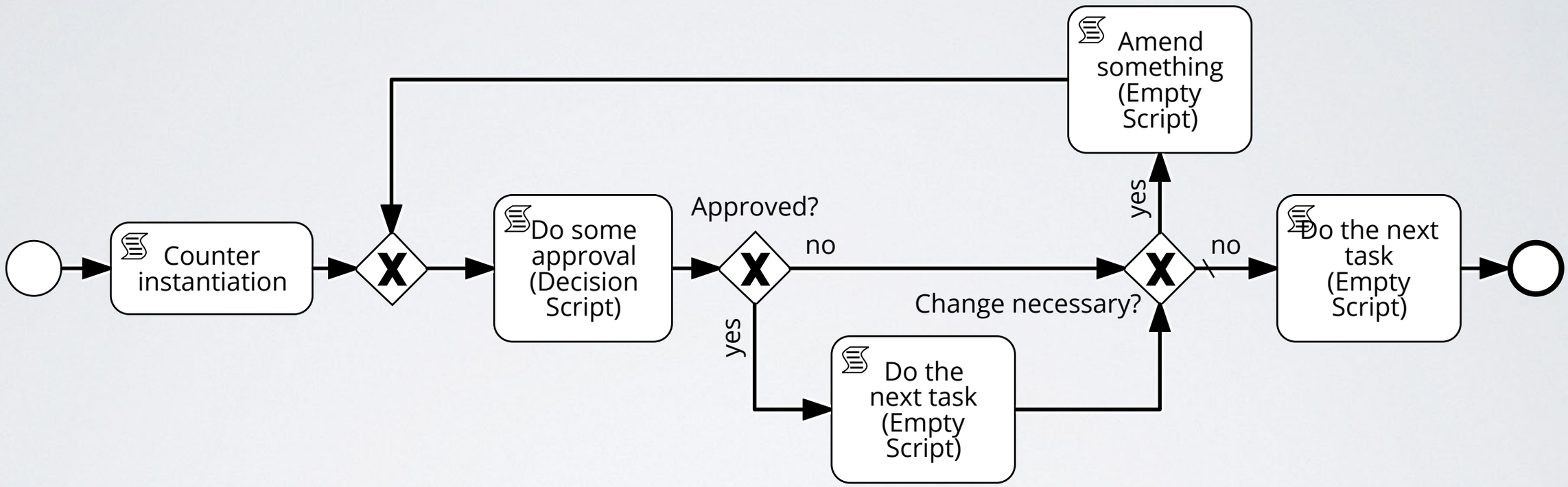
1. → 2. → 3. → 4.

camunda
the business process company

7. ← 6. ← 5.

1. The Workload Mix

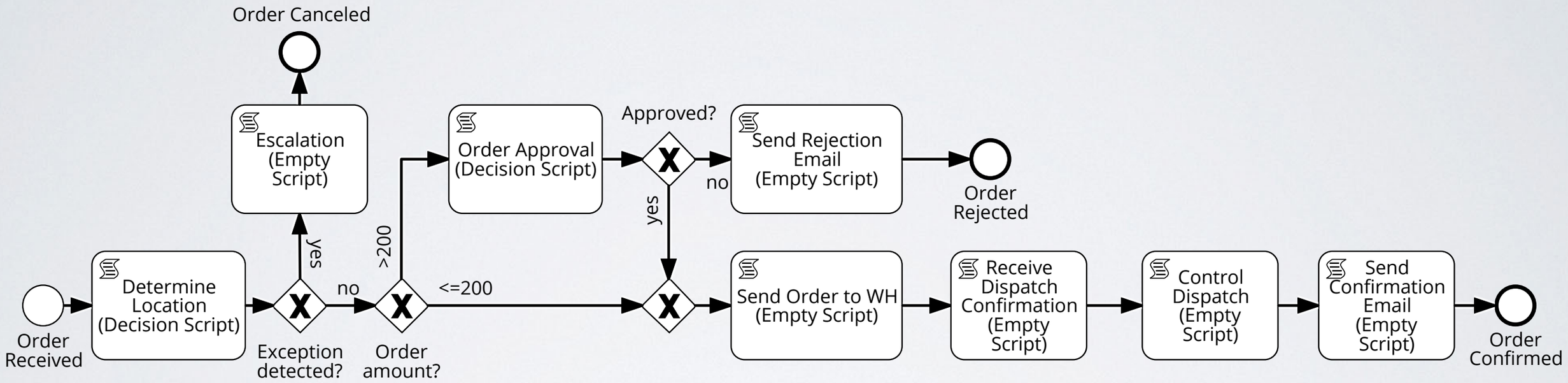
realistic, mix of 5 models with different complexity



21 Elements (Nodes + Edges)

1. The Workload Mix

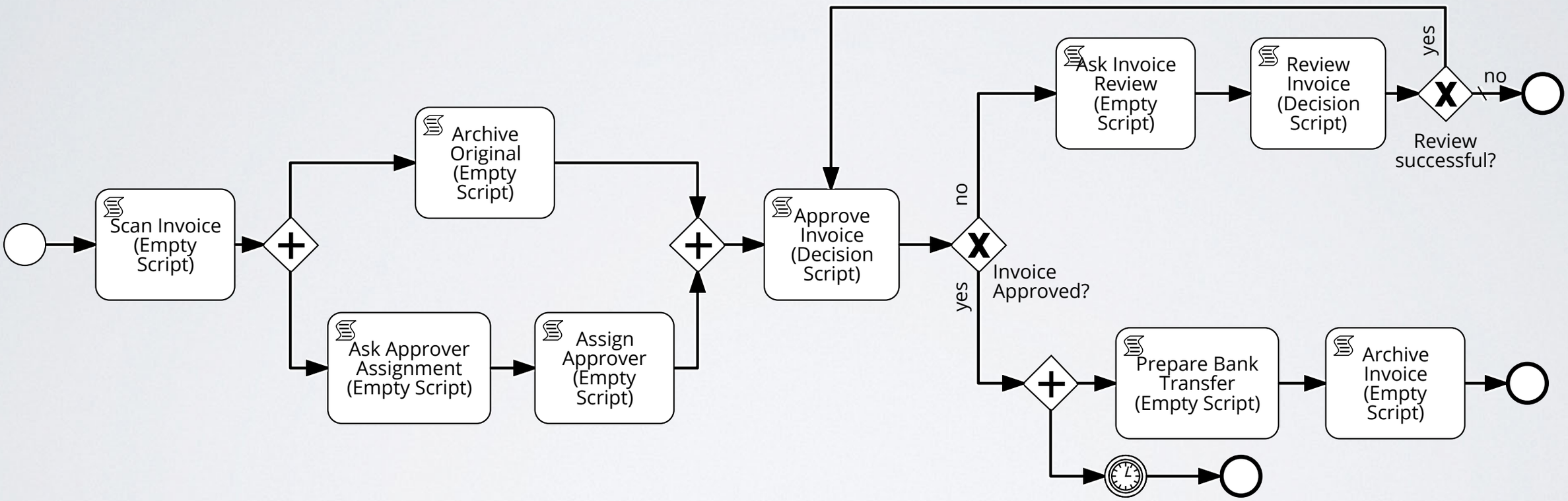
realistic, mix of 5 models with different complexity



32 Elements (Nodes + Edges)

1. The Workload Mix

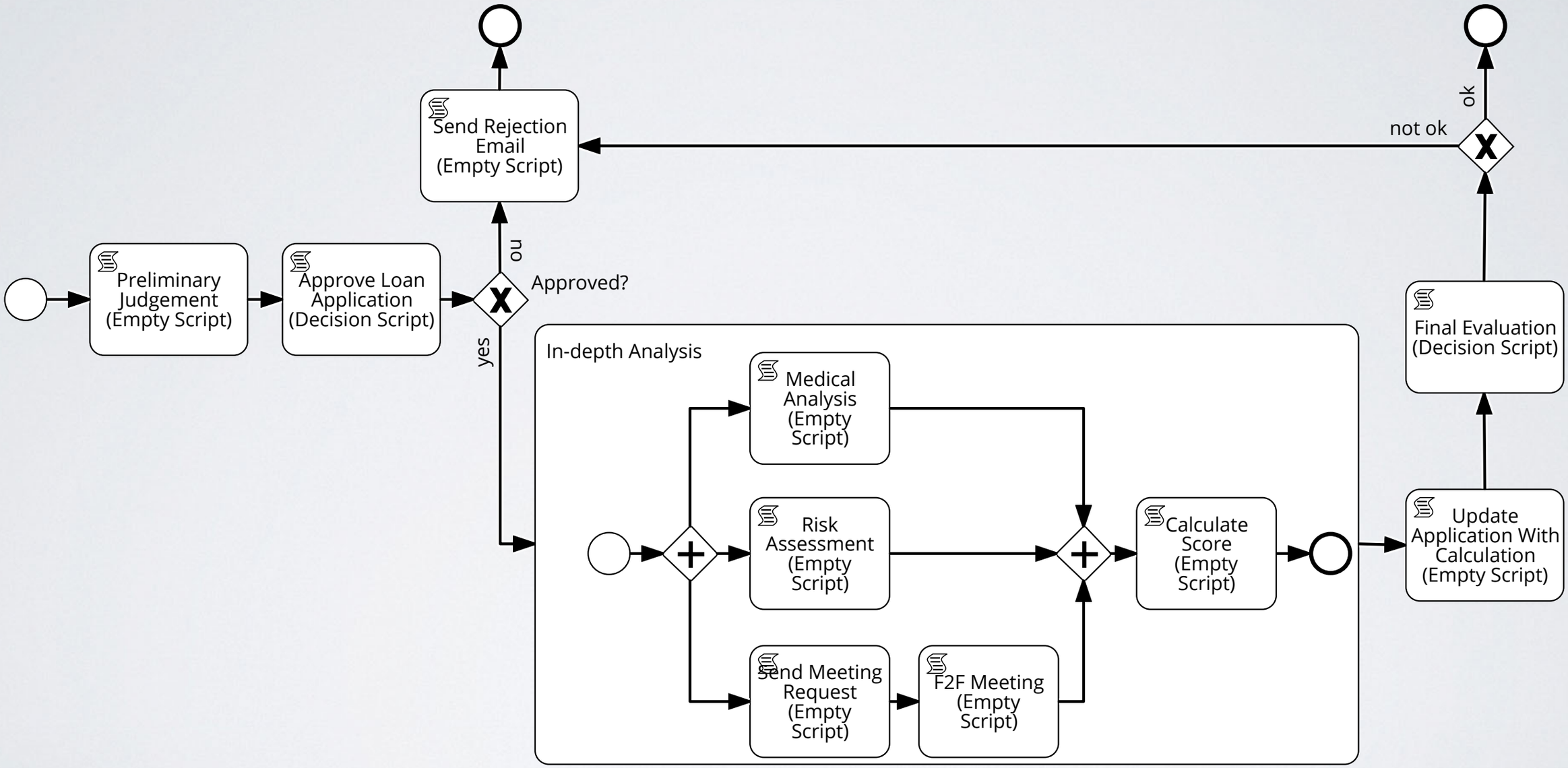
realistic, mix of 5 models with different complexity



39 Elements (Nodes + Edges)

1. The Workload Mix

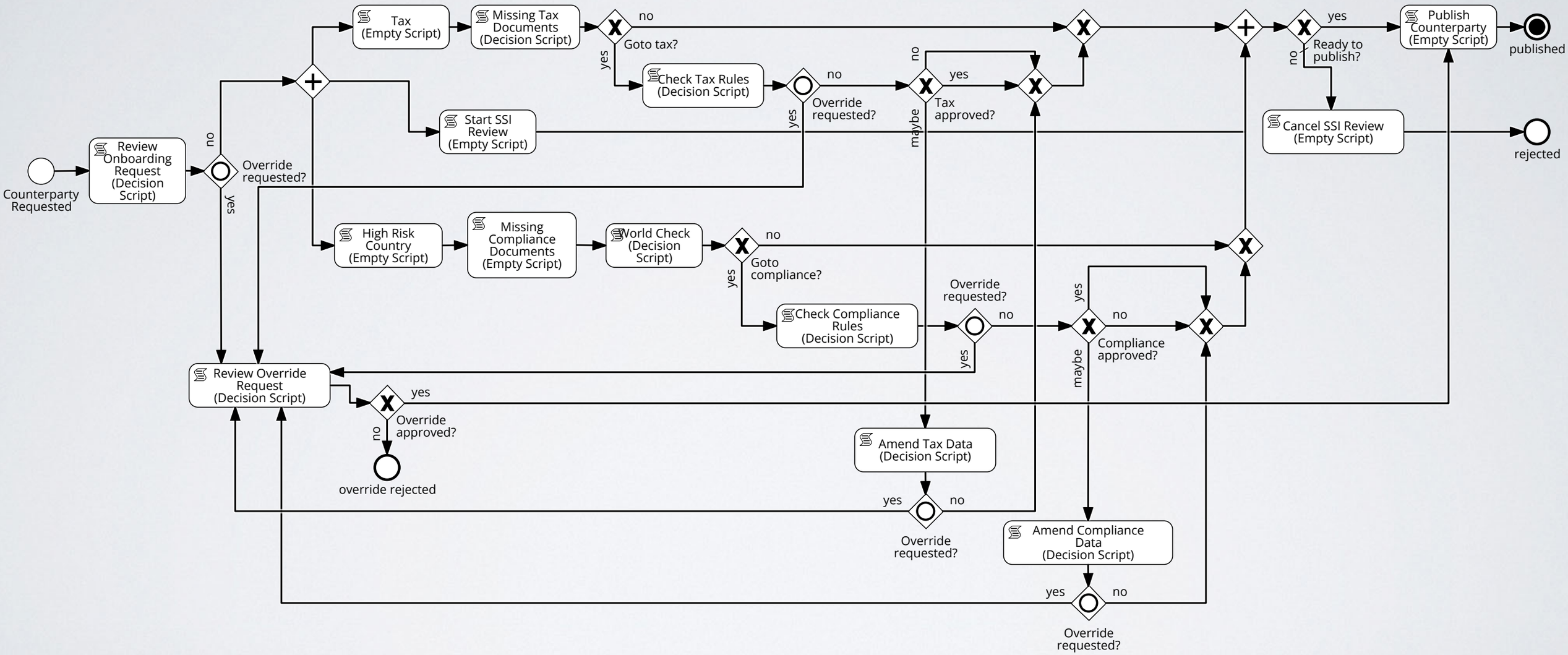
realistic, mix of 5 models with different complexity



41 Elements (Nodes + Edges)

1. The Workload Mix

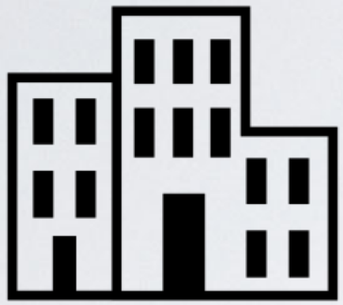
realistic, mix of 5 models with different complexity



