Imperial College London

Quality-driven development of multi-cloud applications

Giuliano Casale
Department of Computing
Imperial College London
g.casale@imperial.ac.uk

Joint work with:
D. Dubois, R. Osman, J. F. Pérez

Imperial College
London

Computing@Imperial

Dept. of Computing:

- Languages & Systems
- Logic & AI
- Quantitative Analysis ⟵
- Software Engineering
- Vision & Robotics

My research:

- Cloud computing
- Performance engineering
- Stochastic modelling

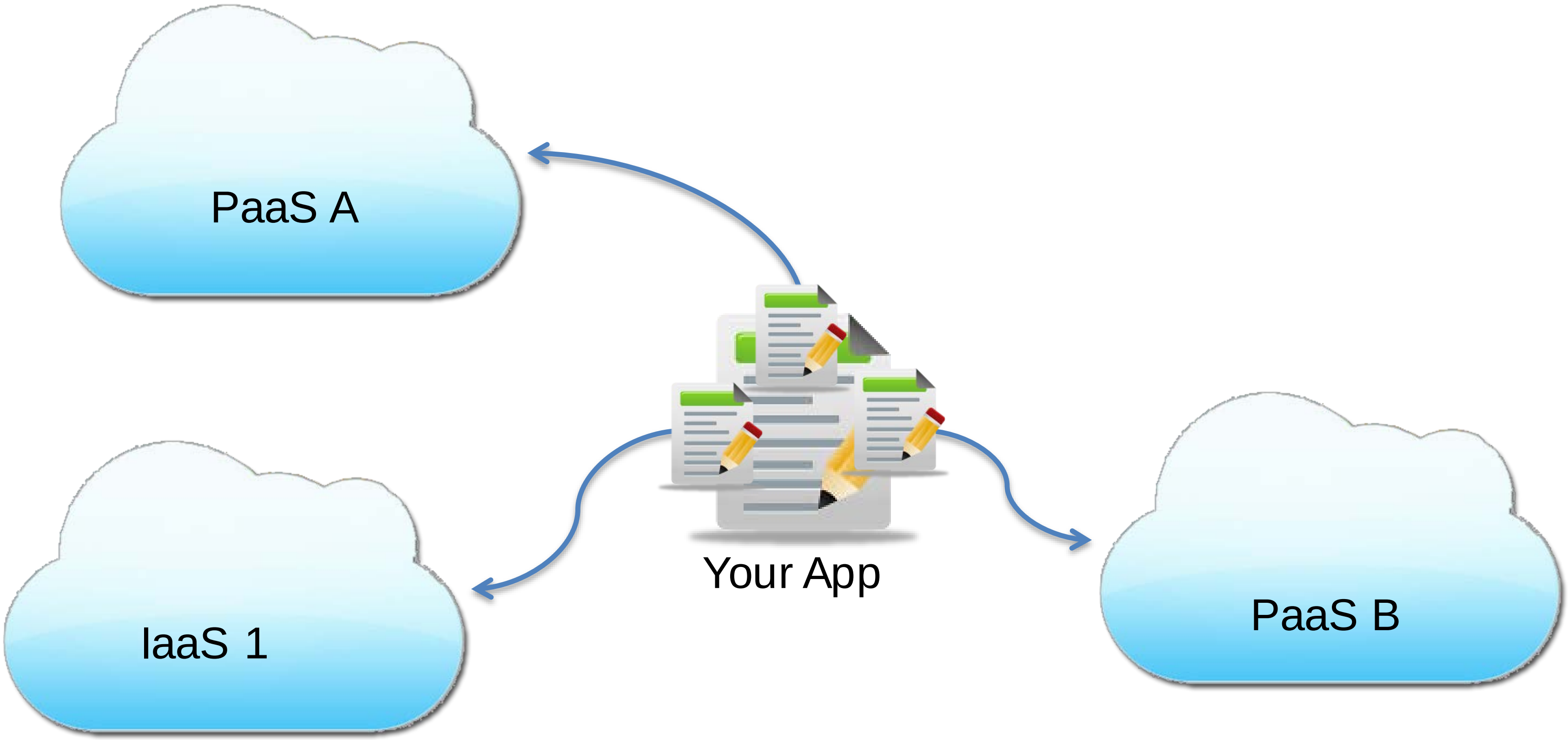
Imperial College London

MODAClouds

- Large-Scale Integrated Project 9M€ (2012-2015)
- Quality-driven development for multi-cloud SW
- Tools: http://www.modaclouds.eu/software

Multi-Cloud Application

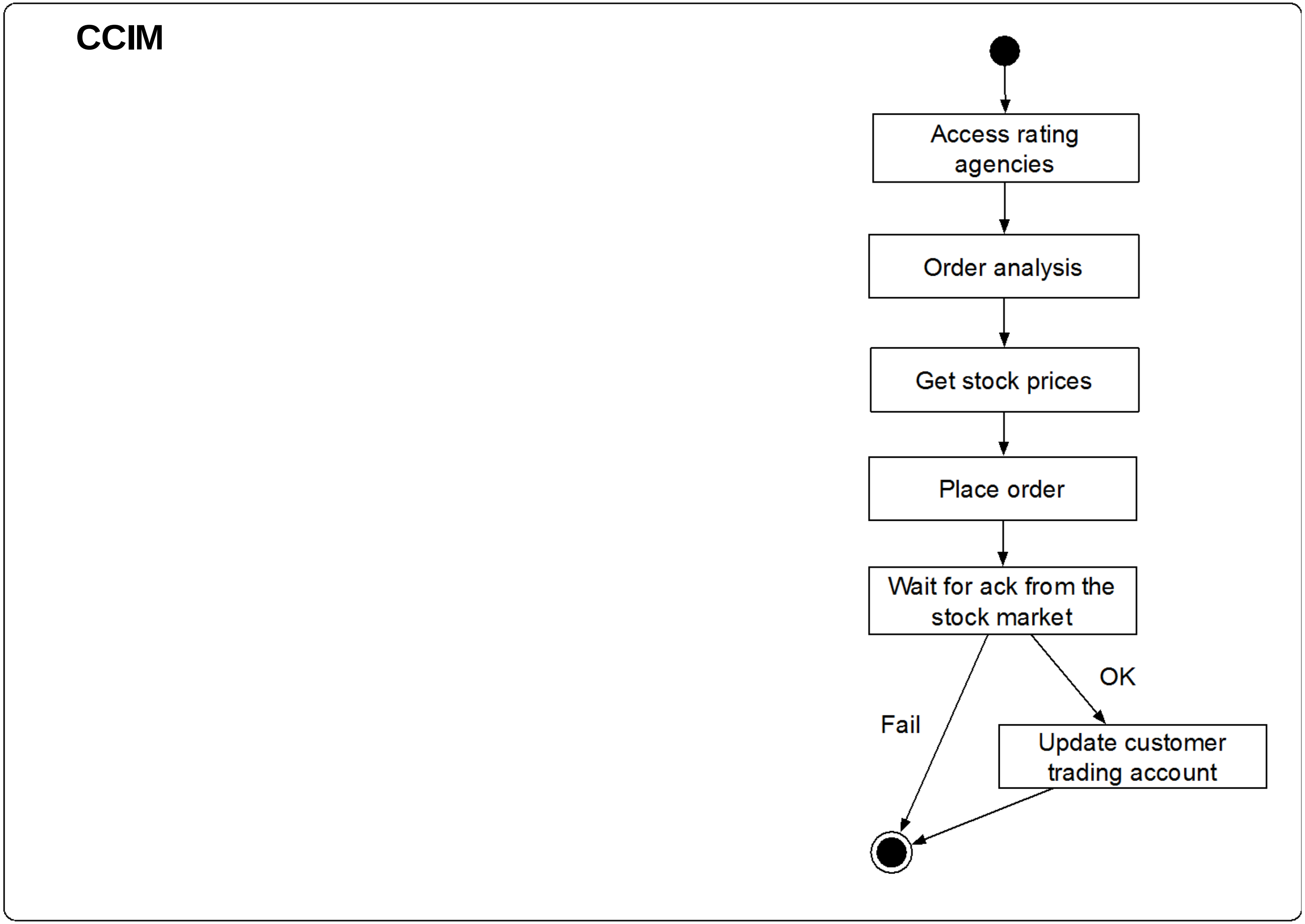
- The application can be deployed on different clouds