84 Elements (Nodes + Edges)

2. The Load Functions

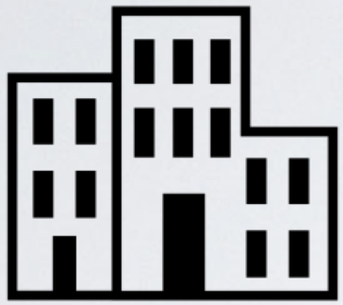
realistic for different sized companies



- 50
- 500 users (*usr*)
- 1000

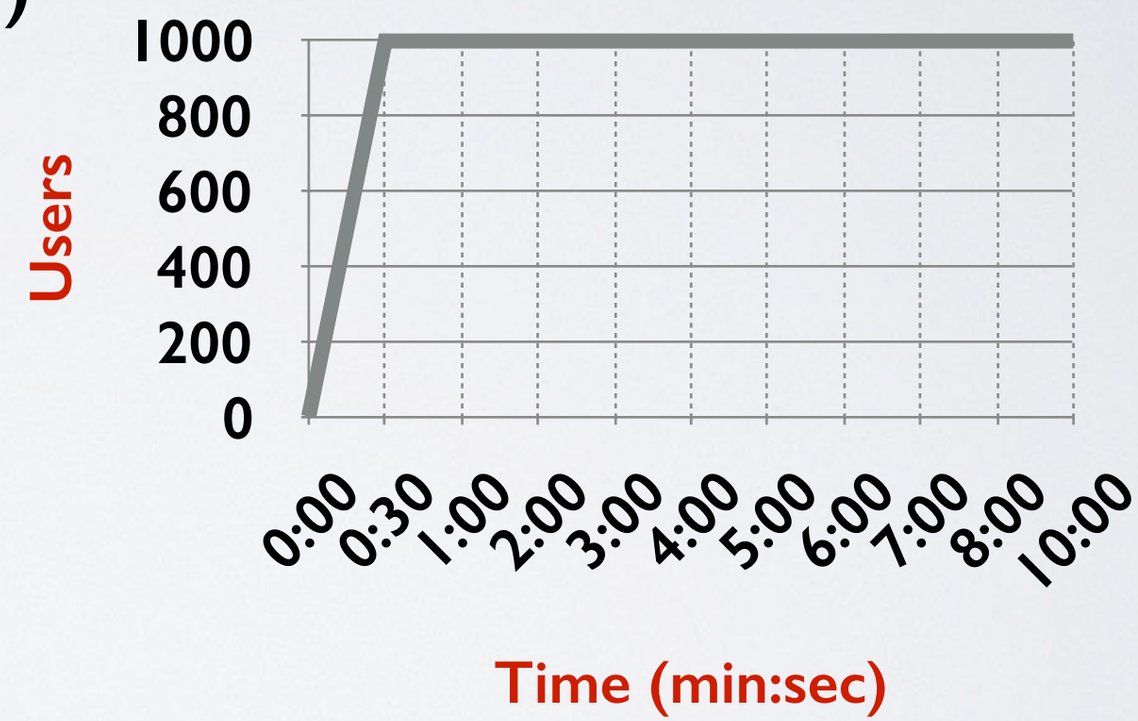
2. The Load Functions

realistic for different sized companies



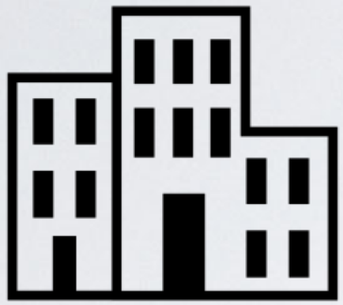
- 50
- 500 users (*usr*)
- 1000

Example for *usr=1000*



2. The Load Functions

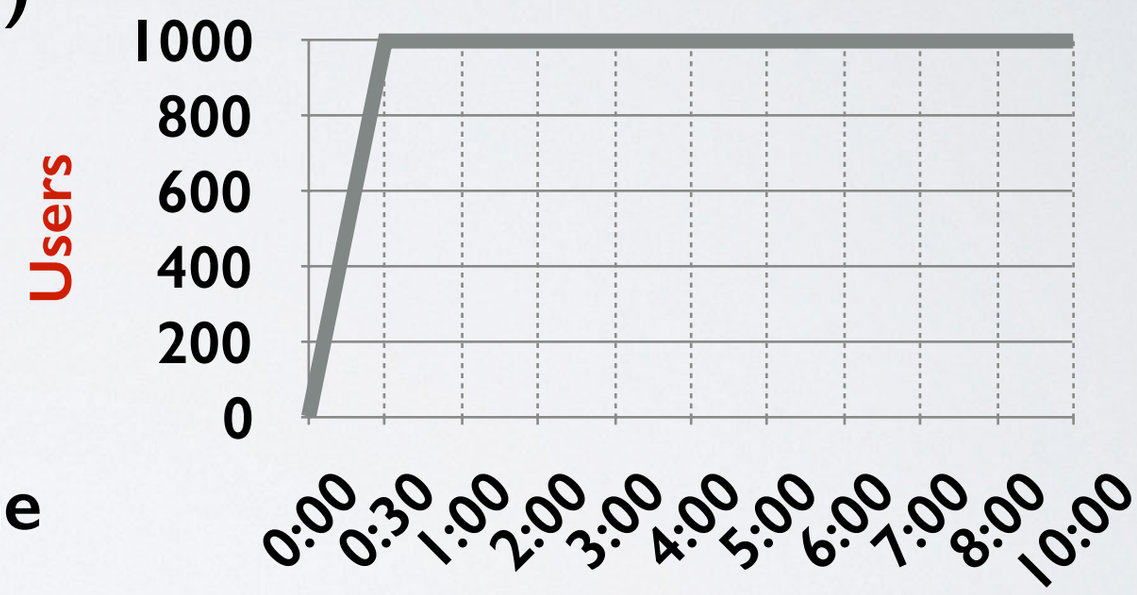
realistic for different sized companies



- 50
- 500 users (*usr*)
- 1000

1 second think time

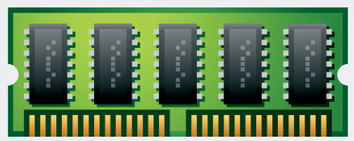
Example for *usr=1000*



3. Performance Requirements

determined with "unbounded" experiments

"Unbounded" *(U)* Configuration:

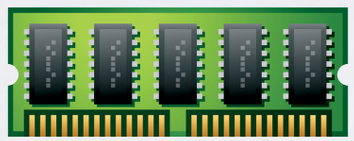
128 GB 
RAM

64 Cores 
CPU

3. Performance Requirements

determined with "unbounded" experiments

"Unbounded" *(U)* Configuration:

128 GB 
RAM

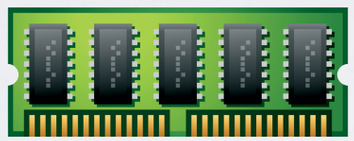
64 Cores 
CPU

Target Performance

3. Performance Requirements

determined with "unbounded" experiments

"Unbounded" *(U)* Configuration:

128 GB  RAM

64 Cores  CPU

Target Performance

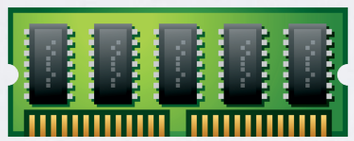
Client-side Metrics:

- Number of Client Requests per second *(REQ/s)*
- Response Time *(RT)*
- …

3. Performance Requirements

determined with "unbounded" experiments

"Unbounded" *(U)* Configuration:

128 GB  RAM

64 Cores  CPU

Target Performance

Client-side Metrics:

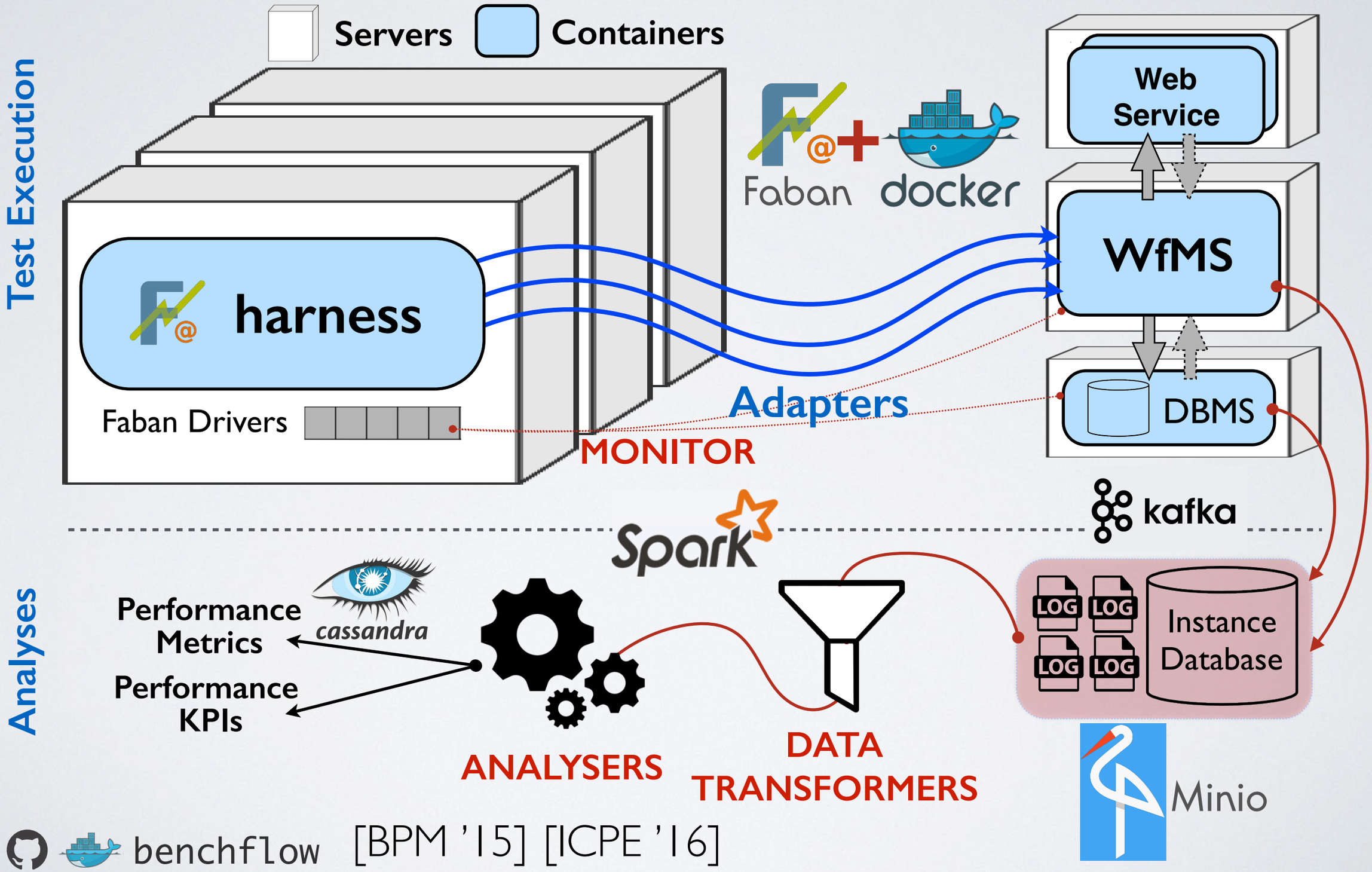
- Number of Client Requests per second *(REQ/s)*
- Response Time *(RT)*
- …

Server-side Metrics:

- Number of executed BP Instances *(N)*
- BP Instance Duration *(D)*
- Throughput *(T)*
- …

4. Run Experiments

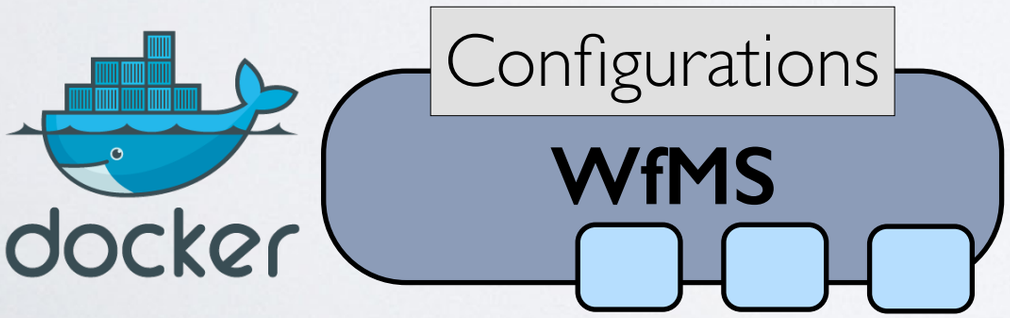
the BenchFlow framework



benchflow [BPM '15] [ICPE '16]

5. Analyse Results

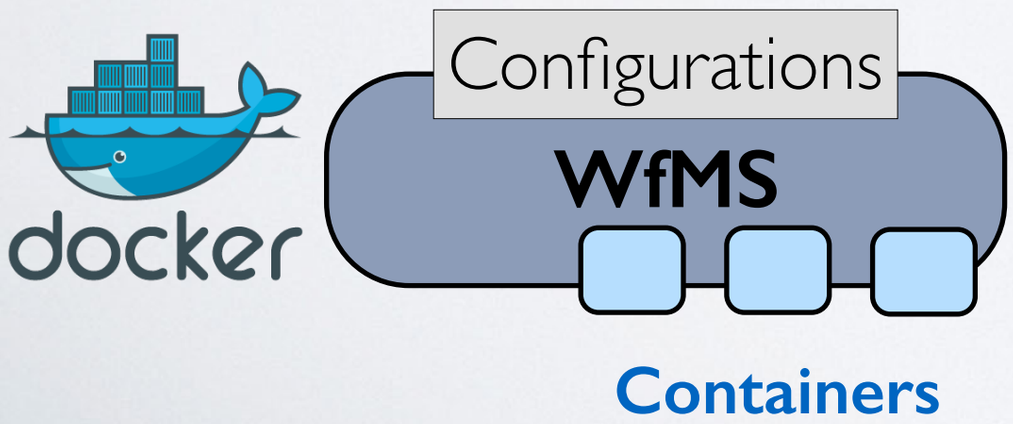
define (determine) the wanted performance