Imperial College London

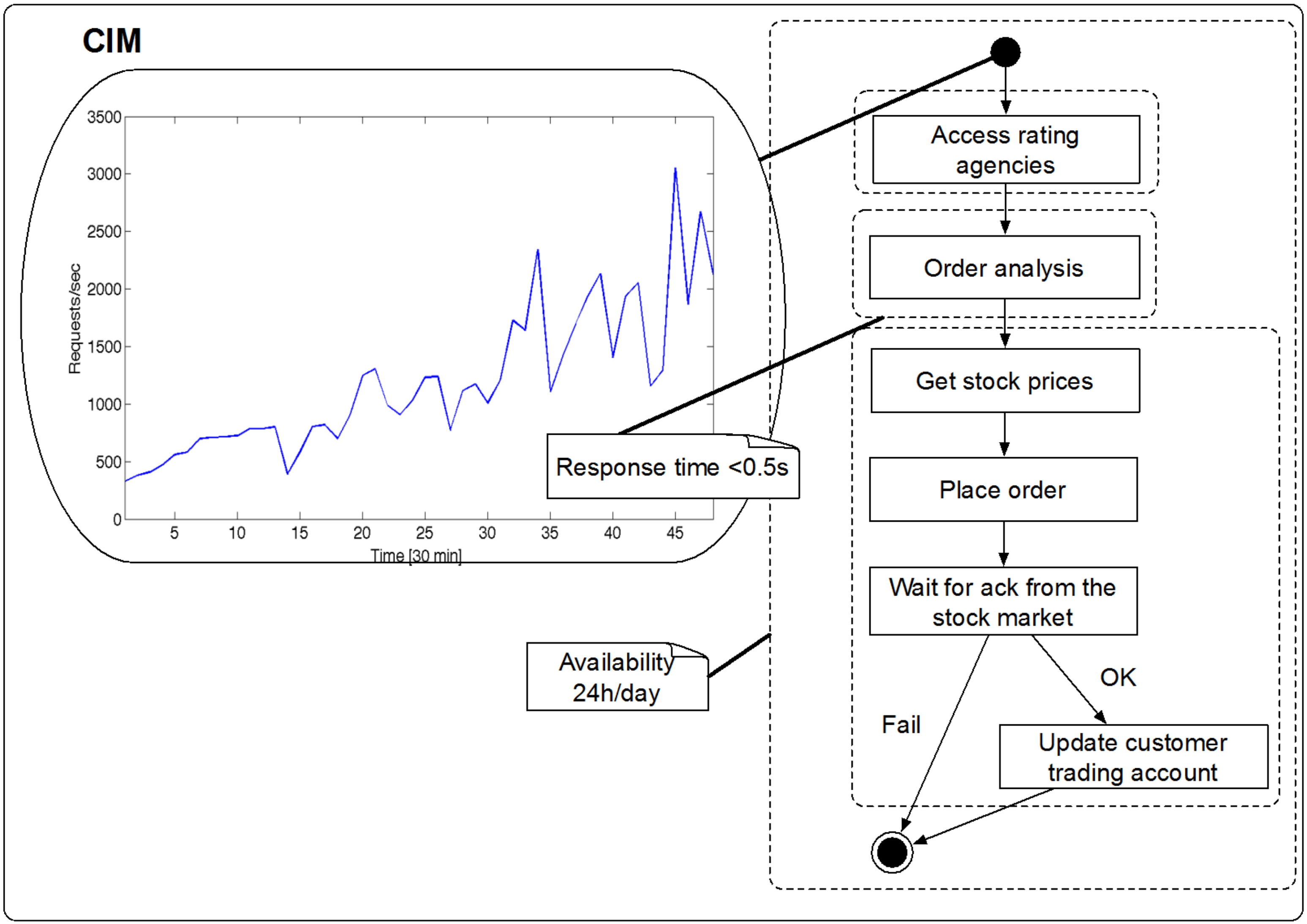
MDE for Multi-Cloud Software

Functional Modelling



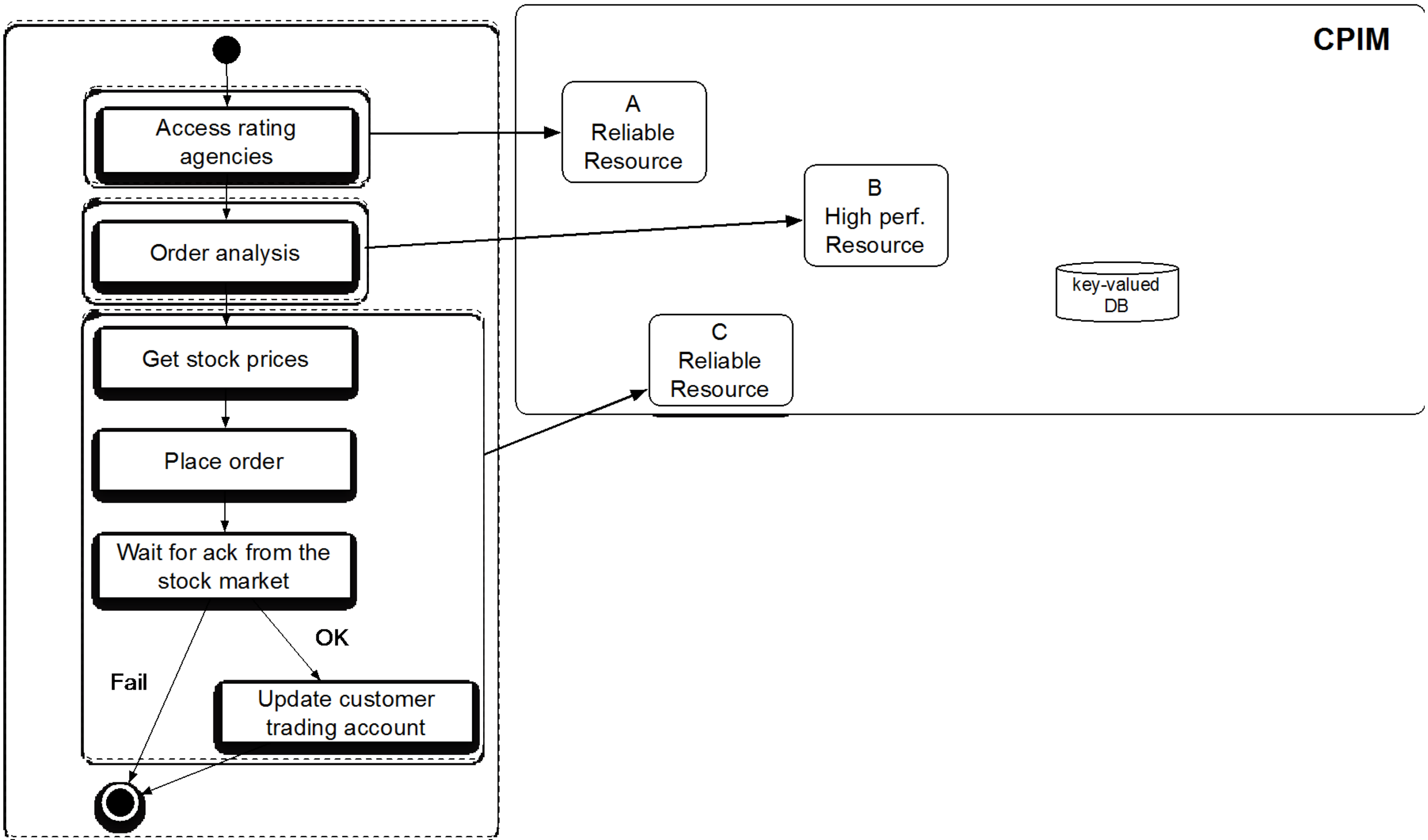
MDE for Multi-Cloud Software

QoS Requirements



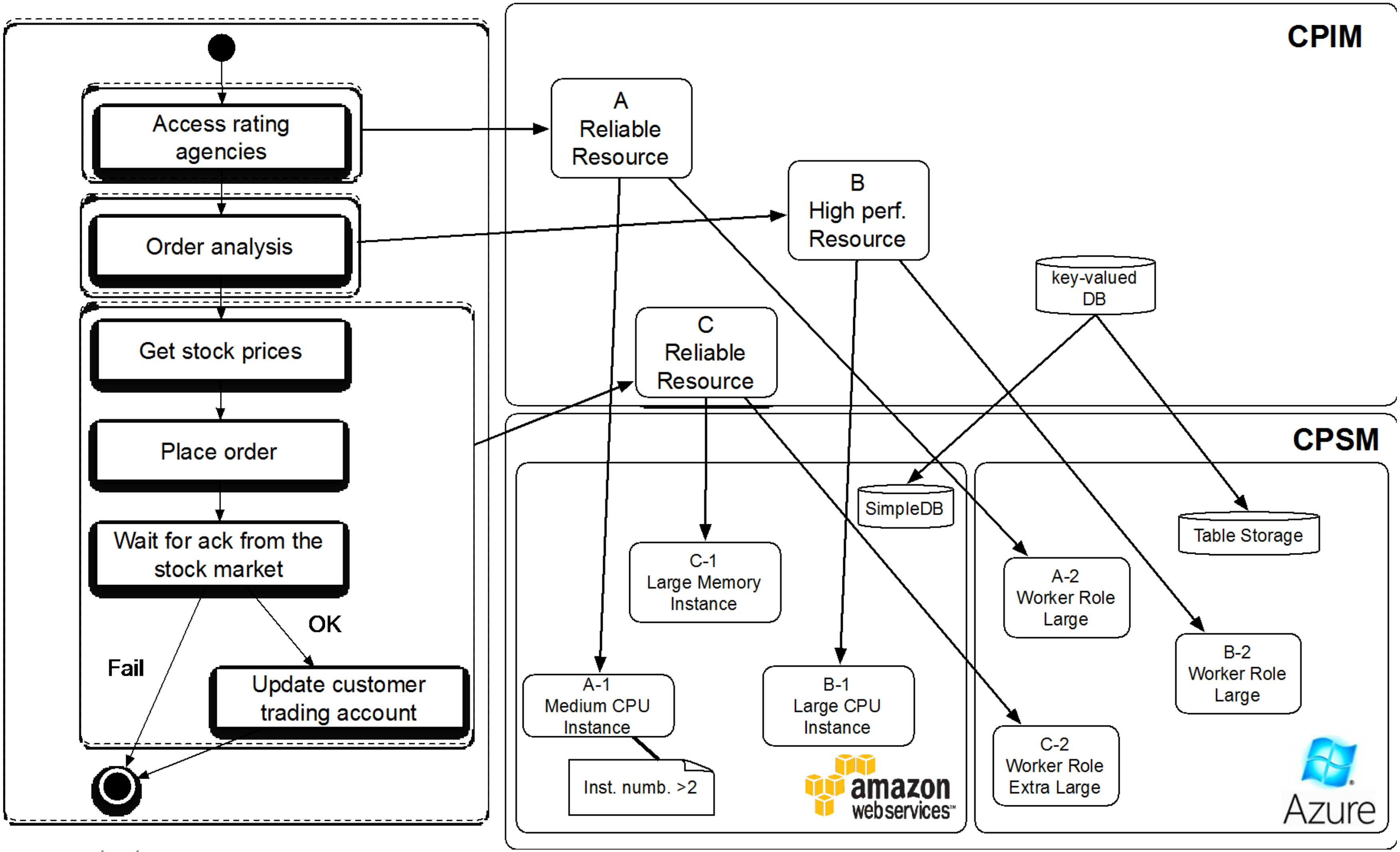
MDE for Multi-Cloud Software

Platform-Independent Models



MDE for Multi-Cloud Software

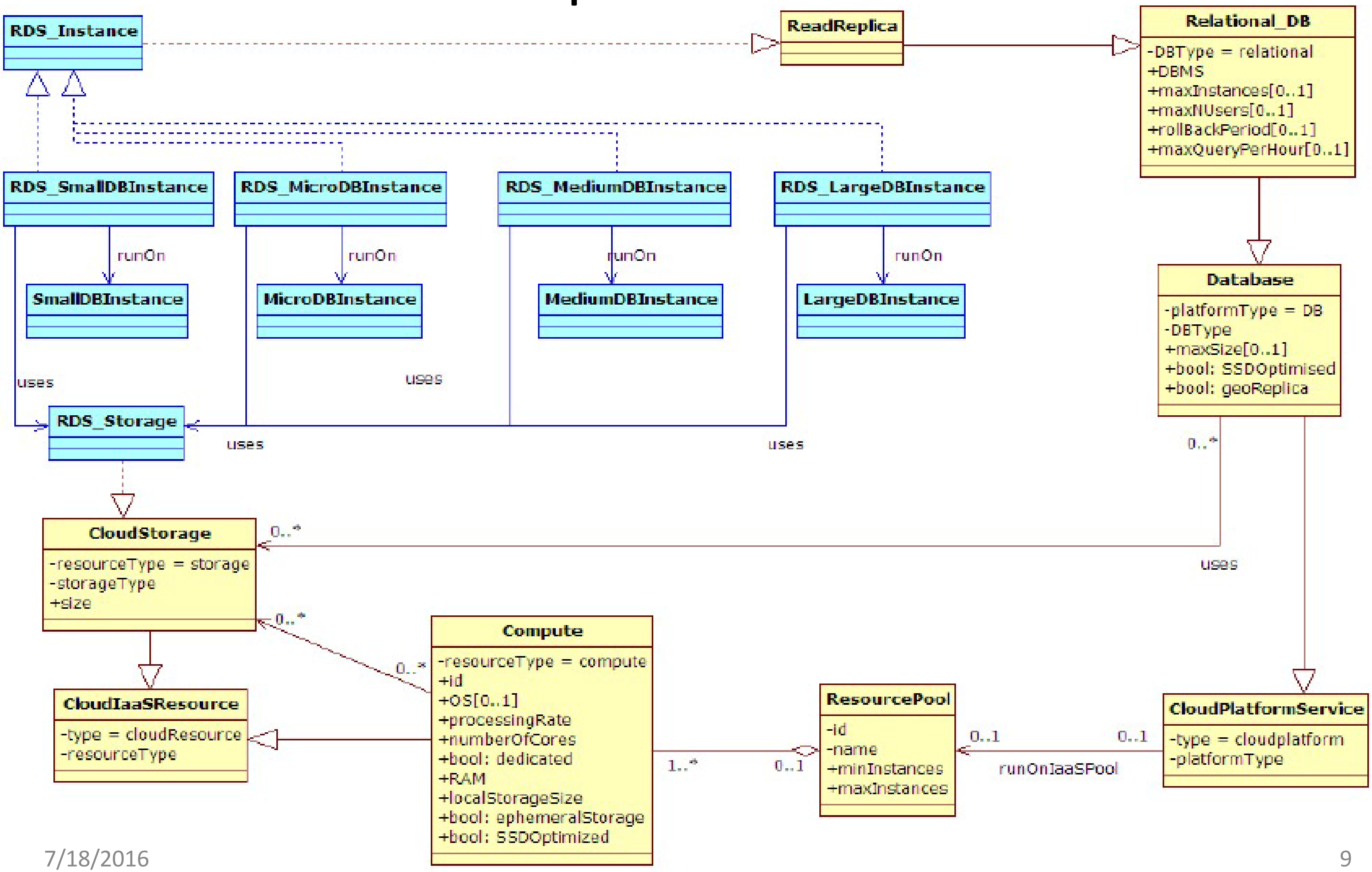