Containers

5. Analyse Results

define (determine) the wanted performance



3 Servers

WfMS, DBMS, Load Generator

10 Gbit/s 10 Gbit/s

5. Analyse Results

define (determine) the wanted performance

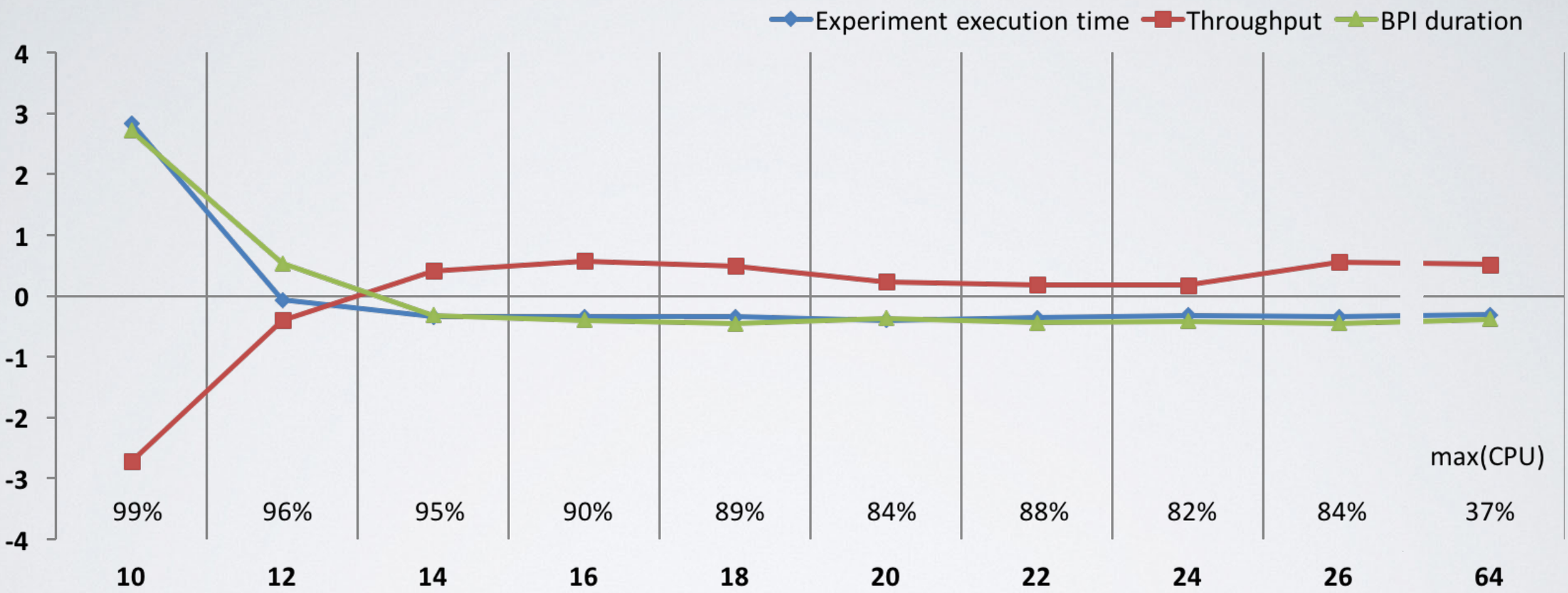
	Users	wavg(D) [ms]	avg(N) [bpi]	avg(T) [bpi/s]	wavg(REQ/s)	wavg(RT) [ms]
U	50	8'238.13	30'623±22	44.45±0.13	48.85	22.97
	500	9'148.13	272'910±6'024	395.90±8.47	434.14	152.18
	1'000	64'023.83	323'783±5'643	329.81±2.73	512.71	946.27

Three Runs to Improve Results Reliability

	Users	WfMS CPU	WfMS RAM	DB CPU	DB RAM
U	50, 500, 1'000	64 Cores	128 GB	64 Cores	128 GB

5. Analyse Results

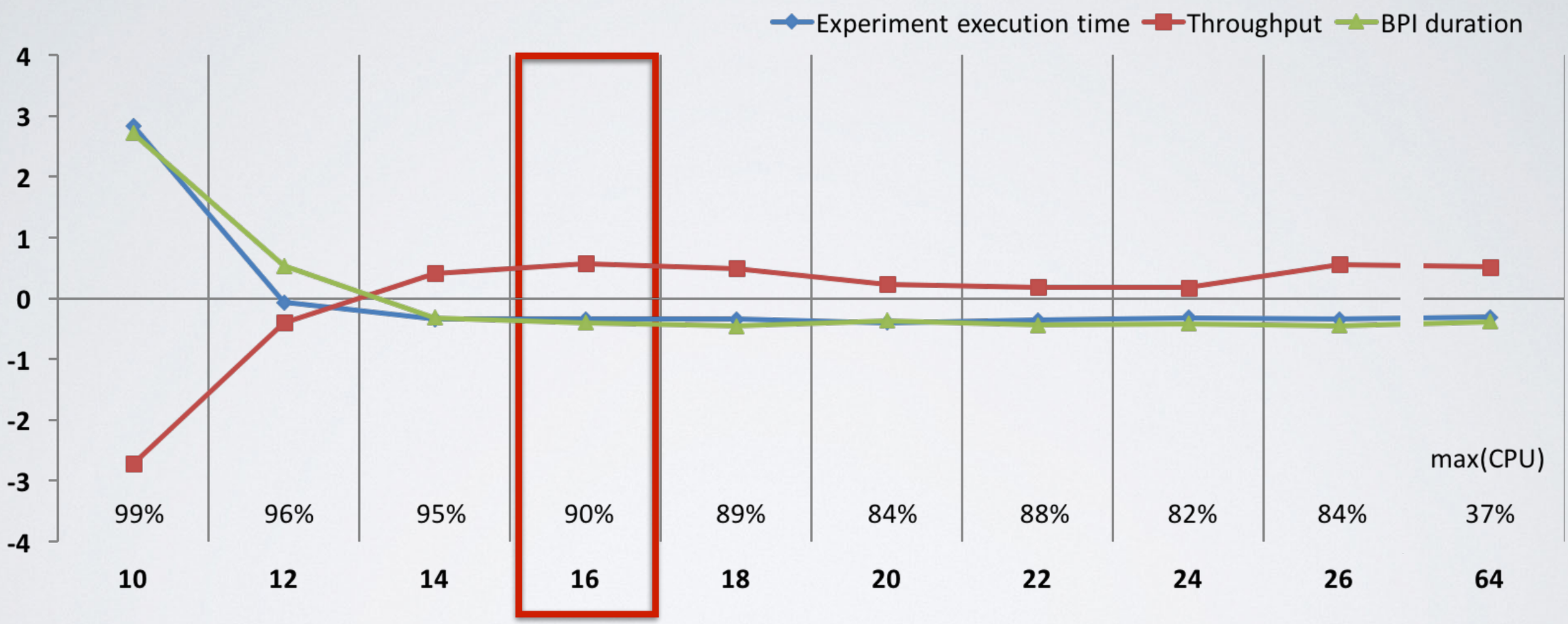
determine the right amount of needed resources



Normalised performance over number of CPU cores with 500 Users

5. Analyse Results

determine the right amount of needed resources



Normalised performance over number of CPU cores with 500 Users

5. Analyse Results

determine the right amount of needed resources

	Users	WfMS CPU	WfMS RAM	DB CPU	DB RAM
B	50	6 Cores	1 GB	6 Cores	2 GB
	500	16 Cores	2 GB	16 Cores	10 GB
	1'000	24 Cores	2 GB	24 Cores	12 GB
U	50, 500, 1'000	64 Cores	128 GB	64 Cores	128 GB

Normalised performance over number of CPU cores with 500 Users

5. Analyse Results

validate the performance results

	Users	wavg(D) [ms]	avg(N) [bpi]	avg(T) [bpi/s]	wavg(REQ/s)	wavg(RT) [ms]
B	50	8'337.41	30'590±43	44.38±0.18	48.84	24.53
	500	9'079.07	273'116±3'063	396.21±4.79	435.27	149.21
	1'000	65'772.55	328'248±2'268	329.80±3.83	519.14	921.57

Unbound vs Bound Performance Results

5. Analyse Results

validate the performance results

	Users	wavg(D) [ms]	avg(N) [bpi]	avg(T) [bpi/s]	wavg(REQ/s)	wavg(RT) [ms]
U	50	8'238.13	30'623±22	44.45±0.13	48.85	22.97
	500	9'148.13	272'910±6'024	395.90±8.47	434.14	152.18
	1'000	64'023.83	323'783±5'643	329.81±2.73	512.71	946.27
B	50	8'337.41	30'590±43	44.38±0.18	48.84	24.53
	500	9'079.07	273'116±3'063	396.21±4.79	435.27	149.21
	1'000	65'772.55	328'248±2'268	329.80±3.83	519.14	921.57

Unbound vs Bound Performance Results

6. Cloud Offers

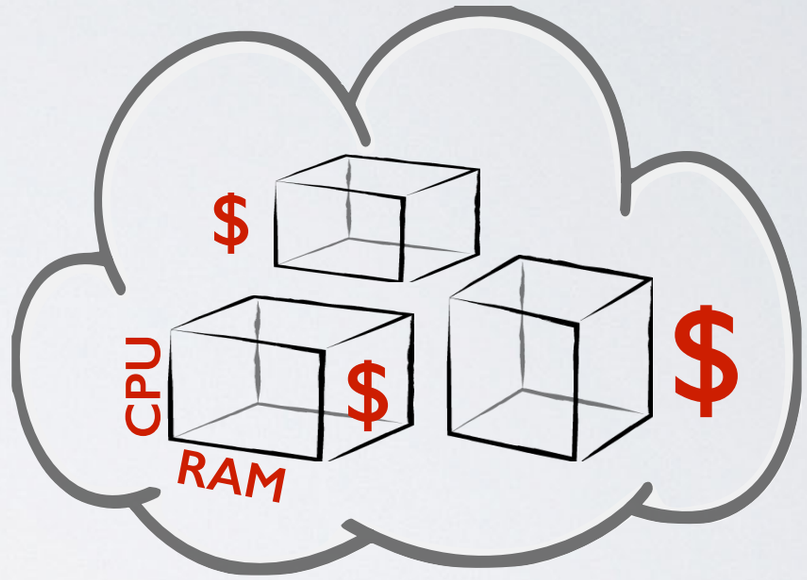
match resource requirements with Cloud offers

Microsoft Azure

Google Cloud Platform

amazon web services

Springs.io



CPU + RAM Bound

	Users	WfMS CPU	WfMS RAM
B	50	6 Cores	1 GB
	500	16 Cores	2 GB
	1'000	24 Cores	2 GB

6. Cloud Offers

match resource requirements with Cloud offers

	Cloud Provider	Instance type	CPU	Memory (GB)	Price (USD/hr)
50 Users	**Required resources**	-	**6 Cores**	**1**	-
	Amazon (Am)	Compute Optimised - c4.2xlarge	8 Cores	15	0.419
	Azure (Az)	General purpose - basic tier - A4	8 Cores	14	0.376
	Google Predefined (Gp)	High-CPU - n1-highcpu-8	8 Cores	7.2	0.232
	Google Custom (Gc)	-	6 Cores	5.4	0.25827
	Springs.io (S)	-	12 GHz	1	0.107

6. Cloud Offers

match resource requirements with Cloud offers

	Cloud Provider	Instance type	CPU	Memory (GB)	Price (USD/hr)
500 Users	**Required resources**	-	**16 Cores**	**2**	-
	Amazon (Am)	Compute Optimised - c4.4xlarge	16 Cores	30	0.838
	Azure (Az)	Compute Optimised - D5 v2	16 Cores	56	1.17
	Google Predefined (Gp)	High-CPU - n1-highcpu-16	16 Cores	14.4	0.464
	Google Custom (Gc)	-	16 Cores	14.4	0.68872
	Springs.io (S)	-	-	-	-

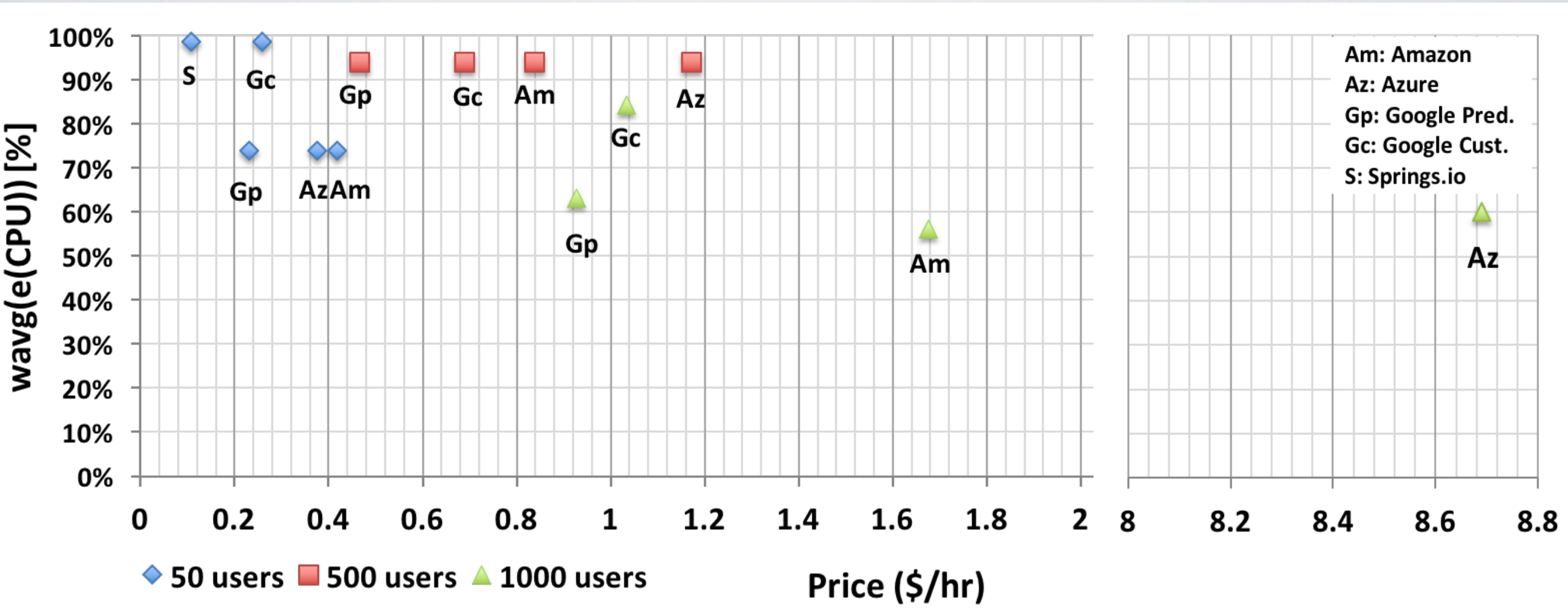
6. Cloud Offers

match resource requirements with Cloud offers

	Cloud Provider	Instance type	CPU	Memory (GB)	Price (USD/hr)
1000 Users	**Required resources**	-	**24 Cores**	**2**	-
	Amazon (Am)	Compute Optimised - c4.8xlarge	36 Cores	60	1.675
	Azure (Az)	Performance optimized compute - G5	32 Cores	448	8.69
	Google Predefined (Gp)	High-CPU - n1-highcpu-32	32 Cores	28.8	0.928
	Google Custom (Gc)	-	24 Cores	21.6	1.03308
	Springs.io (S)	-	-	-	-

7. VMs Selection

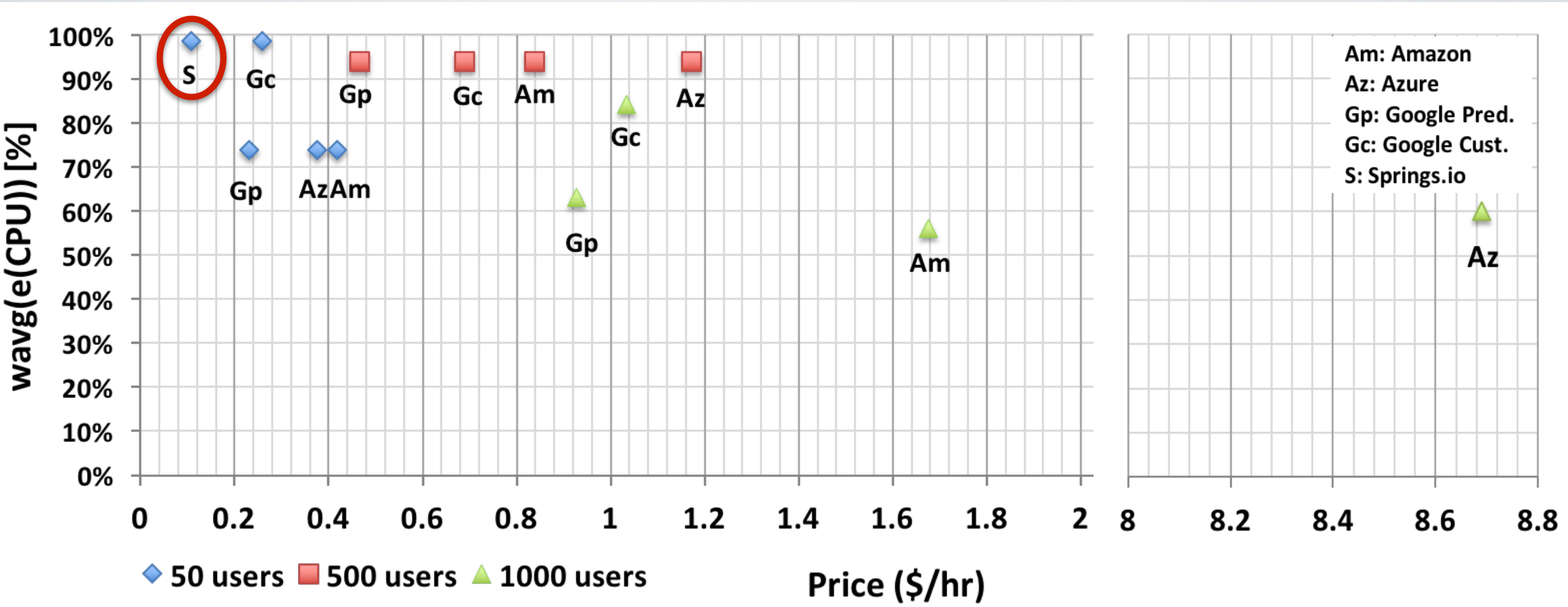
select the set of suitable VMs, and identify the cheapest ones