Platform-Specific Models



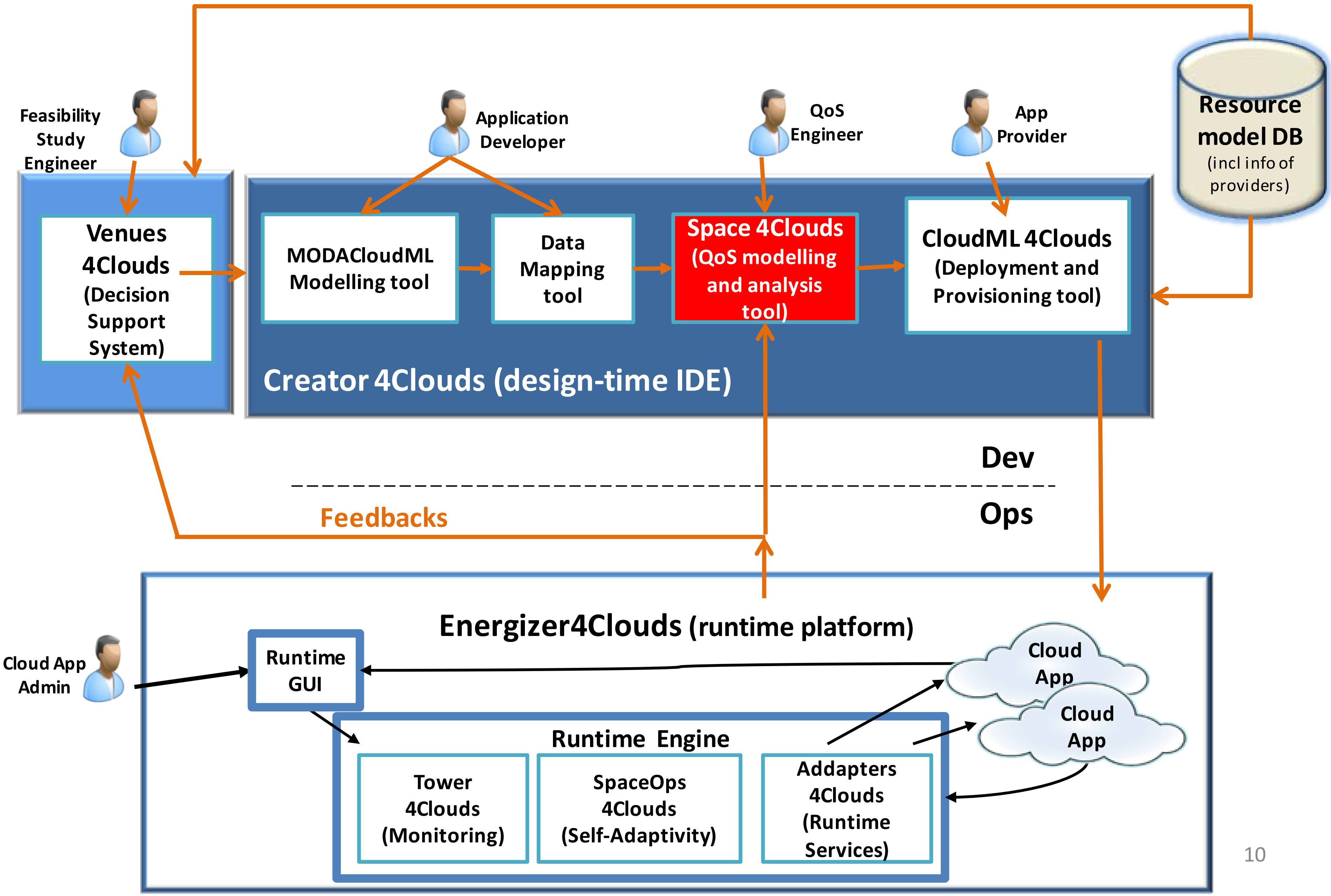
Imperial College London

CloudML DSL (cloudml.org)

Example: Amazon RDS

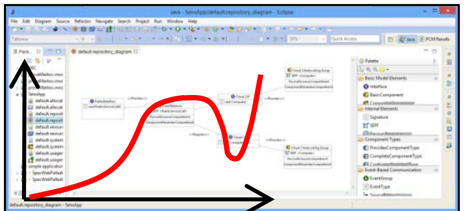


MODAClouds Platform



QoS Modelling and Analysis

Service and QoS model



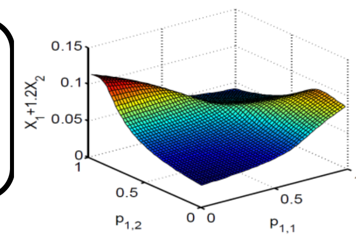
- Optimal deployment
- VM types
- Number of VM instances
- Response time analysis