Normalised Cores Utilisation vs Price of VMs

7. VMs Selection

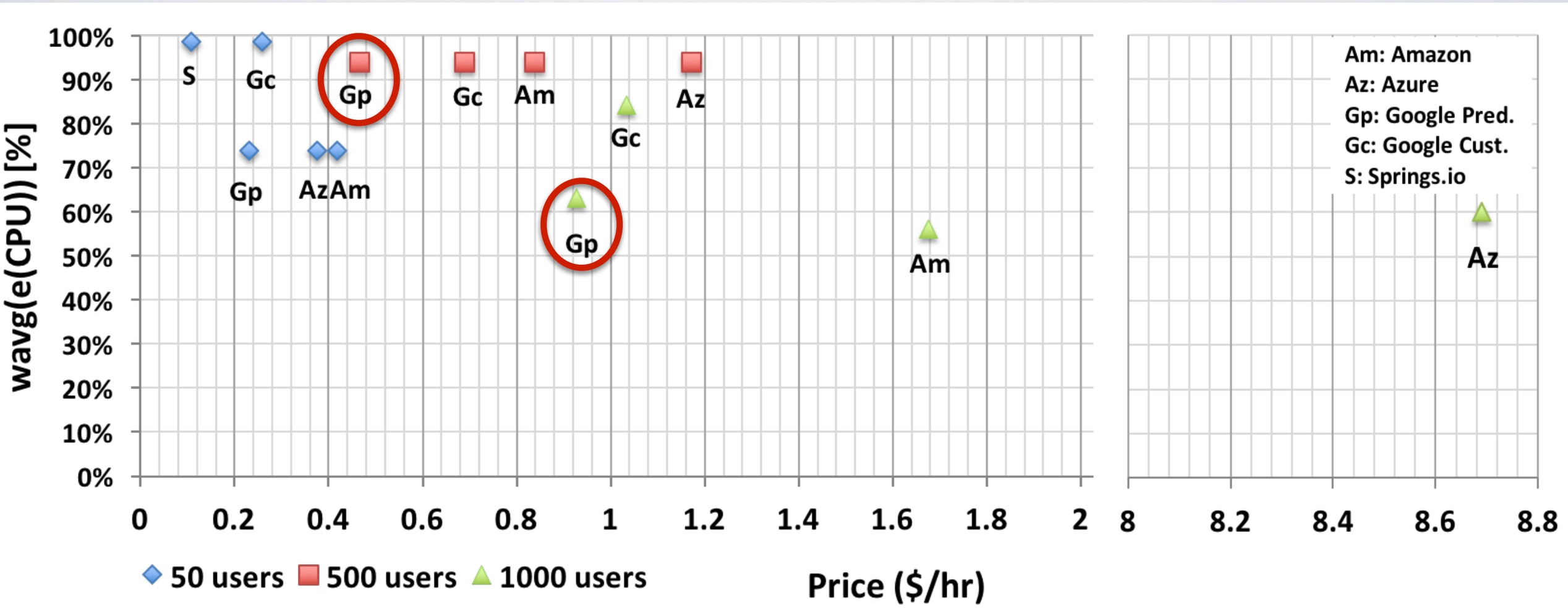
select the set of suitable VMs, and identify the cheapest ones



Normalised Cores Utilisation vs Price of VMs

7. VMs Selection

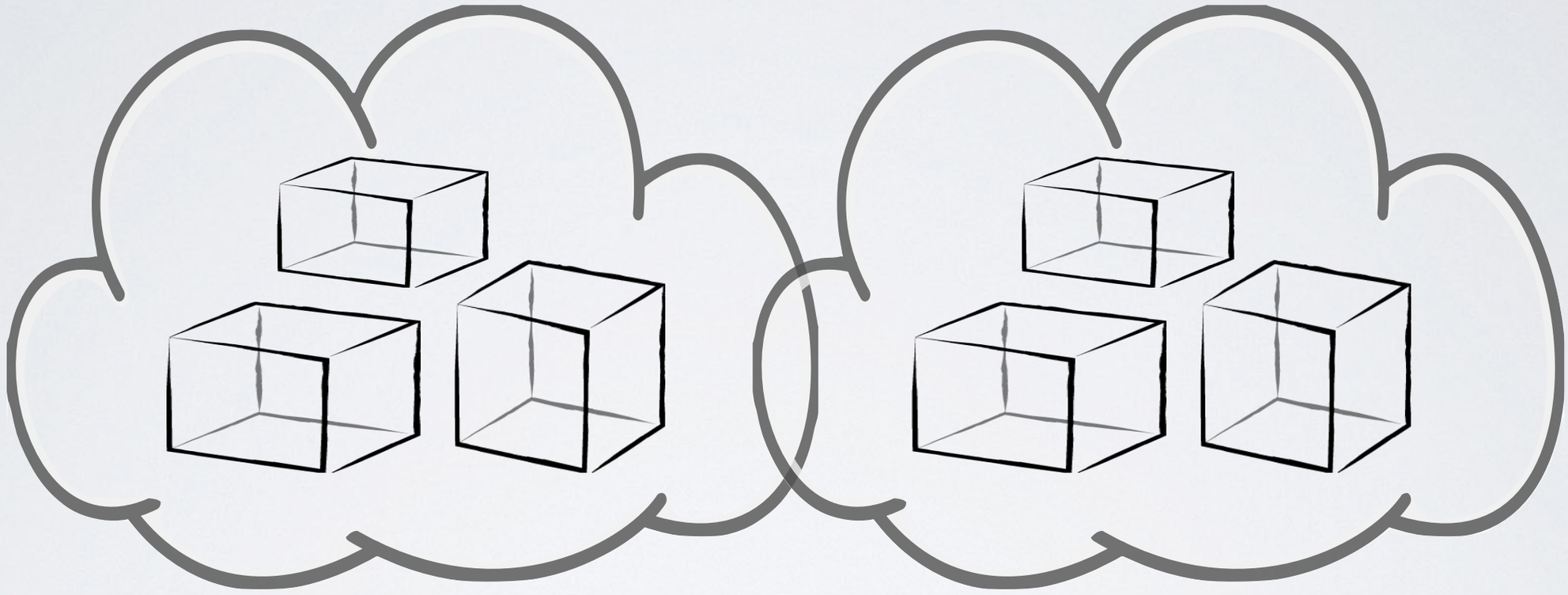
select the set of suitable VMs, and identify the cheapest ones



Normalised Cores Utilisation vs Price of VMs

Limitations and Future Work

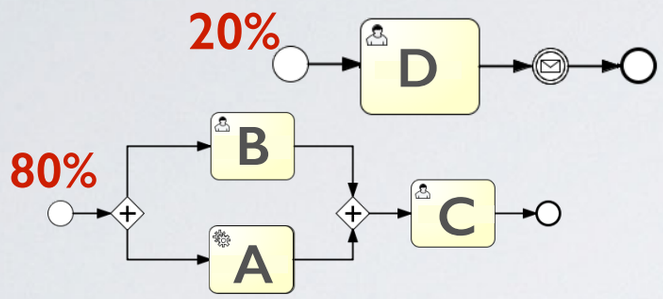
approximation of the actual performance on the Cloud



Performance on the Cloud has more Variance

Limitations and Future Work

measure on the Cloud, validate the approach

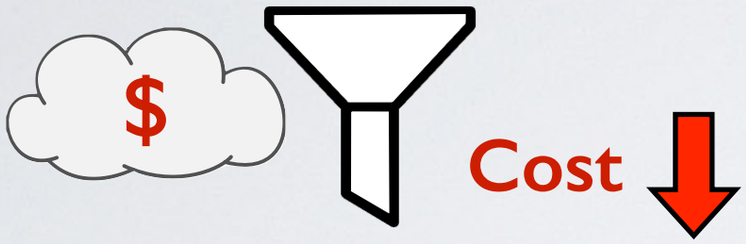


1. Workload Mix

Limitations and Future Work

measure on the Cloud, validate the approach

Res. Efficiency



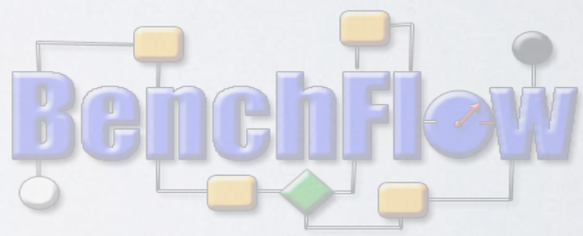
Cost

Metrics
KPIs

7. VMs Selection

2. Load Function

3. Performance Req.



6. Cloud Offers

5. Analyse Results

4. Run Experiments

Limitations and Future Work

measure on the Cloud, validate the approach

Res. Efficiency

$

Cost

Metrics
KPIs

7. VMs Selection → 8. Cloud Exp. → 3. Performance Req.

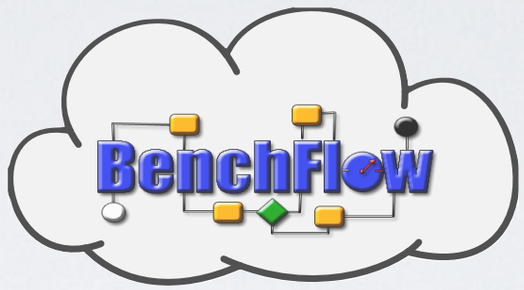
$

Performance
Resources

6. Cloud Offers ← 5. Analyse Results ← 4. Run Experiments

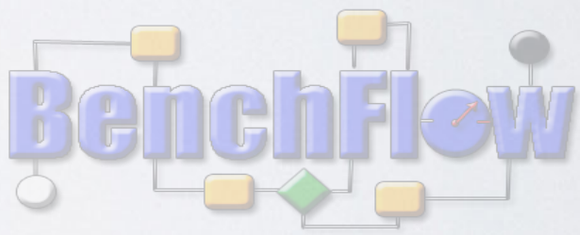
Limitations and Future Work

measure on the Cloud, validate the approach



Metrics
KPIs

7. VMs Selection → 8. Cloud Exp. → 3. Performance Req.

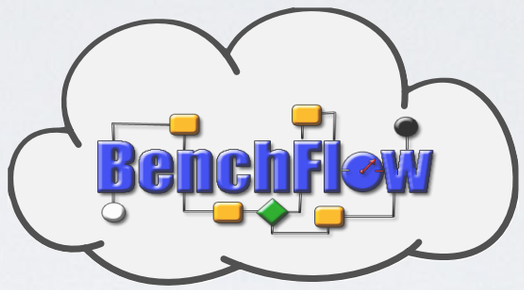


6. Cloud Offers ← 5. Analyse Results ← 4. Run Experiments

Add Network, I/O

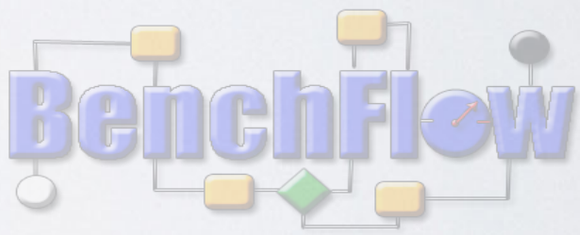
Limitations and Future Work

measure on the Cloud, validate the approach



Metrics
KPIs

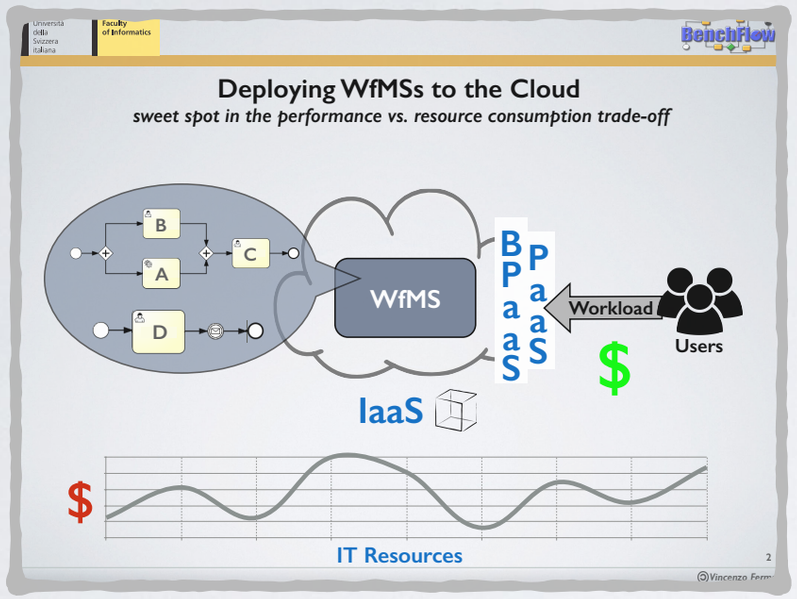
7. VMs Selection → 8. Cloud Exp. → 3. Performance Req.