Output

Transformation
CloudML ⟶ QN

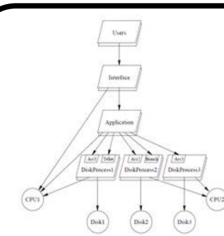
Performance Analysis

- Performance prediction
- Optimization QoS
- Capacity planning



Operational Data

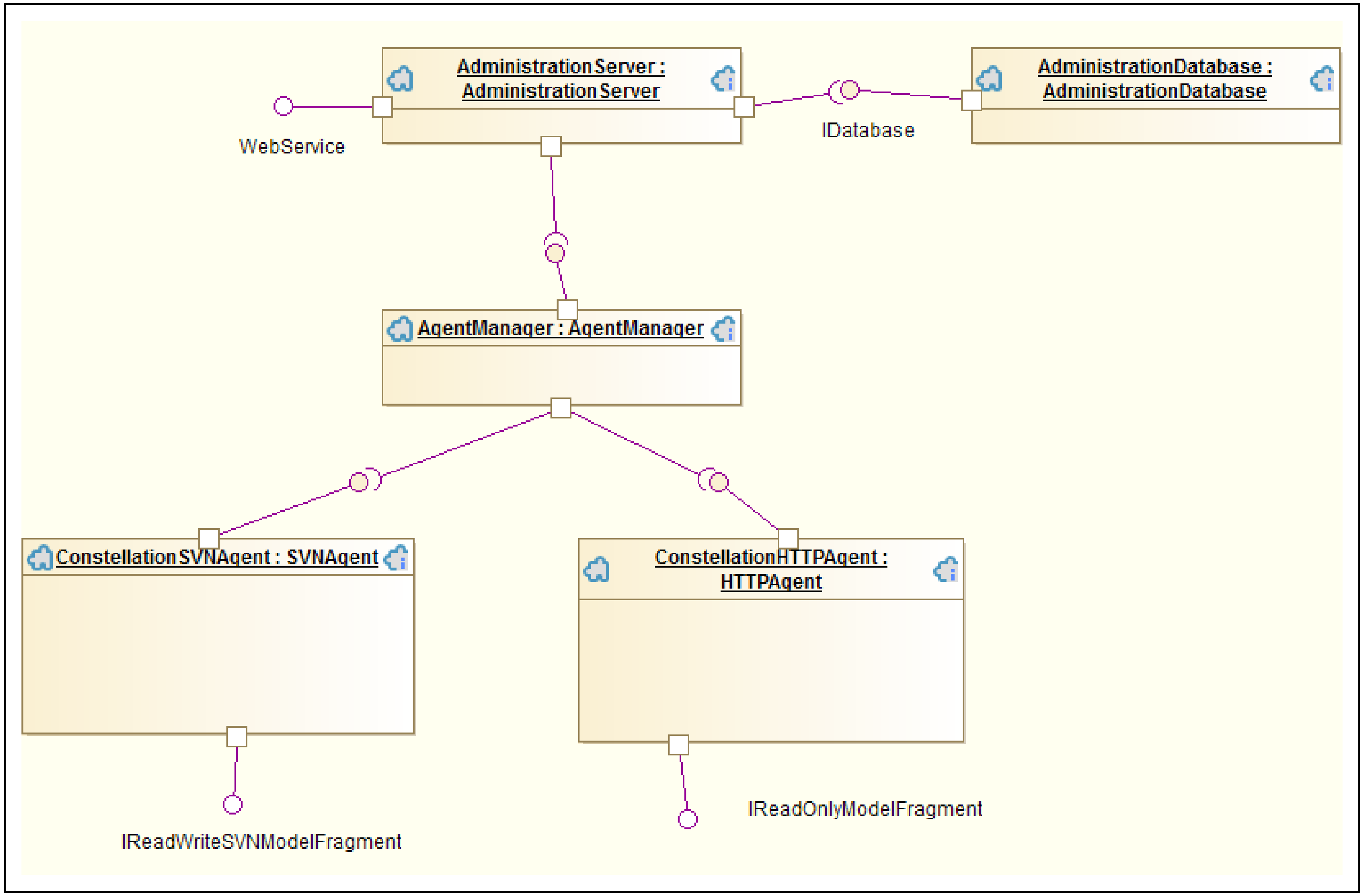
Performance Modelling Solvers



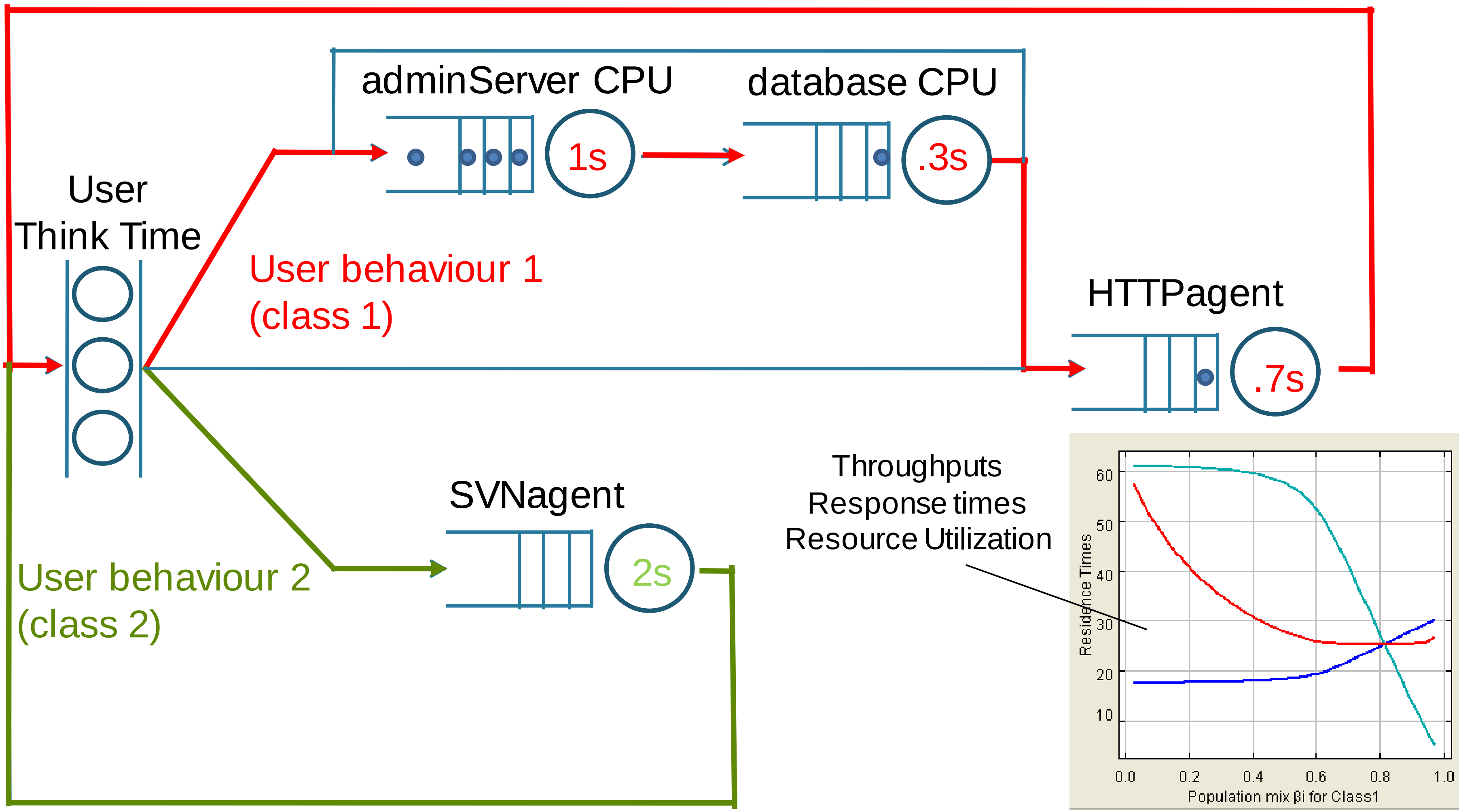
- Queueing solver
- Operational environment
- SLA percentiles

M2M Transformation

CloudML Model



M2M Transformation
Queueing Network Models



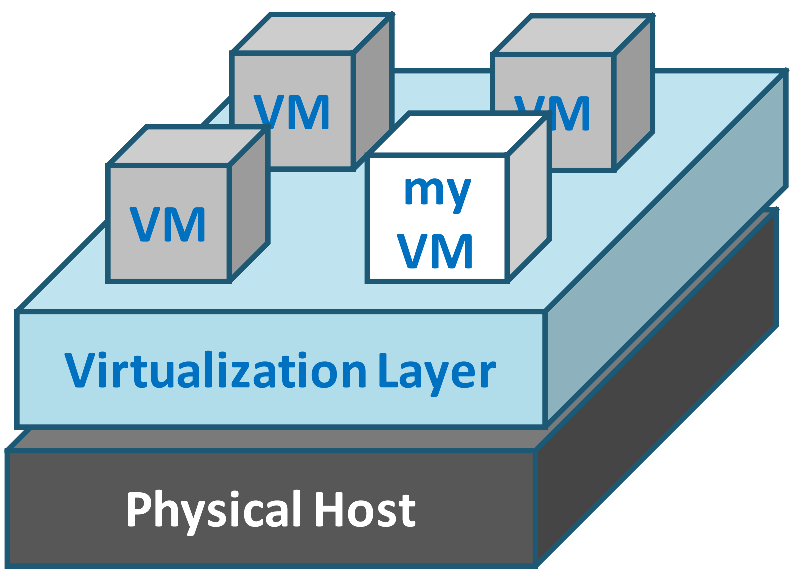
Rest of the Talk

1. How to account for cloud-specific features in QoS modelling?
   - **Approach**: shift towards modelling the operational environment
   - **Contribution**: LINE, a solver for performance engineering in time-varying environments

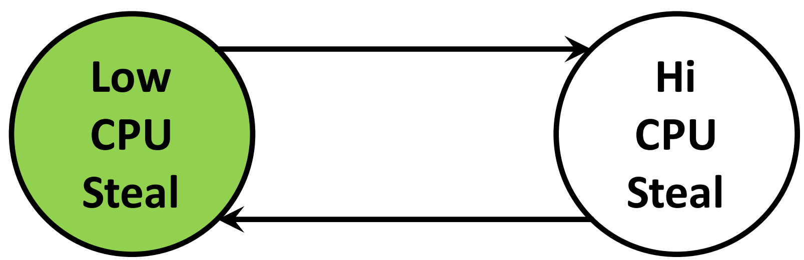
2. Application: Cloud application sizing
   - Spot VM reliability model

Cloud Multi-Tenancy

- Low Contention: normal operation
- High Contention: operational slow-down caused by VM contention on the same host

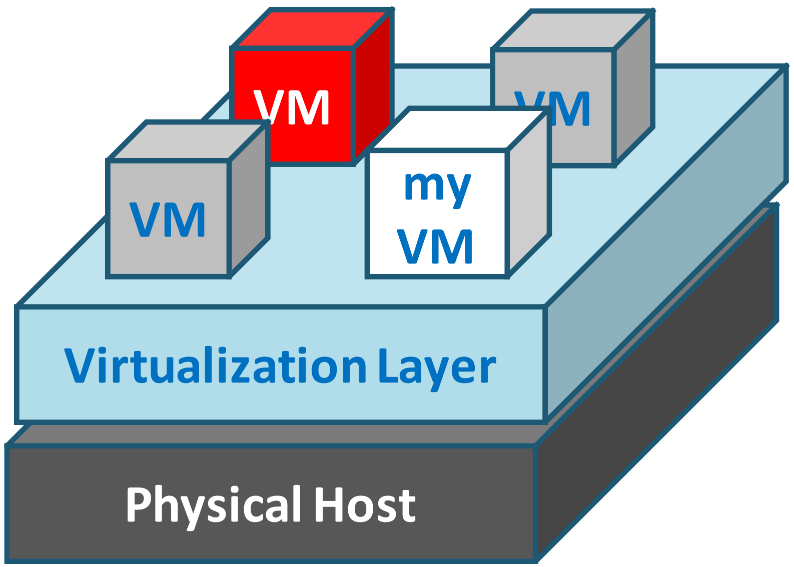


Environment Model

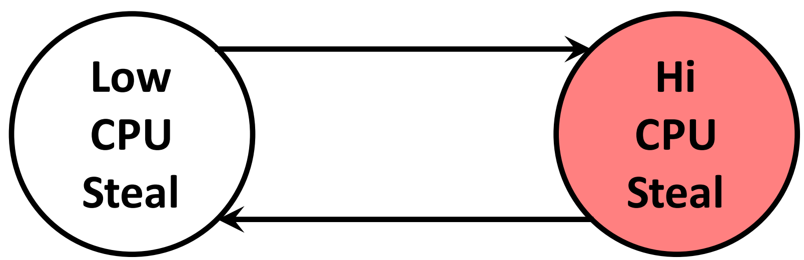


Cloud Multi-Tenancy

- Low Contention: normal operation
- High Contention: operational slow-down caused by VM contention on the same host



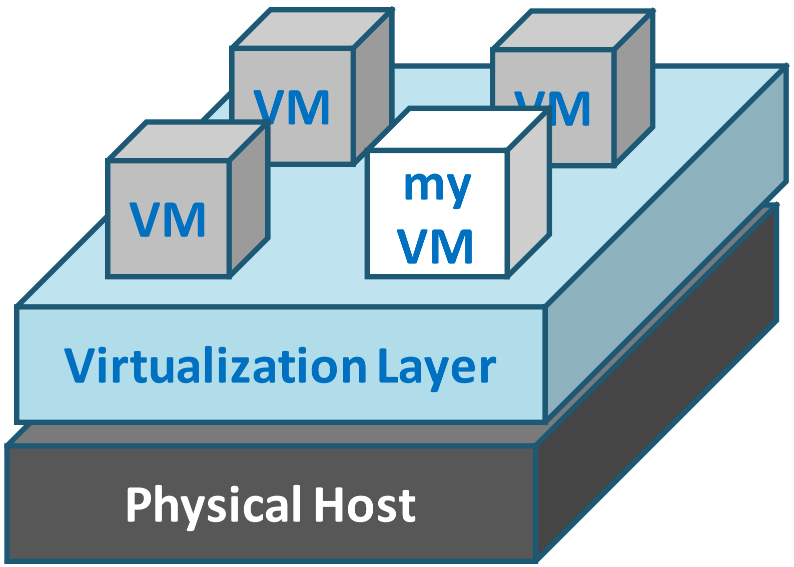
Environment Model



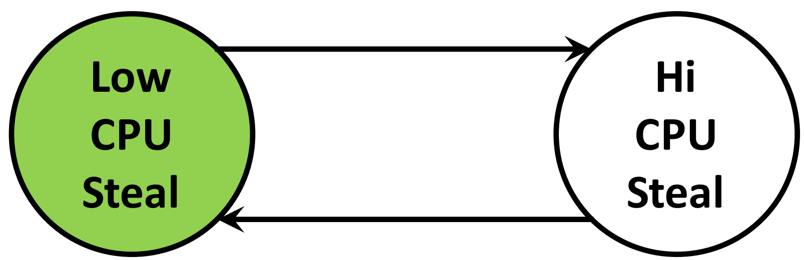
Imperial College London

Cloud Multi-Tenancy

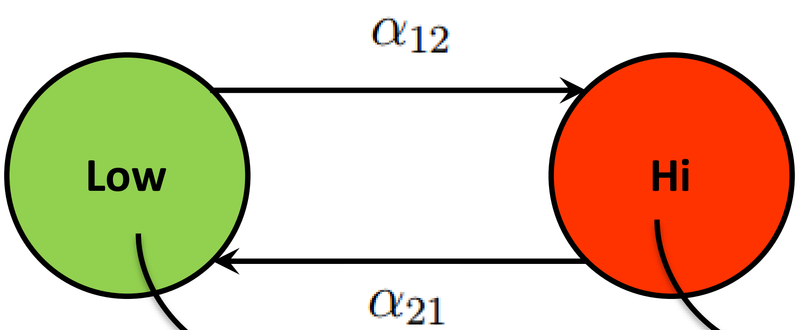
- Low Contention: normal operation
- High Contention: operational slow-down caused by VM contention on the same host



Environment Model

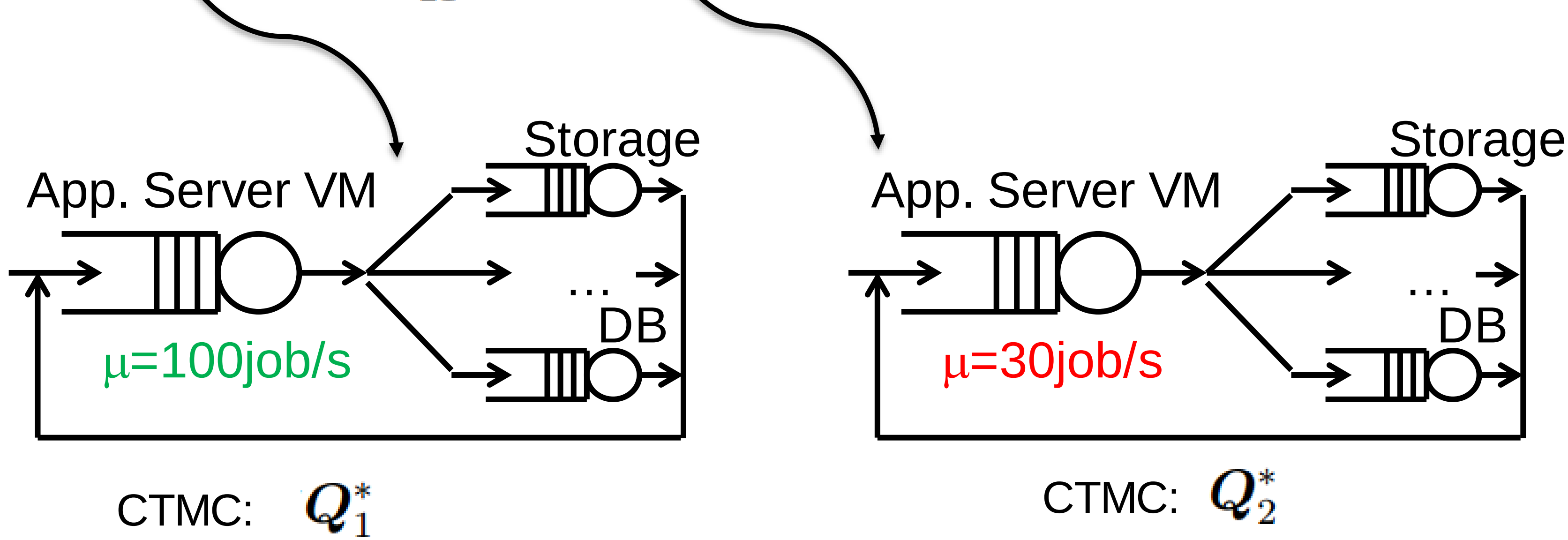


Probabilistic QoS Modelling

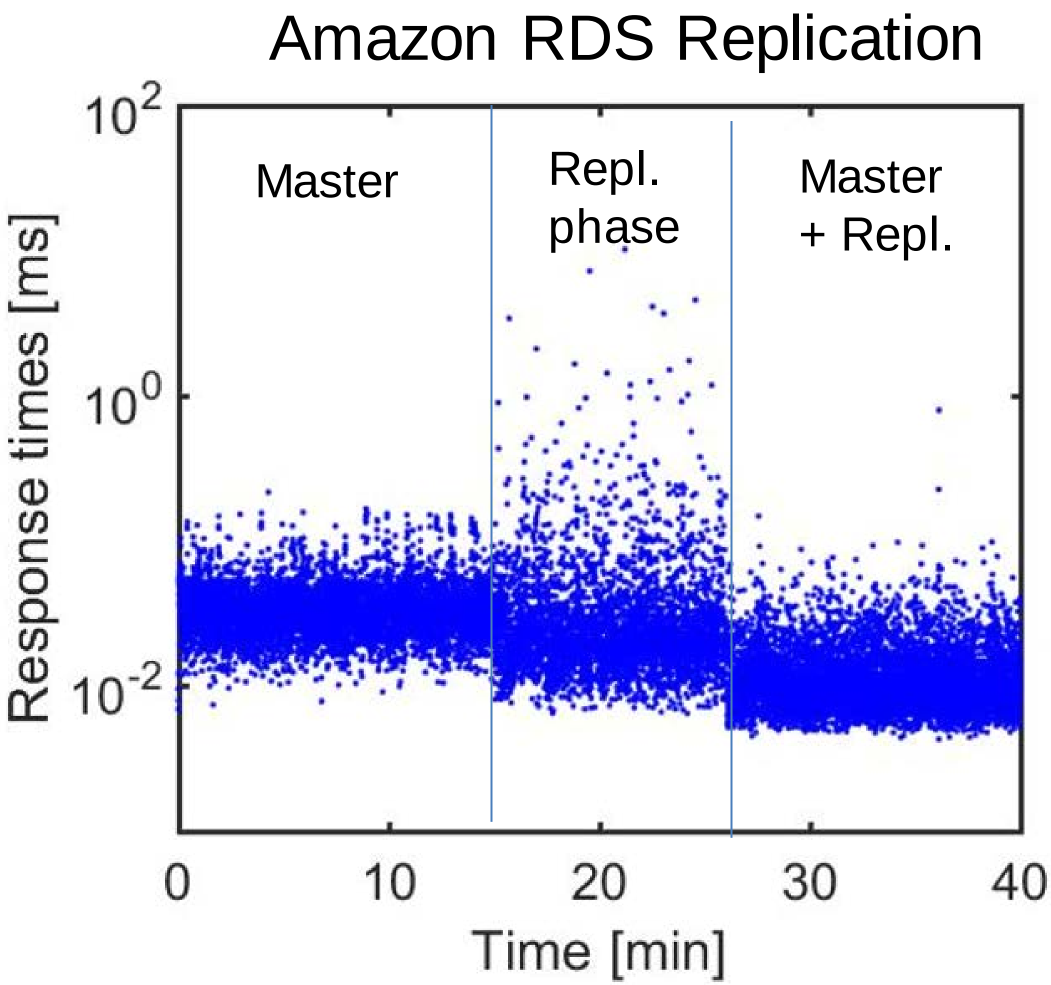


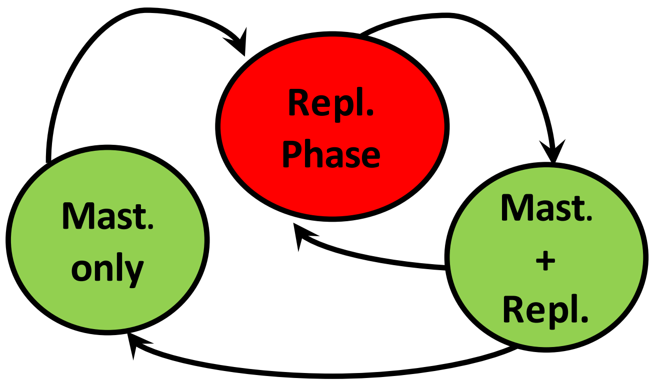
Stochastic model: CTMC

$$Q = \left[\begin{array}{c|c} Q_1^* - \alpha_{12} I & \alpha_{12} I \\ \hline \alpha_{21} I & Q_2^* - \alpha_{21} I \end{array}\right]$$