6. Cloud Offers ← 5. Analyse Results ← 4. Run Experiments

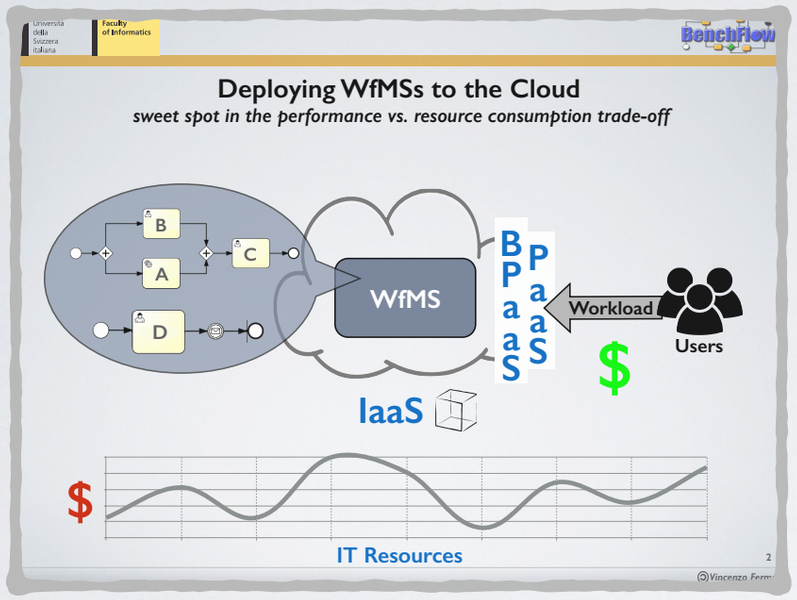
Consider DBMS — **Add Network, I/O**

Highlights

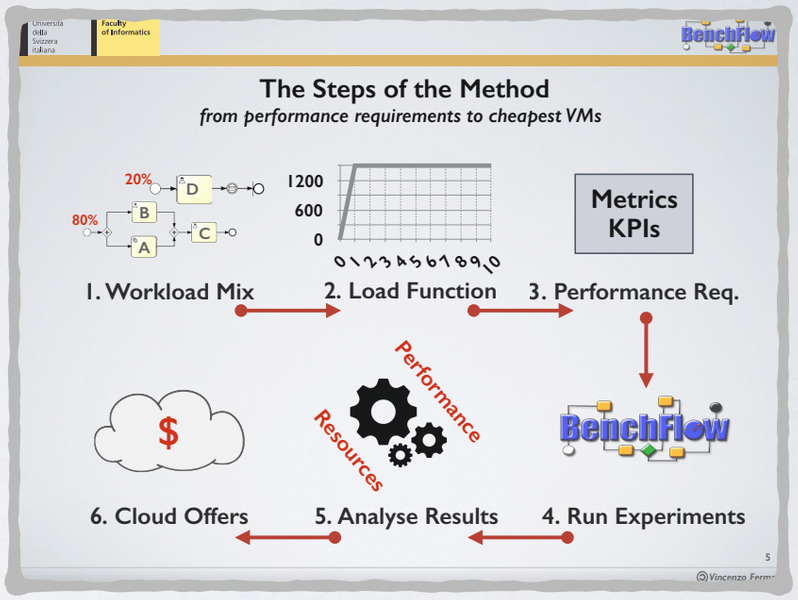


Deploying WfMSs to the Cloud

Highlights

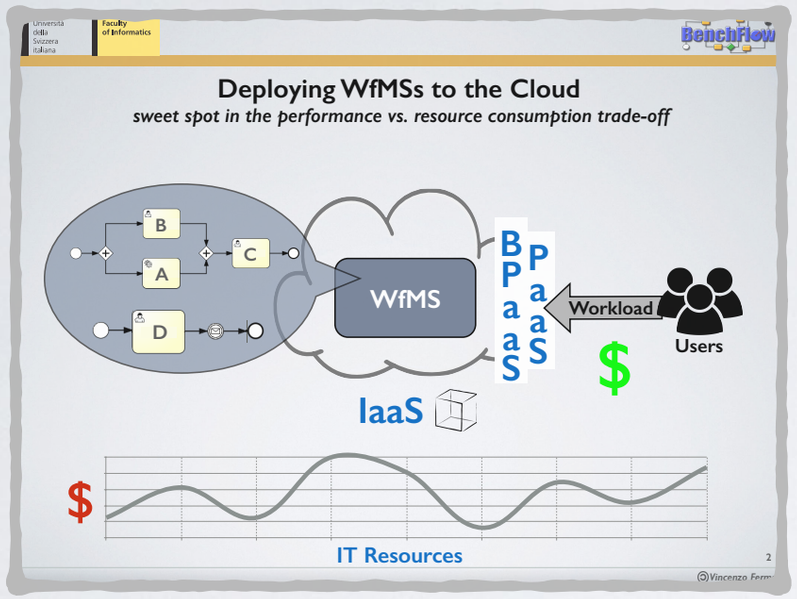


Deploying WfMSs to the Cloud

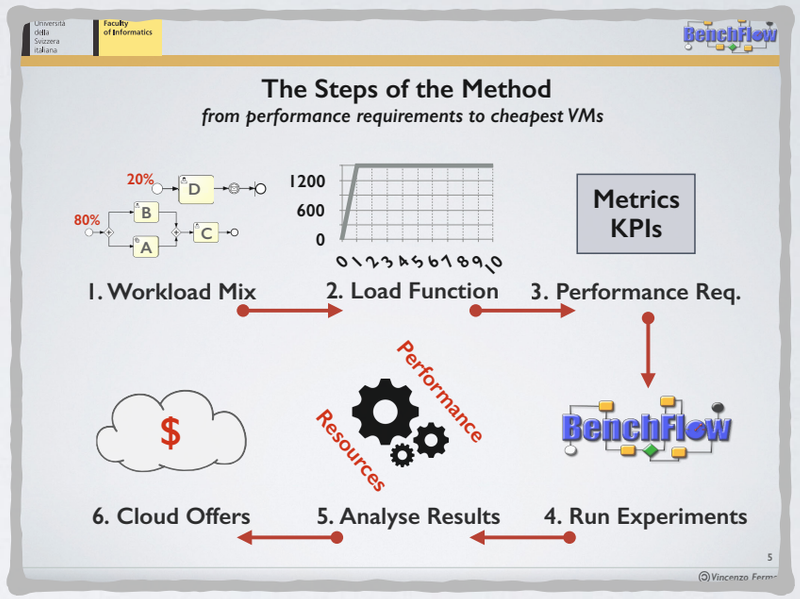


Proposed Method

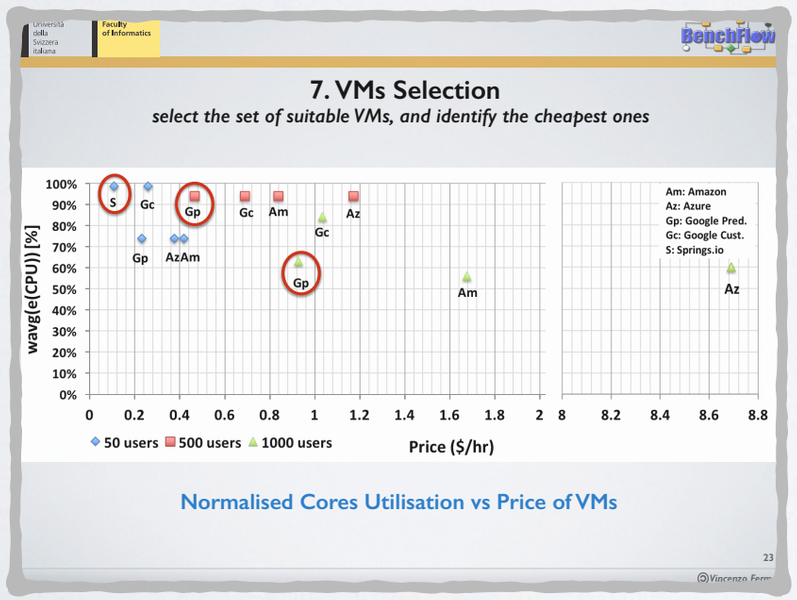
Highlights



Deploying WfMSs to the Cloud



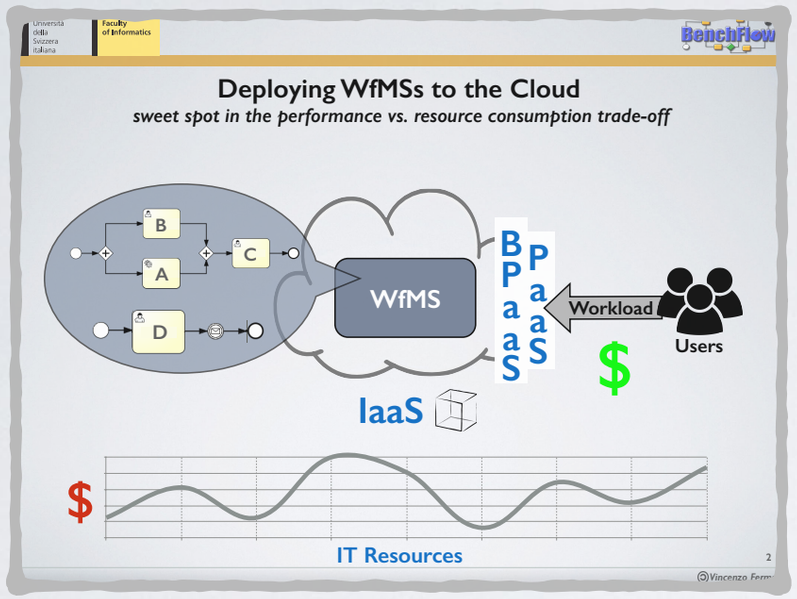
Proposed Method



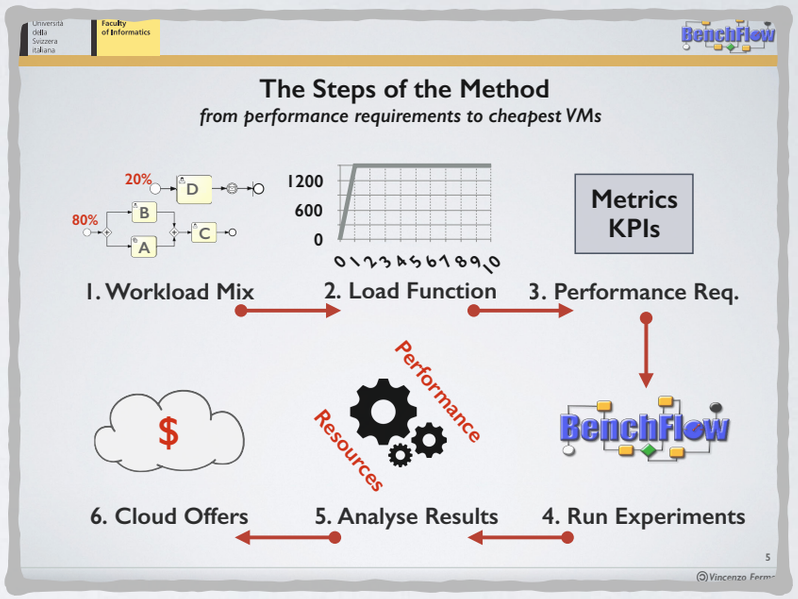
Normalised Cores Utilisation vs Price of VMs

Application to Camunda

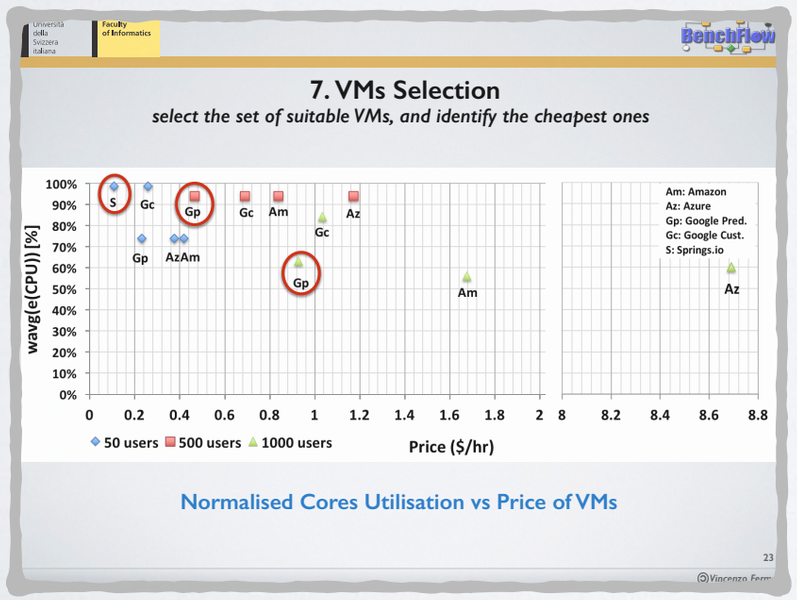
Highlights



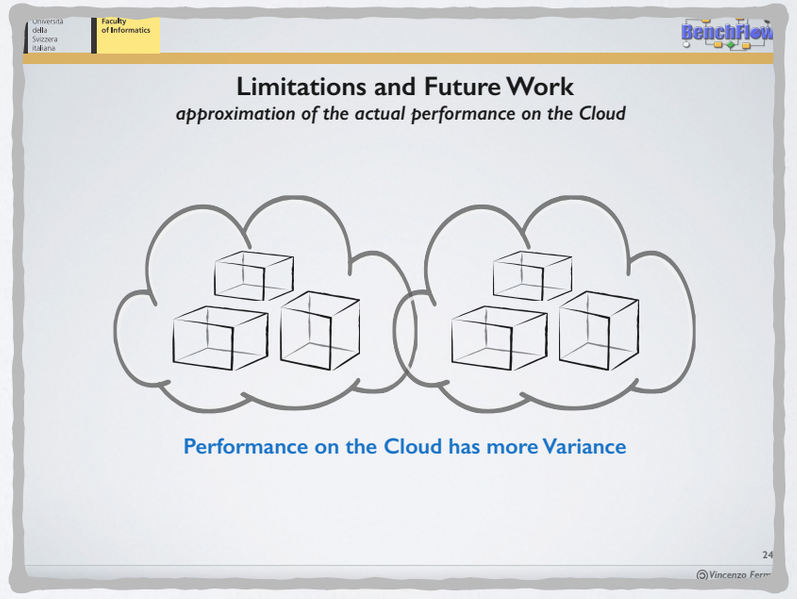
Deploying WfMSs to the Cloud



Proposed Method



Application to Camunda



Limitations and Future Work

Università della Svizzera italiana

Faculty of Informatics

ESTIMATING THE COST FOR EXECUTING BUSINESS PROCESSES IN THE CLOUD

benchflow

benchflow

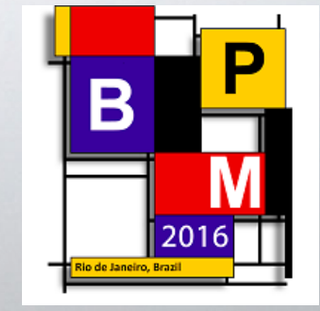
http://benchflow.inf.usi.ch

✉ vincenzo.ferme@usi.ch

Vincenzo Ferme, Ana Ivanchikj, Cesare Pautasso

Faculty of Informatics

USI Lugano, Switzerland



BACKUP SLIDES

Vincenzo Ferme, Ana Ivanchikj, Cesare Pautasso
Faculty of Informatics
USI Lugano, Switzerland

BPM 2016 Rio de Janeiro, Brazil

Published Work

[SSP '14]
M. Skouradaki, D. H. Roller, F. Leymann, V. Ferme, and C. Pautasso. **Technical open challenges on benchmarking workflow management systems**. In Proc. of the 2014 Symposium on Software Performance, SSP 2014, pages 105–112, 2014.

[BTW '15]
C. Pautasso, V. Ferme, D. Roller, F. Leymann, and M. Skouradaki. **Towards workflow benchmarking: Open research challenges**. In Proc. of the 16th conference on Database Systems for Business, Technology, and Web, BTW 2015, pages 331–350, 2015.

[ICPE '15]
M. Skouradaki, D. H. Roller, L. Frank, V. Ferme, and C. Pautasso. **On the Road to Benchmarking BPMN 2.0 Workflow Engines**. In Proc. of the 6th ACM/SPEC International Conference on Performance Engineering, ICPE '15, pages 301–304, 2015.

Published Work

[CLOSER '15]
M. Skouradaki, V. Ferme, F. Leymann, C. Pautasso, and D. H. Roller. **"BPELanon": Protect business processes on the cloud.** In Proc. of the 5th International Conference on Cloud Computing and Service Science, CLOSER 2015. SciTePress, 2015.

[SOSE '15]
M. Skouradaki, K. Goerlach, M. Hahn, and F. Leymann. **Application of Sub-Graph Isomorphism to Extract Reoccurring Structures from BPMN 2.0 Process Models**. In Proc. of the 9th International IEEE Symposium on Service-Oriented System Engineering, SOSE 2015, 2015.

[BPM '15]
V. Ferme, A. Ivanchikj, C. Pautasso. **A Framework for Benchmarking BPMN 2.0 Workflow Management Systems**. In Proc. of the 13th International Conference on Business Process Management, BPM '15, pages 251-259, 2015.

Published Work

[BPMD '15]
A. Ivanchikj, V. Ferme, C. Pautasso. **BPMeter: Web Service and Application for Static Analysis of BPMN 2.0 Collections**. In Proc. of the 13th International Conference on Business Process Management [Demo], BPM '15, pages 30-34, 2015.

[ICPE '16]
V. Ferme, and C. Pautasso. **Integrating Faban with Docker for Performance Benchmarking**. In Proc. of the 7th ACM/SPEC International Conference on Performance Engineering, ICPE '16, 2016.

[CLOSER '16]
V. Ferme, A. Ivanchikj, C. Pautasso., M. Skouradaki, F. Leymann. **A Container-centric Methodology for Benchmarking Workflow Management Systems**. In Proc. of the 6th International Conference on Cloud Computing and Service Science, CLOSER 2016. SciTePress, 2016.

Published Work

[ICWE '16]
C. Jürgen, V. Ferme, H.C. Gall. **Using Docker Containers to Improve Reproducibility in Software and Web Engineering Research**. In Proc. of the 16th International Conference on Web Engineering, 2016.

[CAiSE '16]
M. Skouradaki, V. Ferme, C. Pautasso, F. Leymann, A. van Hoorn. **Micro-Benchmarking BPMN 2.0 Workflow Management Systems with Workflow Patterns** . In Proc. of the 28th International Conference on Advanced Information Systems Engineering, CAiSE '16, 2016.

[ICWS '16]
M. Skouradaki, V. Andrikopoulos, F. Leymann. **Representative BPMN 2.0 Process Model Generation from Recurring Structures**. In Proc. of the 23rd IEEE International Conference on Web Services, ICWS '16, 2016.

Published Work

[SummerSOC '16]
M. Skouradaki, T. Azad, U. Breitenbücher, O. Kopp, F. Leymann. **A Decision Support System for the Performance Benchmarking of Workflow Management Systems**. In Proc. of the 10th Symposium and Summer School On Service-Oriented Computing, SummerSOC '16, 2016.

[BPM Forum '16]
V. Ferme, A. Ivanchikj, C. Pautasso. **Estimating the Cost for Executing Business Processes in the Cloud**. In Proc. of the 14th International Conference on Business Process Management, BPM Forum '16, 2016.

[OTM '16]
M. Skouradaki, V. Andrikopoulos, O. Kopp, F. Leymann. **RoSE: Reoccurring Structures Detection in BPMN 2.0 Process Models Collections.** In Proc. of On the Move to Meaningful Internet Systems Conference, OTM '16, 2016. (to appear)

Docker Performance

[IBM '14]
W. Felter, A. Ferreira, R. Rajamony, and J. Rubio. **An updated performance comparison of virtual machines and Linux containers.** IBM Research Report, 2014.

> "Our results show that **containers result in equal or better performance than VMs in almost all cases.**"

> "Although **containers themselves have almost no overhead**, **Docker is not without performance gotchas**. Docker volumes have noticeably better performance than files stored in AUFS. Docker's NAT also introduces overhead for workloads with high packet rates. These **features** represent a **tradeoff between ease of management and performance and should be considered on a case-by-case basis.**"

BenchFlow Configures Docker for Performance by Default

Call for Collaboration

WfMSs, process models, process logs

WfMSs

- We want to add more and more WfMSs to the benchmark
- **Contact us for collaboration, and BenchFlow framework support**

Process Models

- We want to characterise the Workload Mix using Real-World process models
- **Share your executable BPMN 2.0 process models, even anonymised**

Execution Logs

- We want to characterise the Load Functions using Real-World behaviours
- **Share your execution logs, even anonymised**

State of the Art of WfMSs in the Cloud

Cloud WfMSs, business process optimisations, capacity planning

- Cloud WfMSs: emerging market
- BPaaS: started to be evaluated
- Business Process Execution Optimisations
- Cost-aware WfMSs

State of the Art of WfMSs in the Cloud

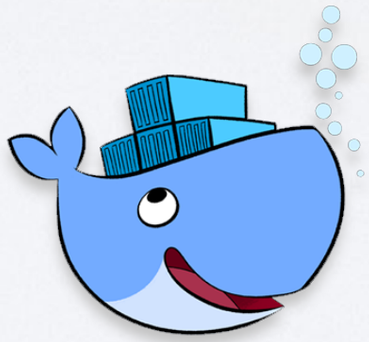
Cloud WfMSs, business process optimisations, capacity planning

- Cloud WfMSs: emerging market
- BPaaS: started to be evaluated
- Business Process Execution Optimisations
- Cost-aware WfMSs

We Focus on Capacity Planning with a Widely Used WfMSs

BenchFlow Framework

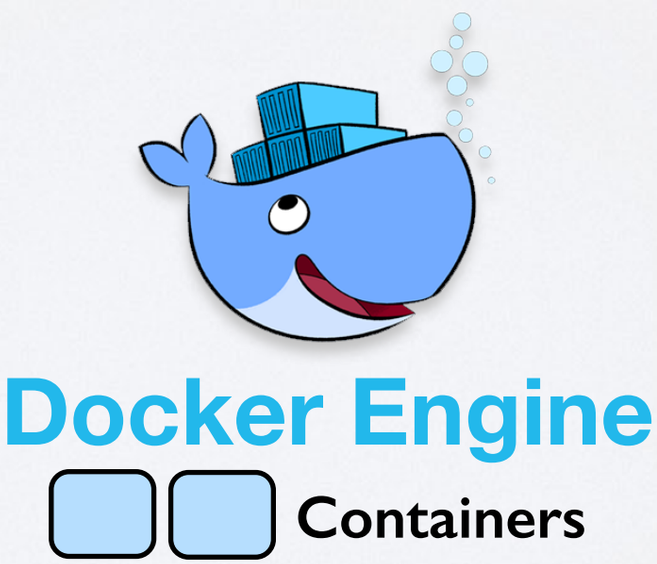
system under test



Docker Engine

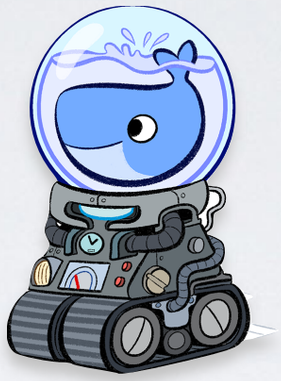
BenchFlow Framework

system under test



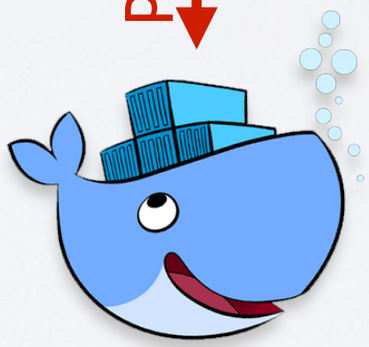
BenchFlow Framework

system under test



Docker Machine

provides

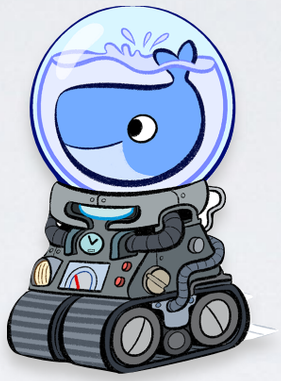


Docker Engine

Containers

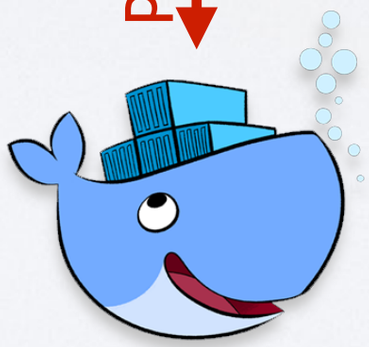
BenchFlow Framework

system under test



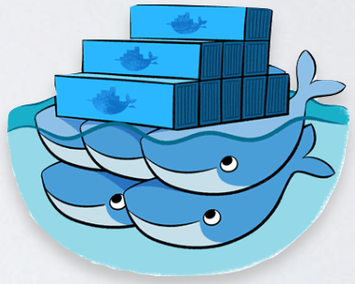
Docker Machine

provides



Docker Engine

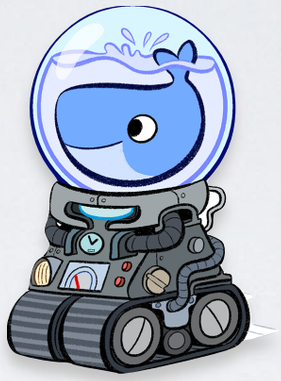
Containers



Docker Swarm

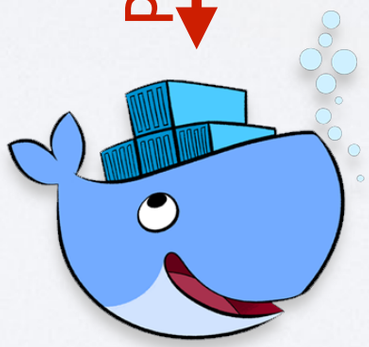
BenchFlow Framework

system under test



Docker Machine

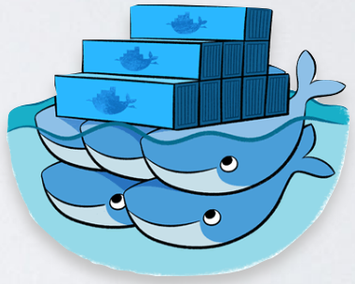
provides



Docker Engine

Containers

manages

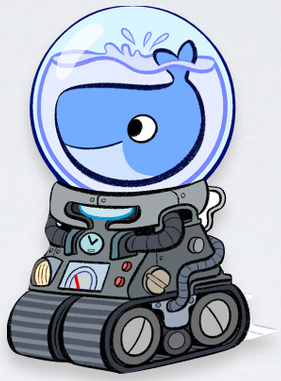


Docker Swarm

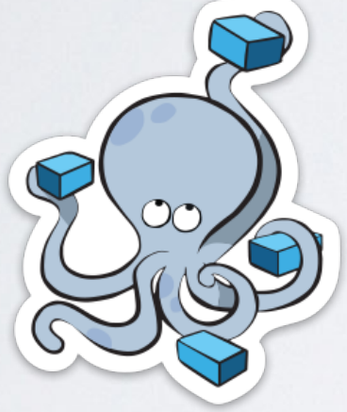
Servers

BenchFlow Framework

system under test

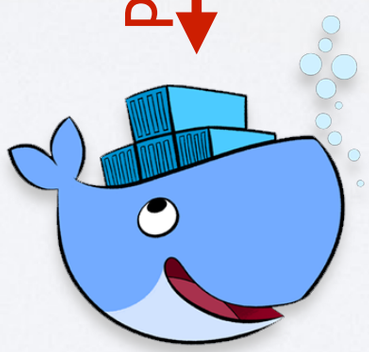


Docker Machine



Docker Compose

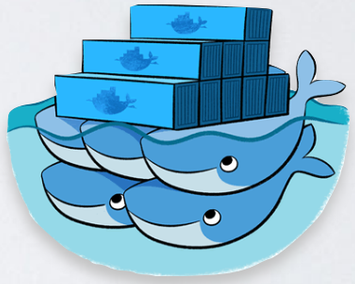
provides



Docker Engine

Containers

manages

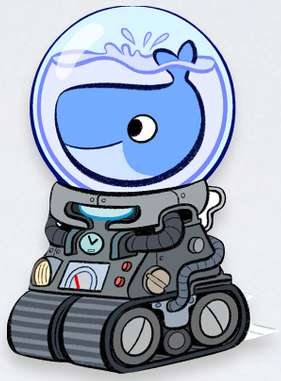


Docker Swarm

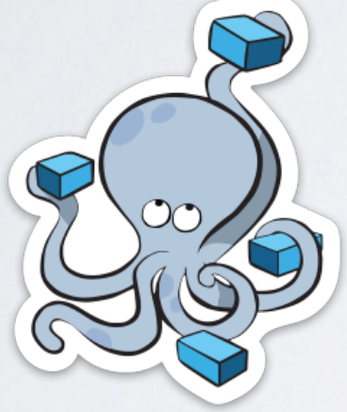
Servers

BenchFlow Framework

system under test



Docker Machine

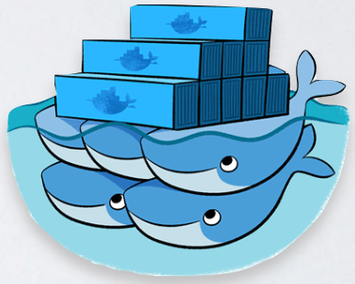


Docker Compose

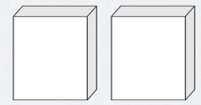
SUT's Deployment Conf.