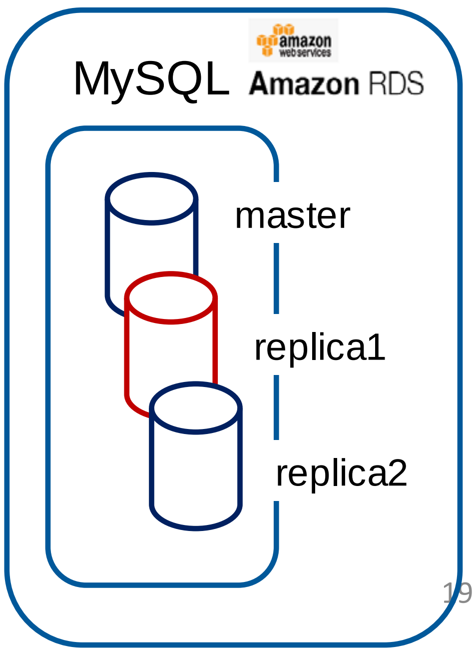


Vertical Scaling





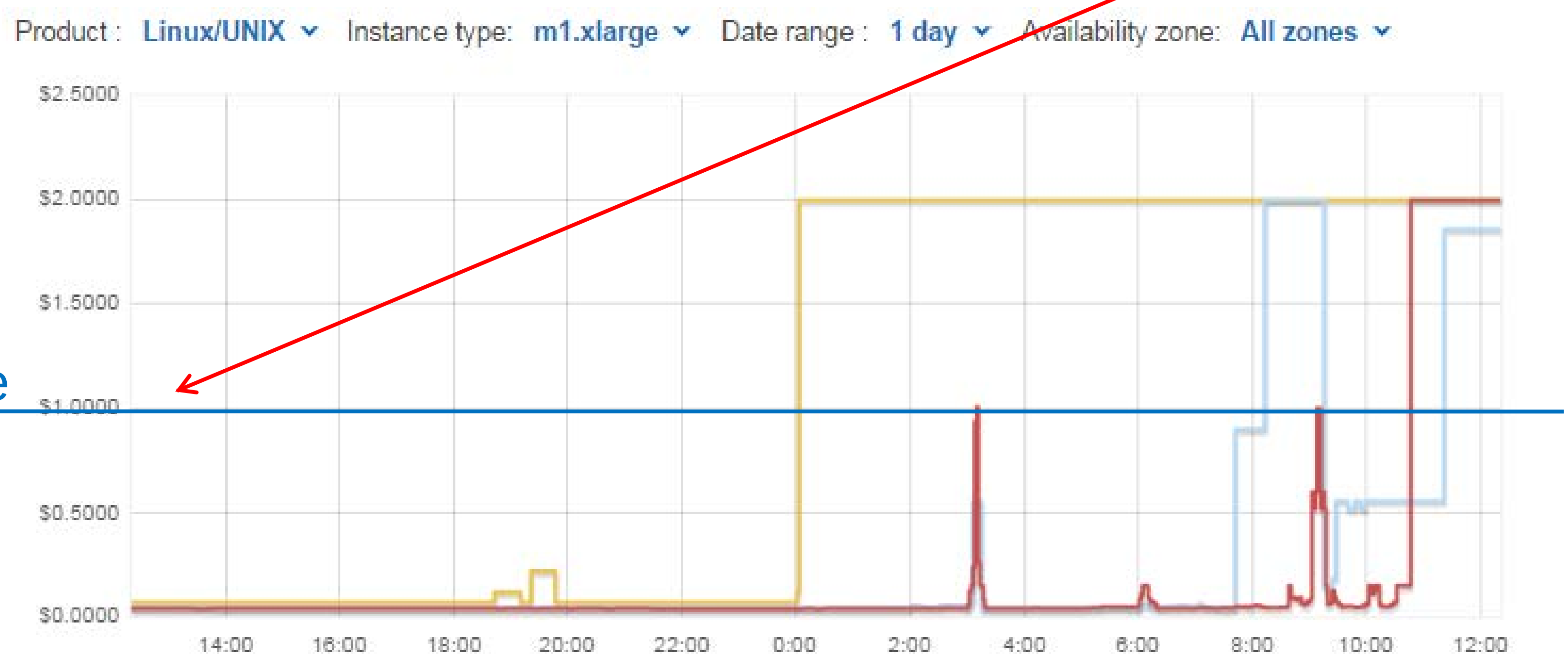
Environment Model



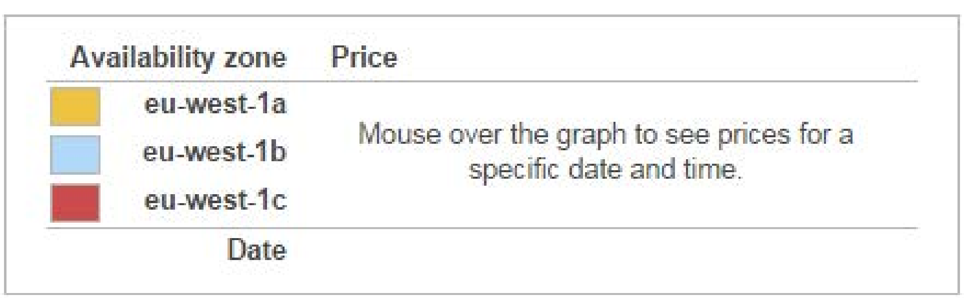
Spot Instances (EC2)

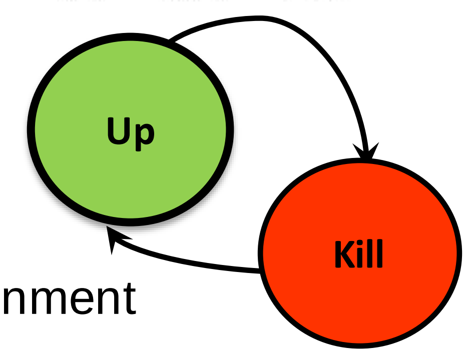
Spot price < Bid

Spot prices



Price
Bid





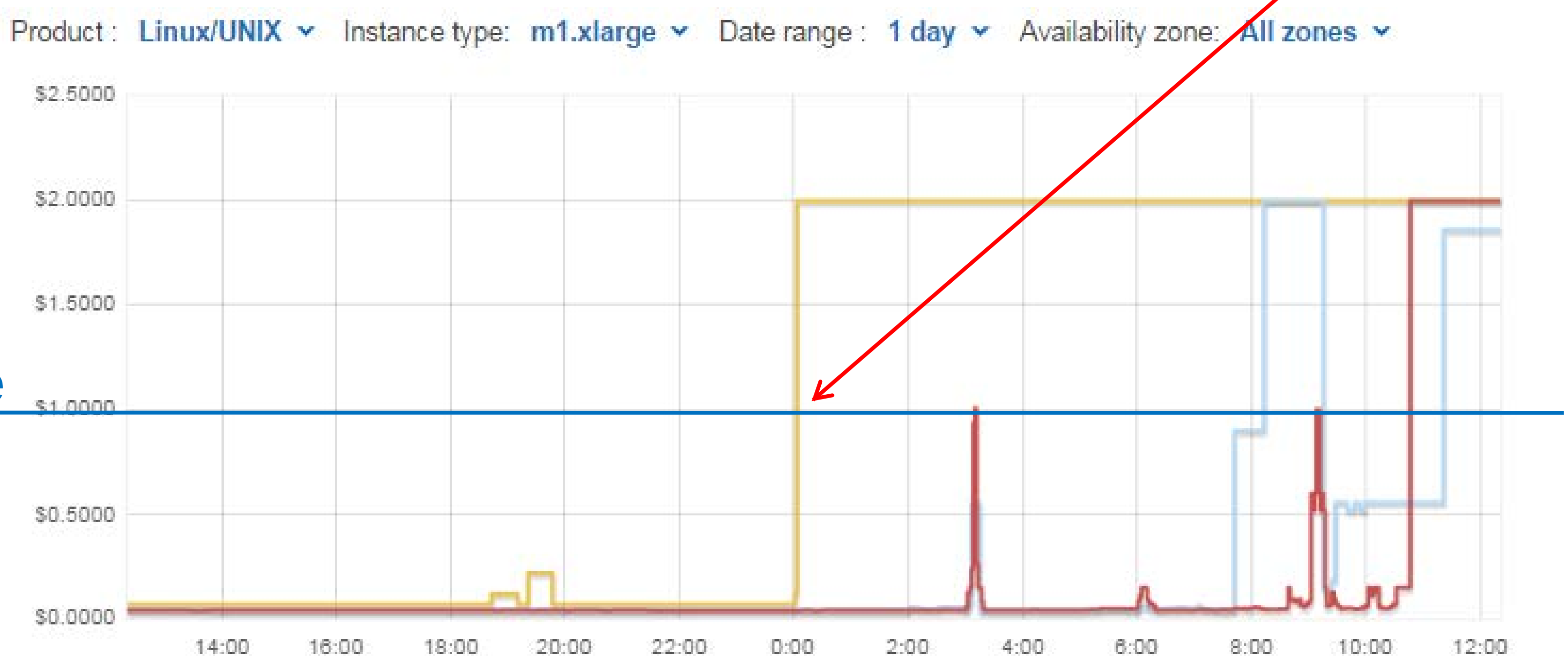
Environment Model

Imperial College London

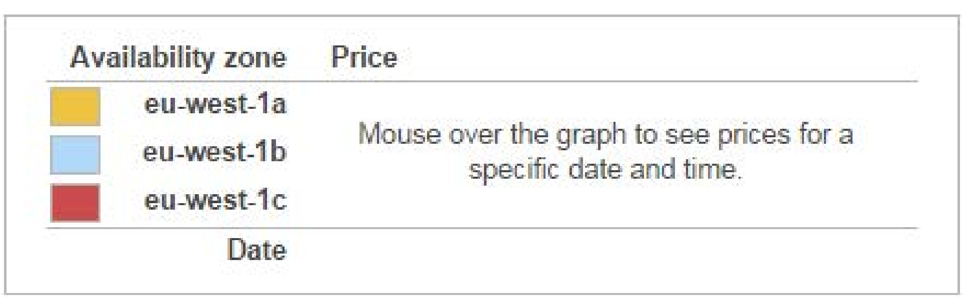
Spot Instances (EC2)

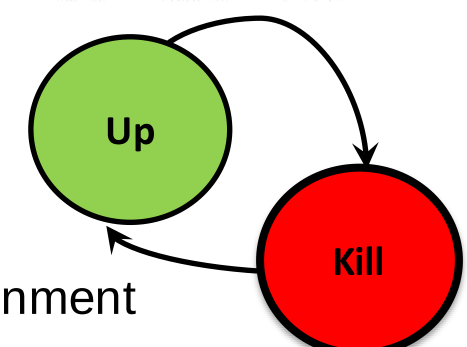
Spot price > Bid
(VM can be reclaimed)

Spot prices



Price
Bid

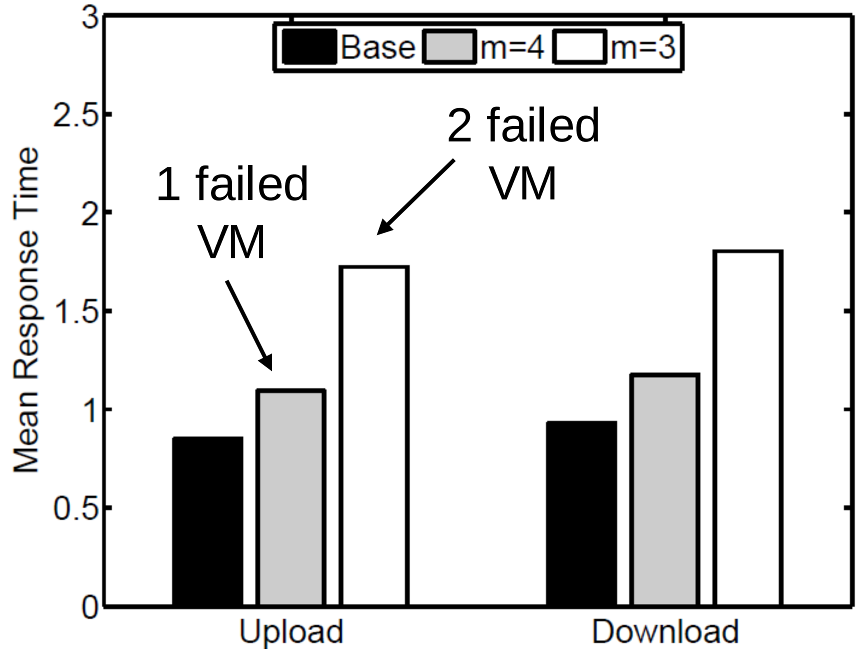


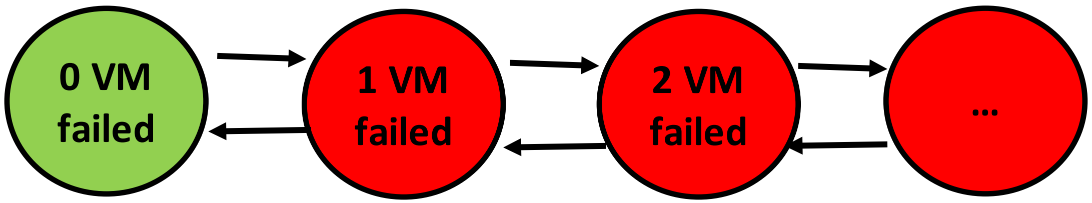


Environment Model

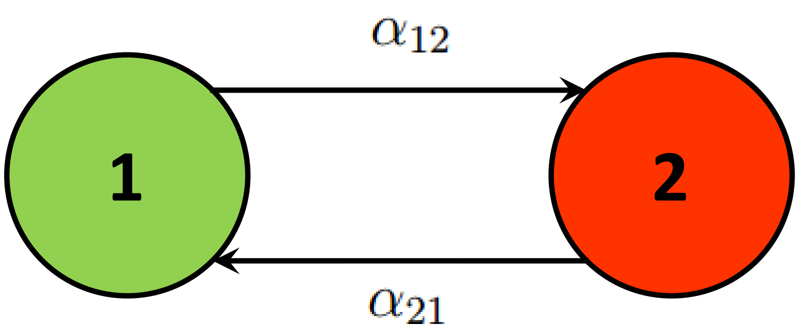
Environment Model: VM Breakdowns

Effect of VM failures **response time**:





Probabilistic QoS Modelling



Continuous-Time Markov chains (CTMCs)

$$\boldsymbol{Q} = \left[\begin{array}{c|c} \boldsymbol{Q}_1^* - \alpha_{12}\boldsymbol{I} & \alpha_{12}\boldsymbol{I} \\ \hline \alpha_{21}\boldsymbol{I} & \boldsymbol{Q}_2^* - \alpha_{21}\boldsymbol{I} \end{array}\right]$$

Exact solution

$$\boldsymbol{\pi}_1 \left(\boldsymbol{Q}^{AVG} - \frac{\boldsymbol{Q}_1^* \boldsymbol{Q}_2^*}{\nu} \right) = \boldsymbol{0} \qquad \boldsymbol{\pi}_2 \left(\boldsymbol{Q}^{AVG} - \frac{\boldsymbol{Q}_2^* \boldsymbol{Q}_1^*}{\nu} \right) = \boldsymbol{0}$$

$$\nu = \alpha_{12} + \alpha_{21}$$

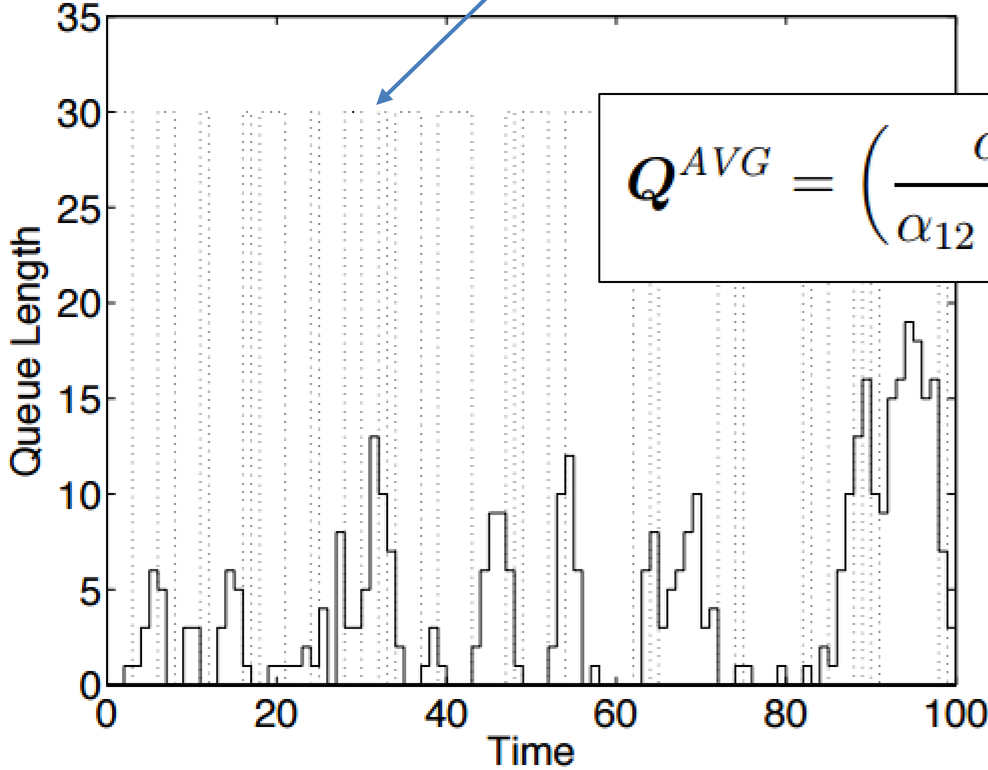
- Direct numerical methods seldom applicable
  - State-space explosion
  - Exponential growth with number of jobs and nodes

Imperial College London

Average Approximation

Tandem closed queueing network (2 queues)

Limit case: frequent environment changes



(a) High ν rate (AVG approx. accurate).

Average approximation

$$\boldsymbol{Q}^{AVG} = \left(\frac{\alpha_{21}}{\alpha_{12}+\alpha_{21}}\right)\boldsymbol{Q}_1^* + \left(\frac{\alpha_{12}}{\alpha_{12}+\alpha_{21}}\right)\boldsymbol{Q}_2^*$$

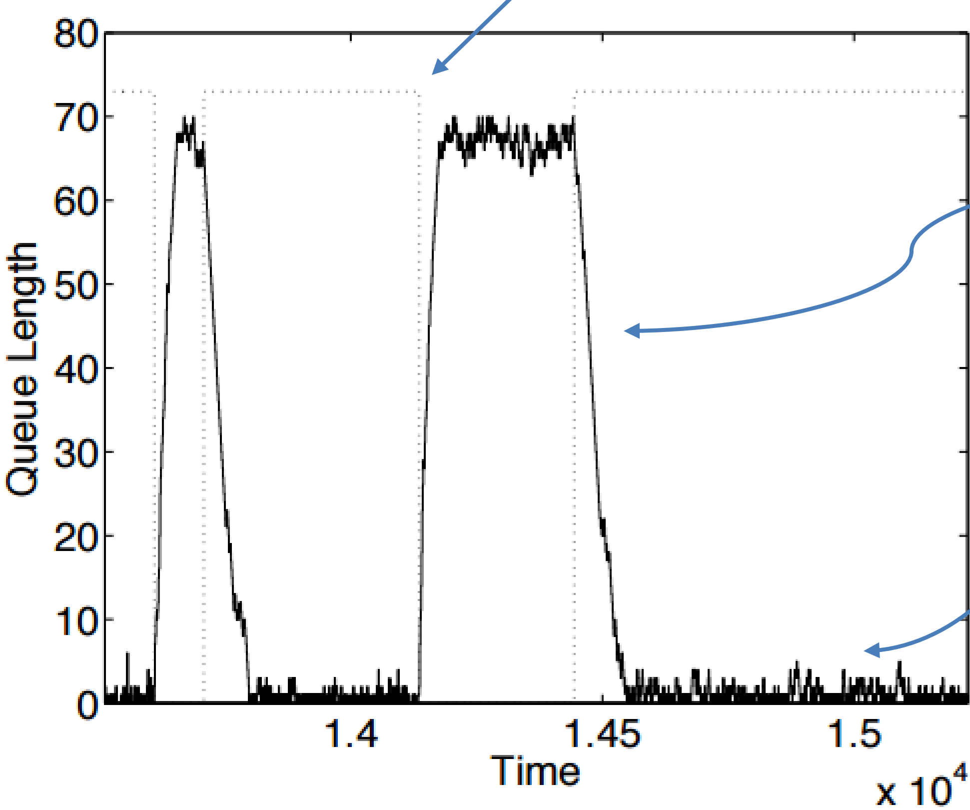
Close to exact for large jump rates

$$\nu = \alpha_{12} + \alpha_{21} \gg 0$$

Decomposition Approximation

Tandem closed queueing network (2 queues)

Limit case: rare environment changes



(c) Low ν rate (DEC approx. accurate).

Decomposition method:

$$\boldsymbol{\pi}_1^{DEC}\boldsymbol{Q}_1^* = \mathbf{0},$$
$$\boldsymbol{\pi}_2^{DEC}\boldsymbol{Q}_2^* = \mathbf{0},$$

Results are weighted by mean time in each environment state

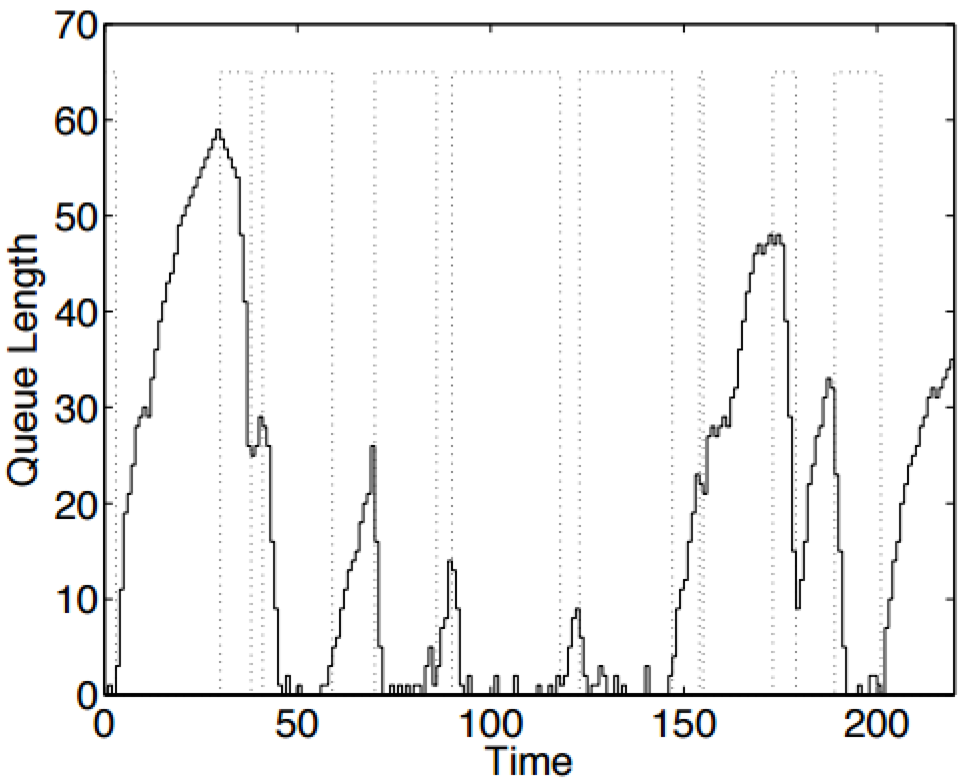
Close to exact for small jump rates

$$\nu \approx 0$$

Intermediate case

Tandem closed queueing network (2 queues)

Intermediate case



(b) Intermediate ν rate (no accurate method).