deploys

provides

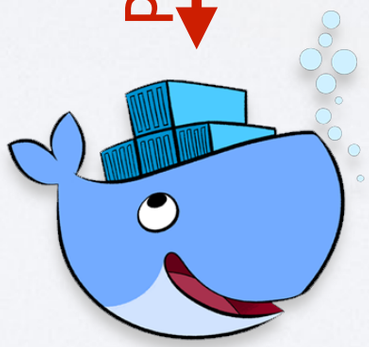


Docker Swarm



Servers

manages



Docker Engine

Containers

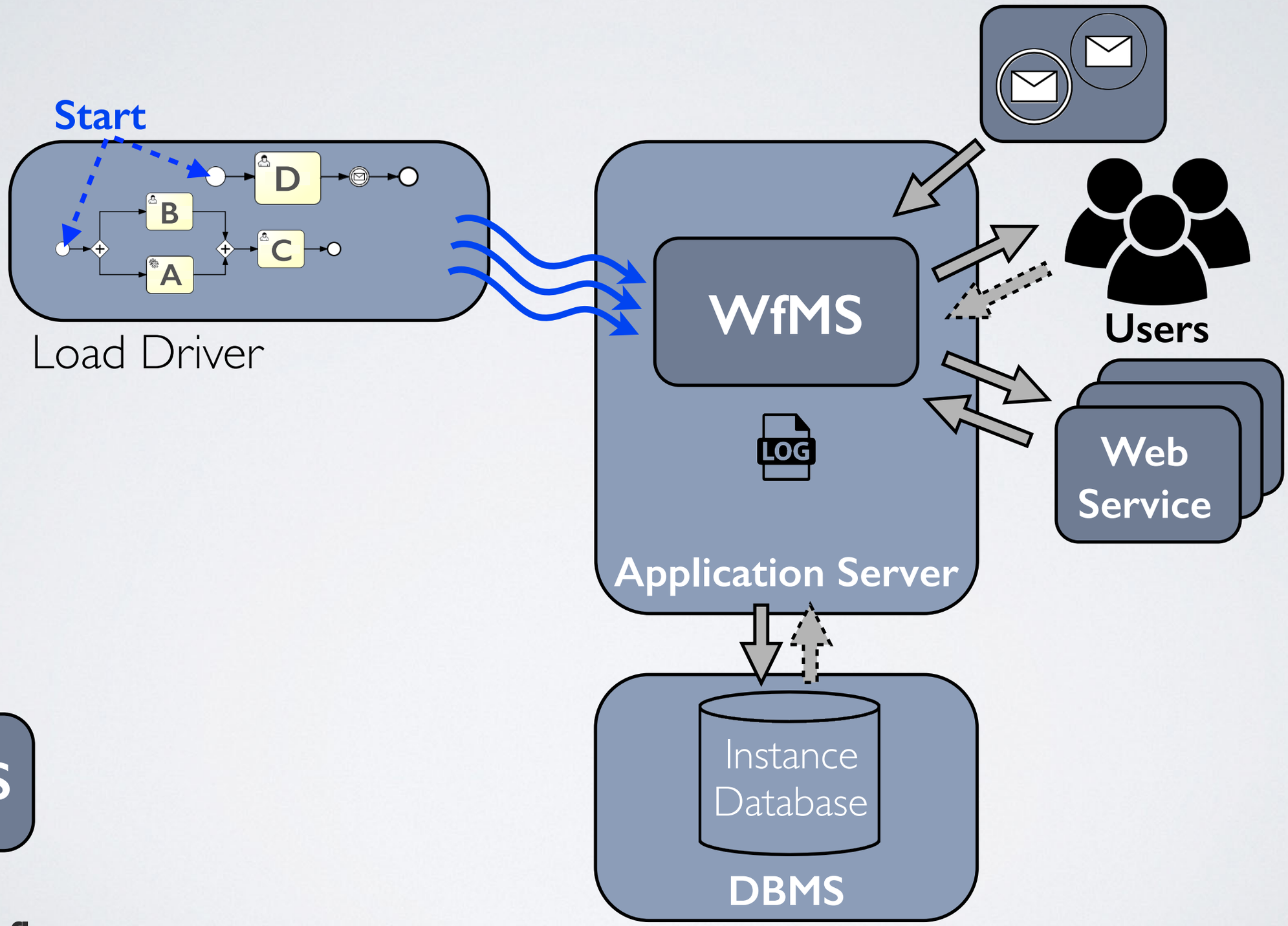
Server-side Data and Metrics Collection

asynchronous execution of workflows

Start Workflow

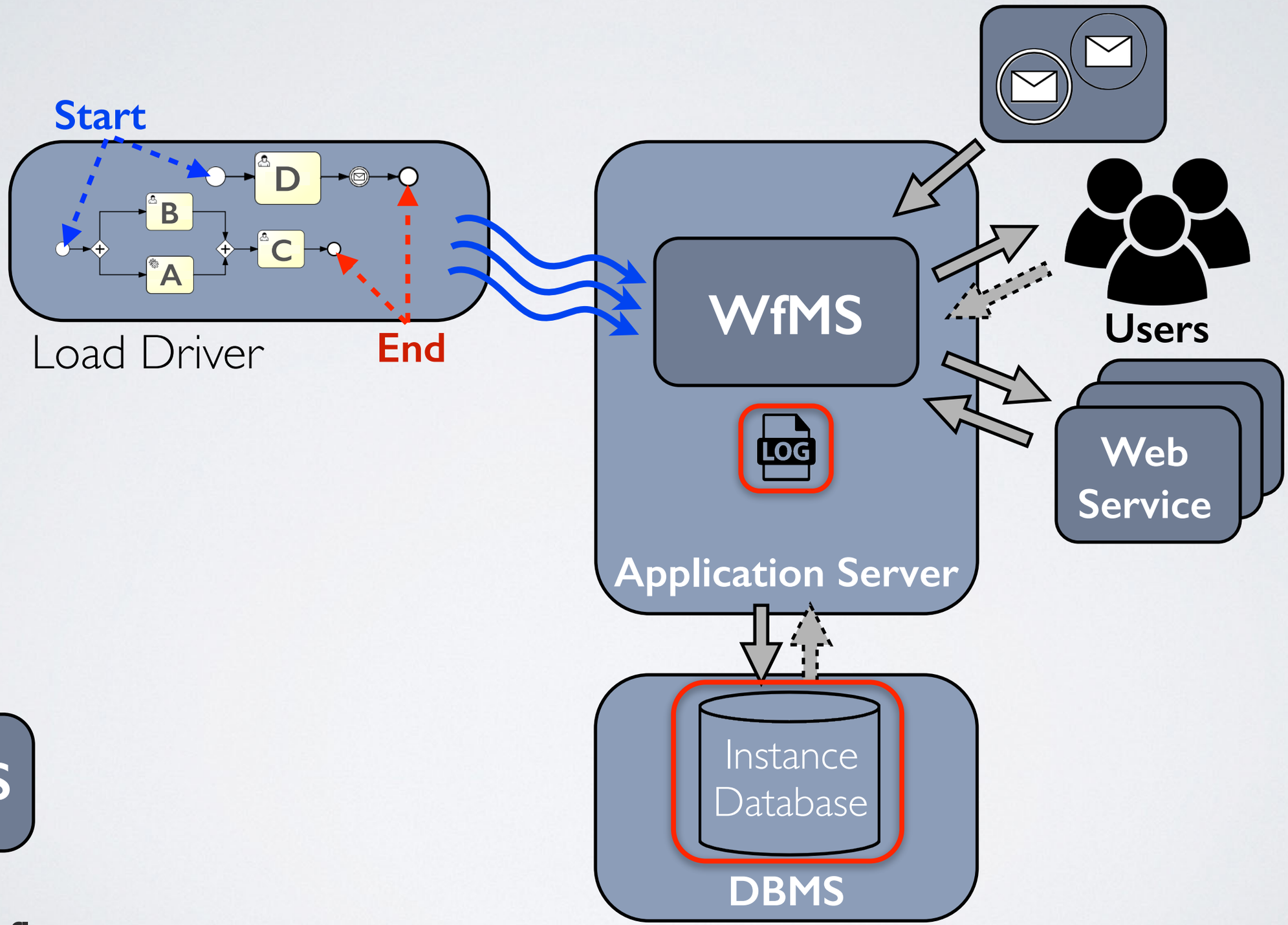
Server-side Data and Metrics Collection

asynchronous execution of workflows



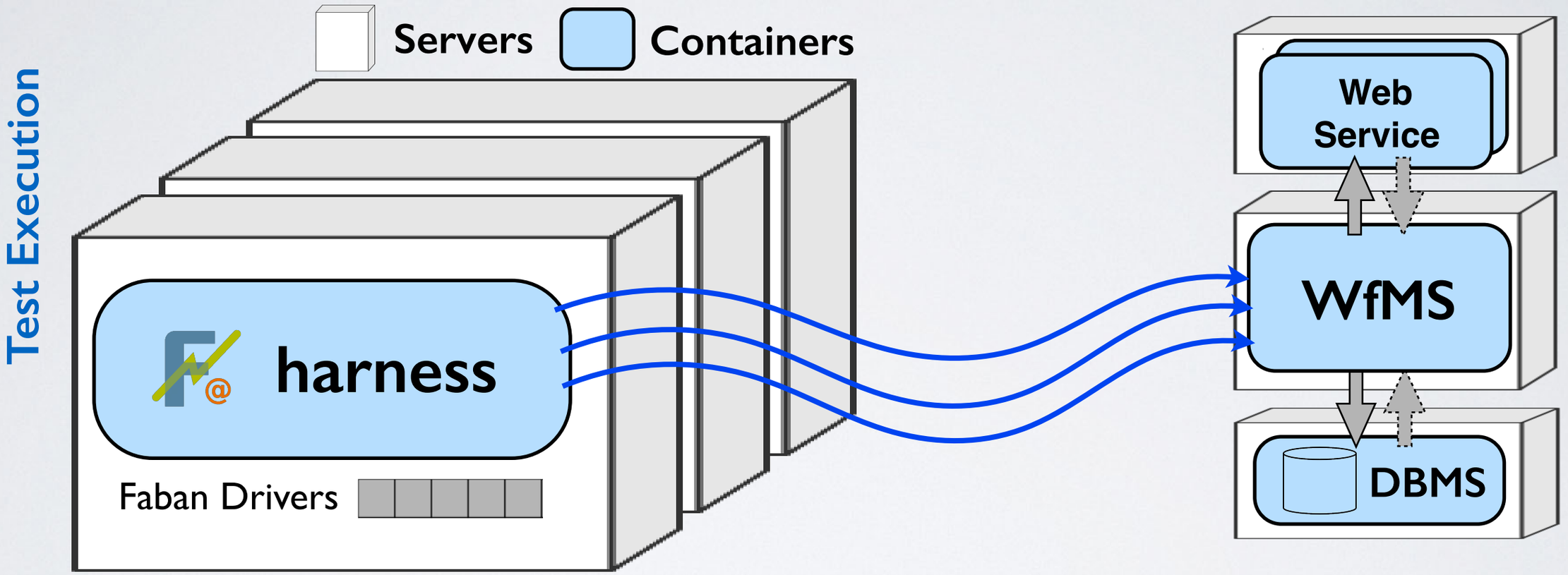
Server-side Data and Metrics Collection

asynchronous execution of workflows



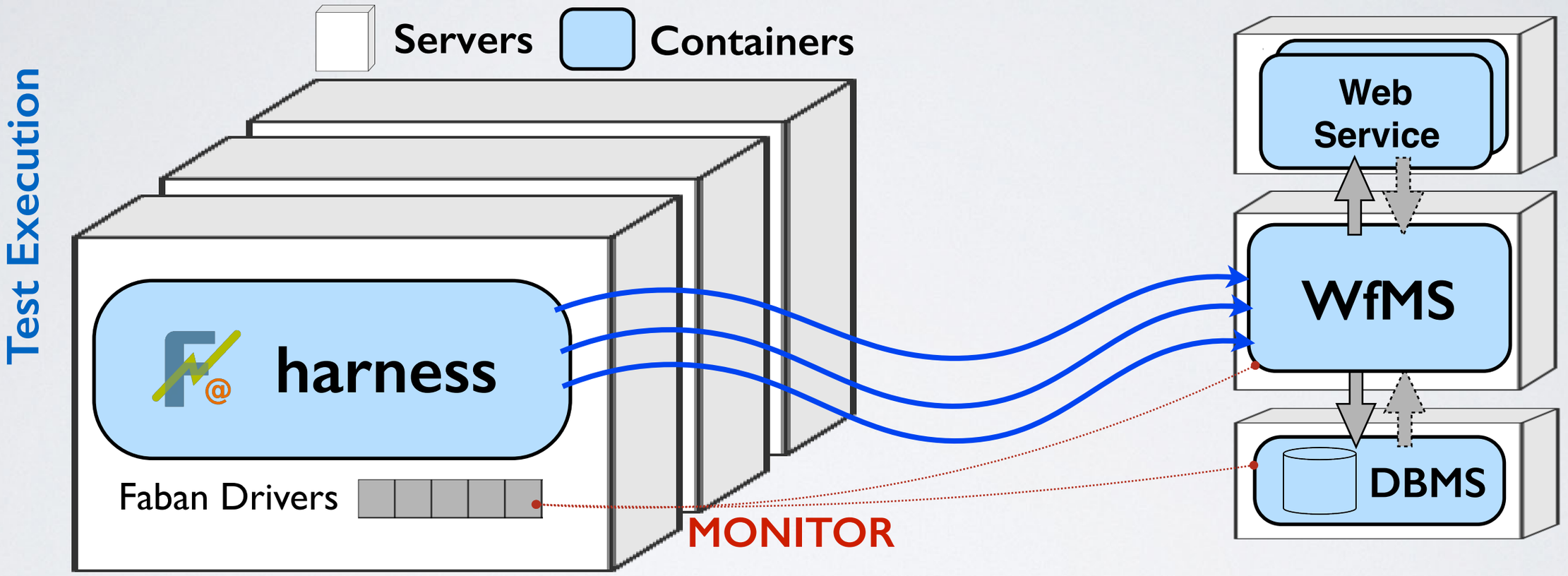
Server-side Data and Metrics Collection

monitors



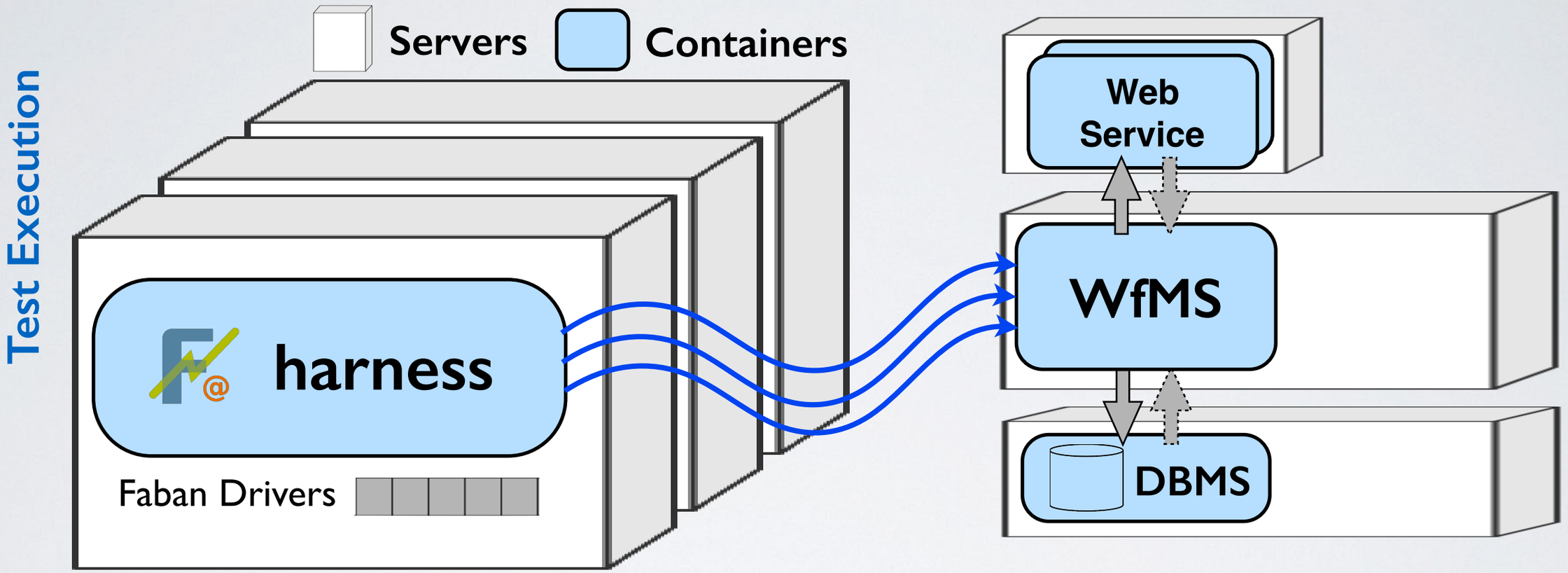
Server-side Data and Metrics Collection

monitors



Server-side Data and Metrics Collection

monitors



Monitors' Characteristics:

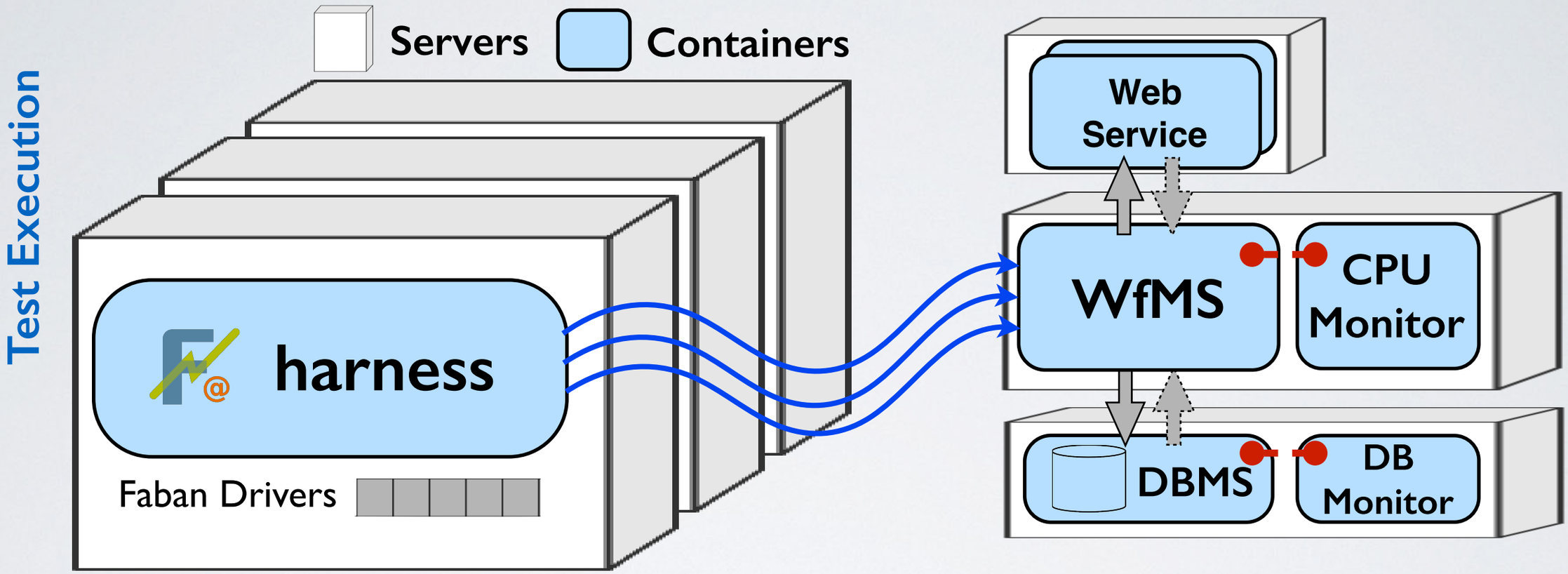
- RESTful services
- Lightweight (written in Go)
- As less invasive on the SUT as possible

Examples of Monitors:

- CPU usage
- Database state

Server-side Data and Metrics Collection

monitors



Monitors' Characteristics:

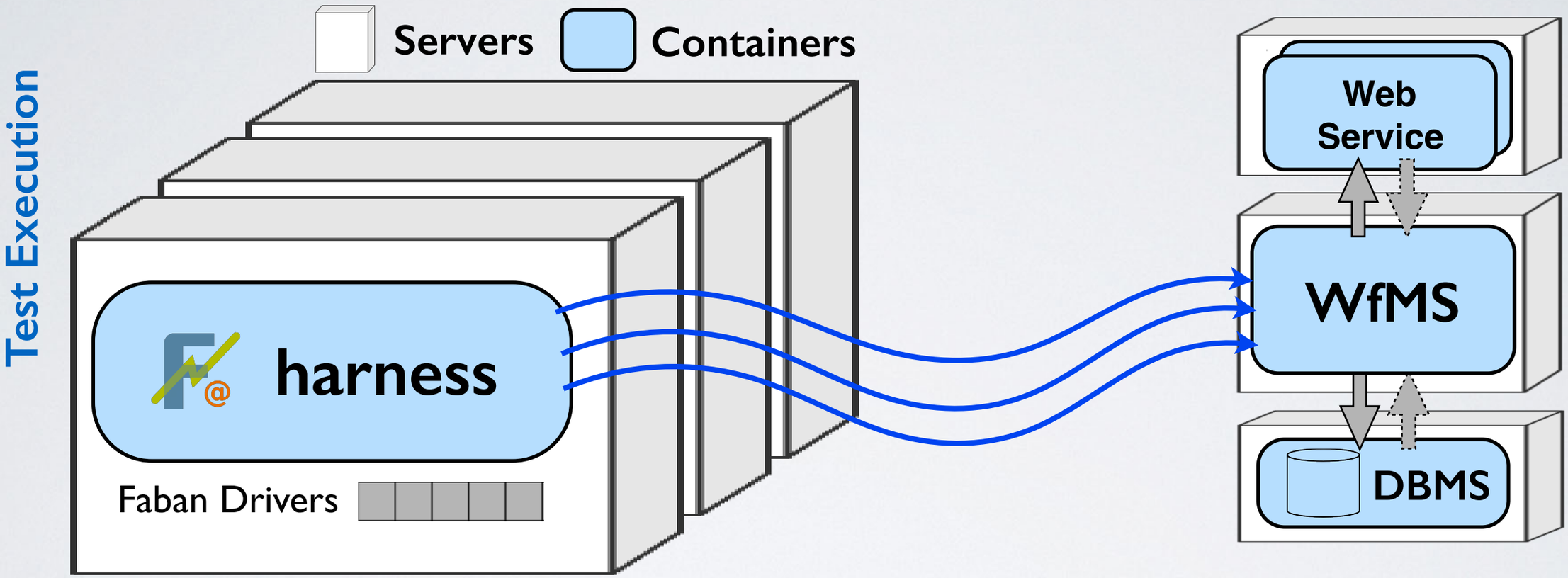
- RESTful services
- Lightweight (written in Go)
- As less invasive on the SUT as possible

Examples of Monitors:

- CPU usage
- Database state

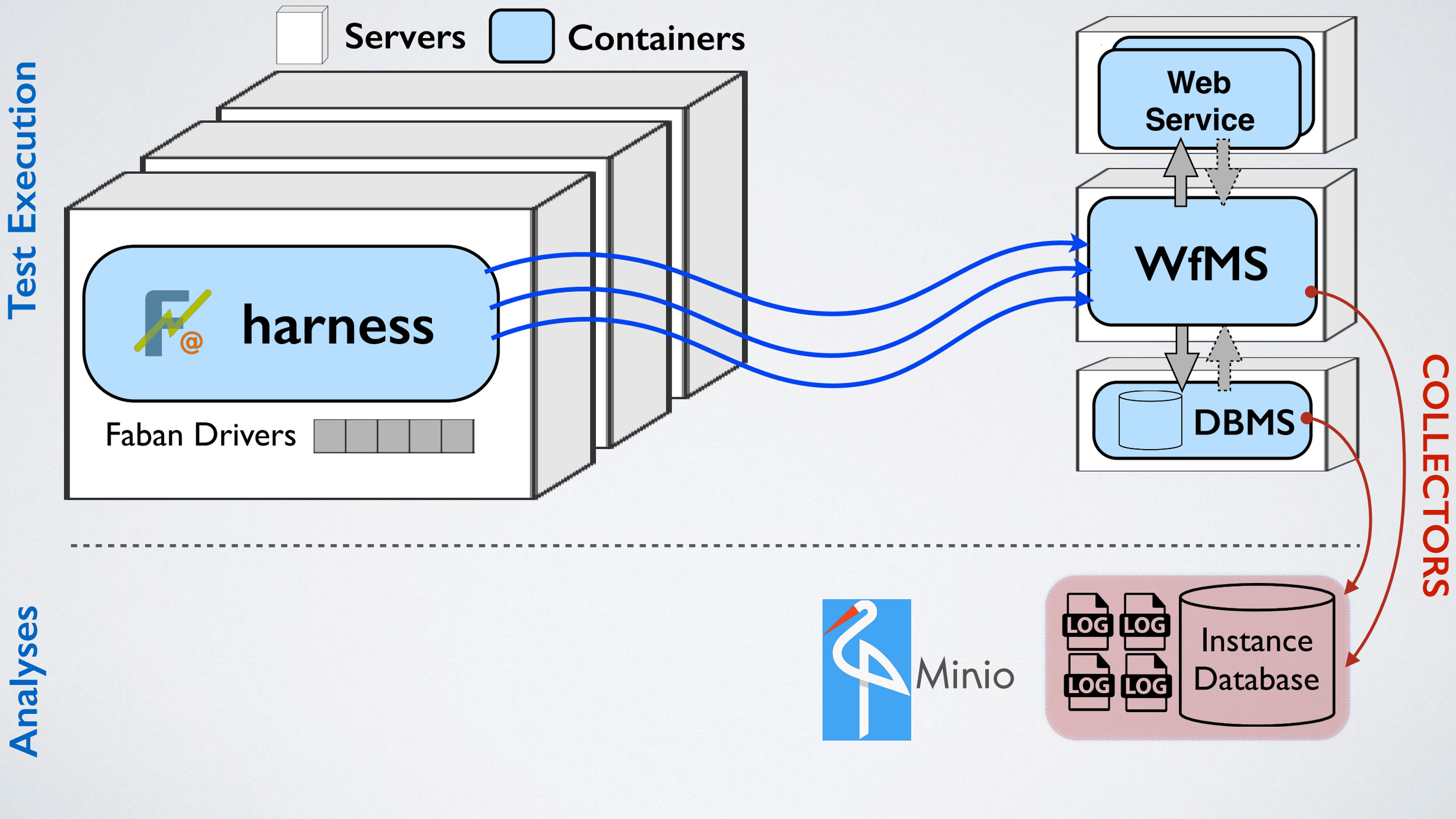
Server-side Data and Metrics Collection

collect data



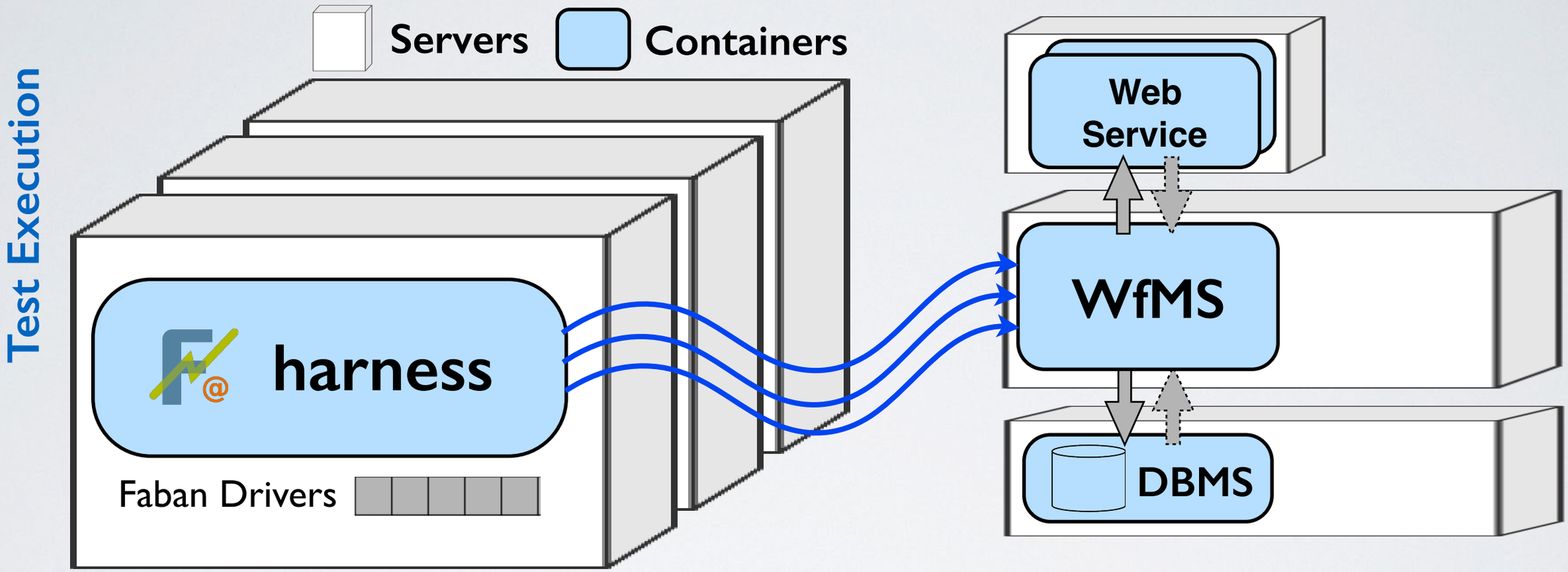
Server-side Data and Metrics Collection

collect data



Server-side Data and Metrics Collection

collect data



Collectors' Characteristics:

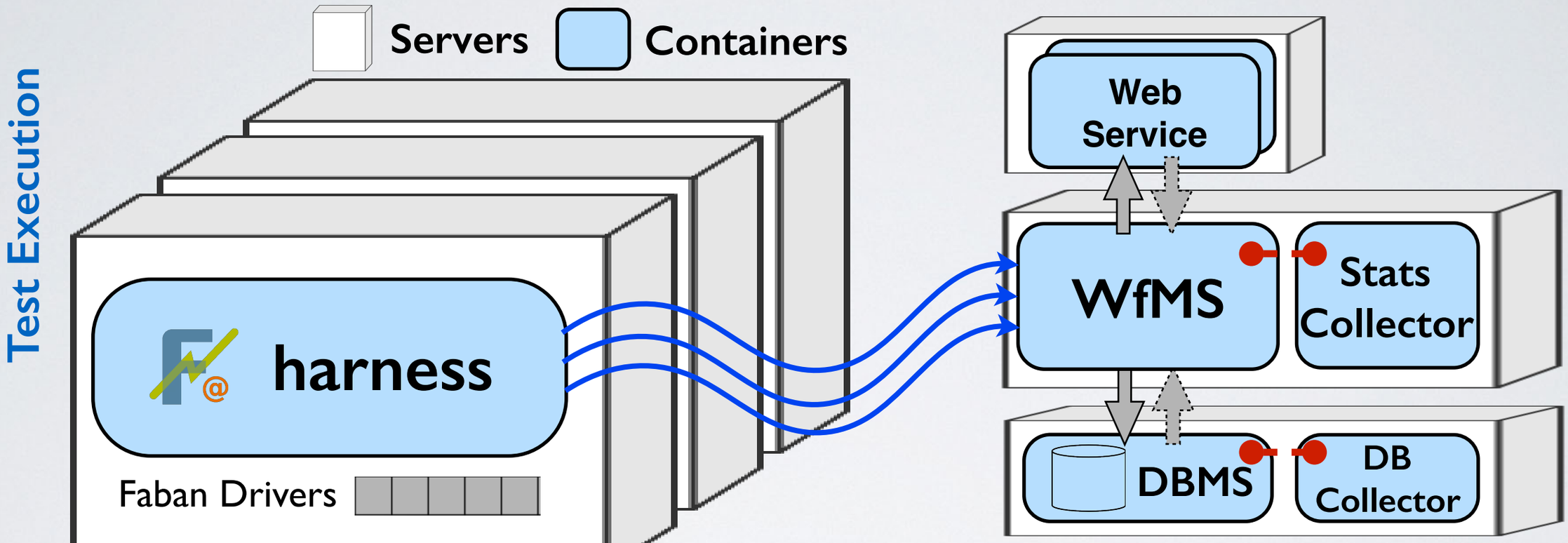
- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Buffer data locally

Examples of Collectors:

- Container's Stats (e.g., CPU usage)
- Database dump
- Applications Logs

Server-side Data and Metrics Collection

collect data



Collectors' Characteristics:

- RESTful services
- Lightweight (written in Go)
- Two types: online and offline
- Buffer data locally

Examples of Collectors:

- Container's Stats (e.g., CPU usage)
- Database dump
- Applications Logs