Only direct solution methods available (unscalable)

Contribution: LINE

- Accurate and efficient analysis of time-varying environments

LINE Solver

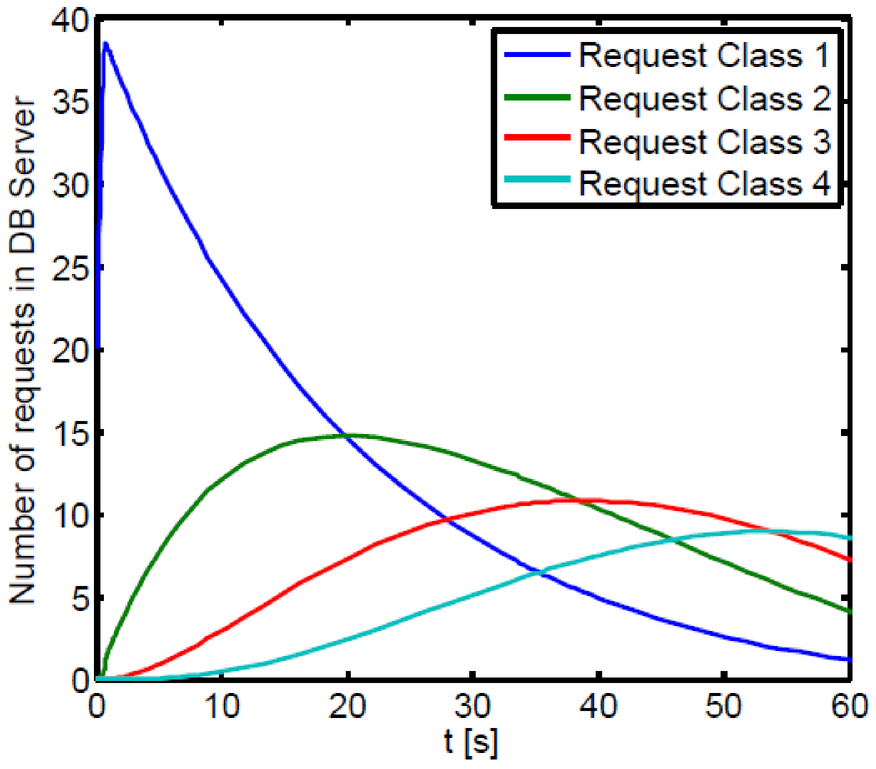
- **Fluid solver** for QNs in random environments
  - Fast solution by **ordinary differential equations**
- Goes beyond mean values used in canonical solvers:
  - Transient analysis
  - Response time percentiles (e.g. SLA assessment)
  - Handling of multi-server resources (e.g., vCPUs)
  - Synchronous and asynchronous calls
  - Non-exponential service distributions (Coxian)

URL: http://line-solver.sf.net

Imperial College London

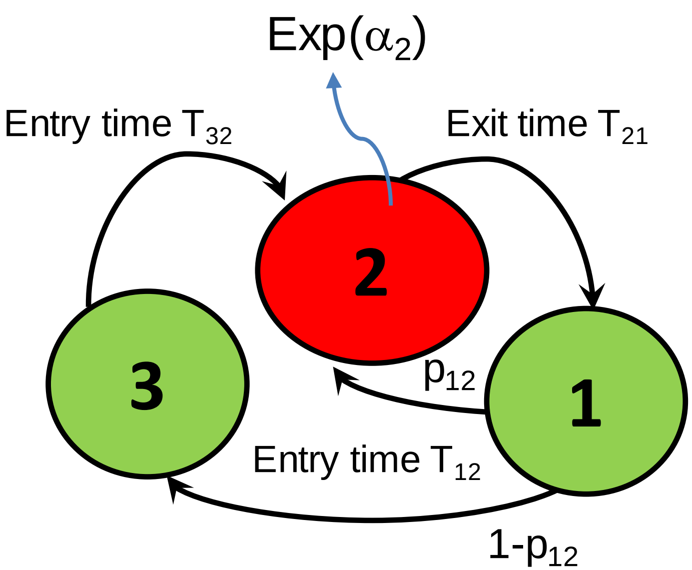
Fluid Modelling

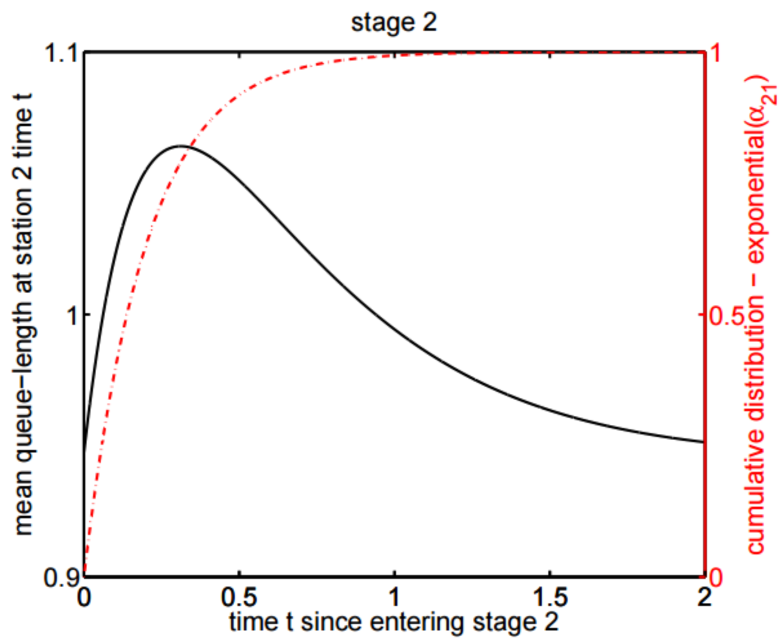
- Non-linear ODEs provide time-varying mean queue sizes
- CTMC solution converges to ODE as system size grows



Blending Method

- Leverages time-varying solutions to analyze systems operating in random environments





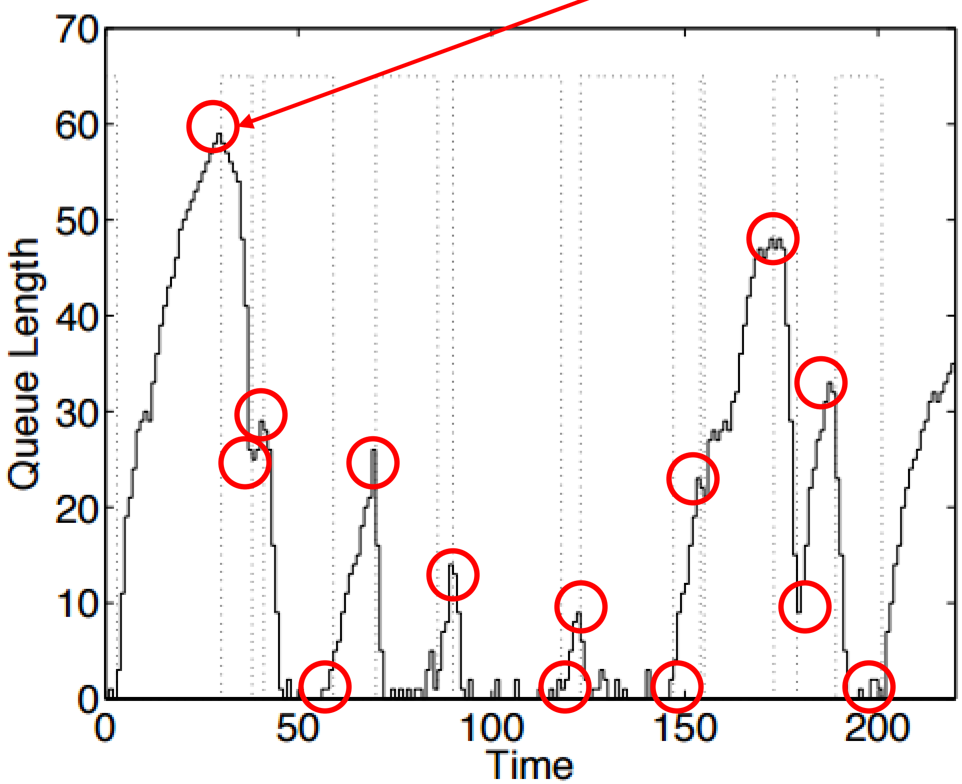
- **Main result**: found dynamic equilibrium between entry and exit time distributions
  - Fixed point iteration involving their Laplace transforms

Blending Method

State of node i seen at jump instants:

Entry state

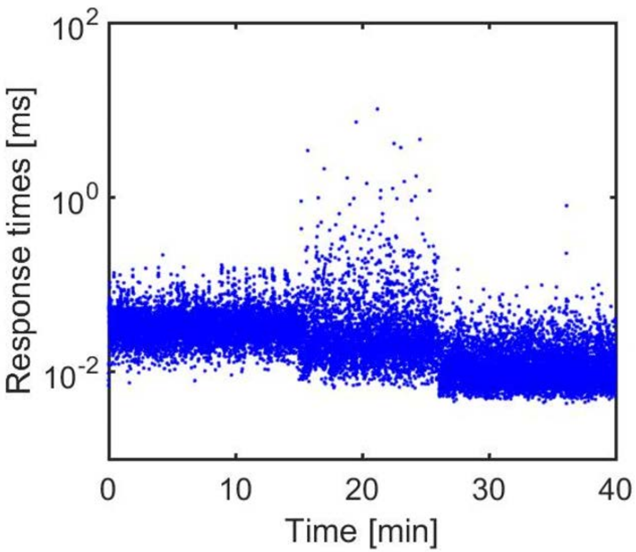
$$\bar{n}_i^e(0) = \sum_{\substack{h=1 \\ h \neq e}}^{E} p_{he}^{src} \alpha_h \left(\int_0^{+\infty} \underbrace{\bar{n}_i^h(t) e^{-\alpha_h t}}_{\text{Transient state computed by ODEs}} dt \right)$$

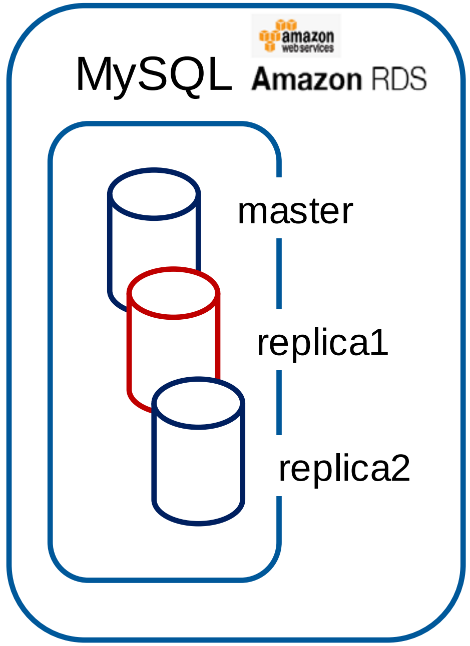


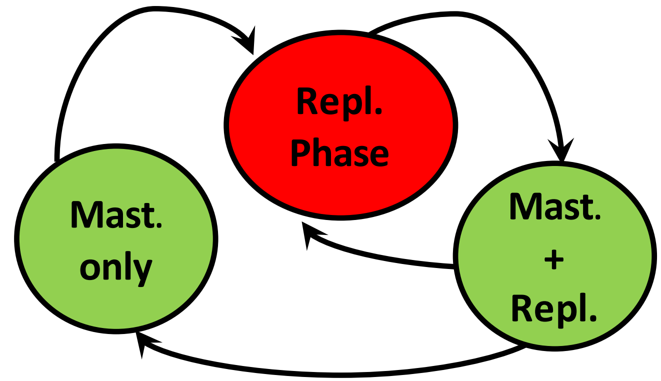
(b) Intermediate ν rate (no accurate method).

Entry states at equilibrium determined by fixed-point iteration

Example: Scaling in Amazon RDS







Environment Model

Avg. response time	Before Repl.	During Repl.	After Repl.
Measurement (s)	0.062	0.063	0.034
Prediction (s)	0.059	0.060	0.031
Error %	4.2%	4.4%	8.9%

Imperial College London

Application: Sizing in the Cloud

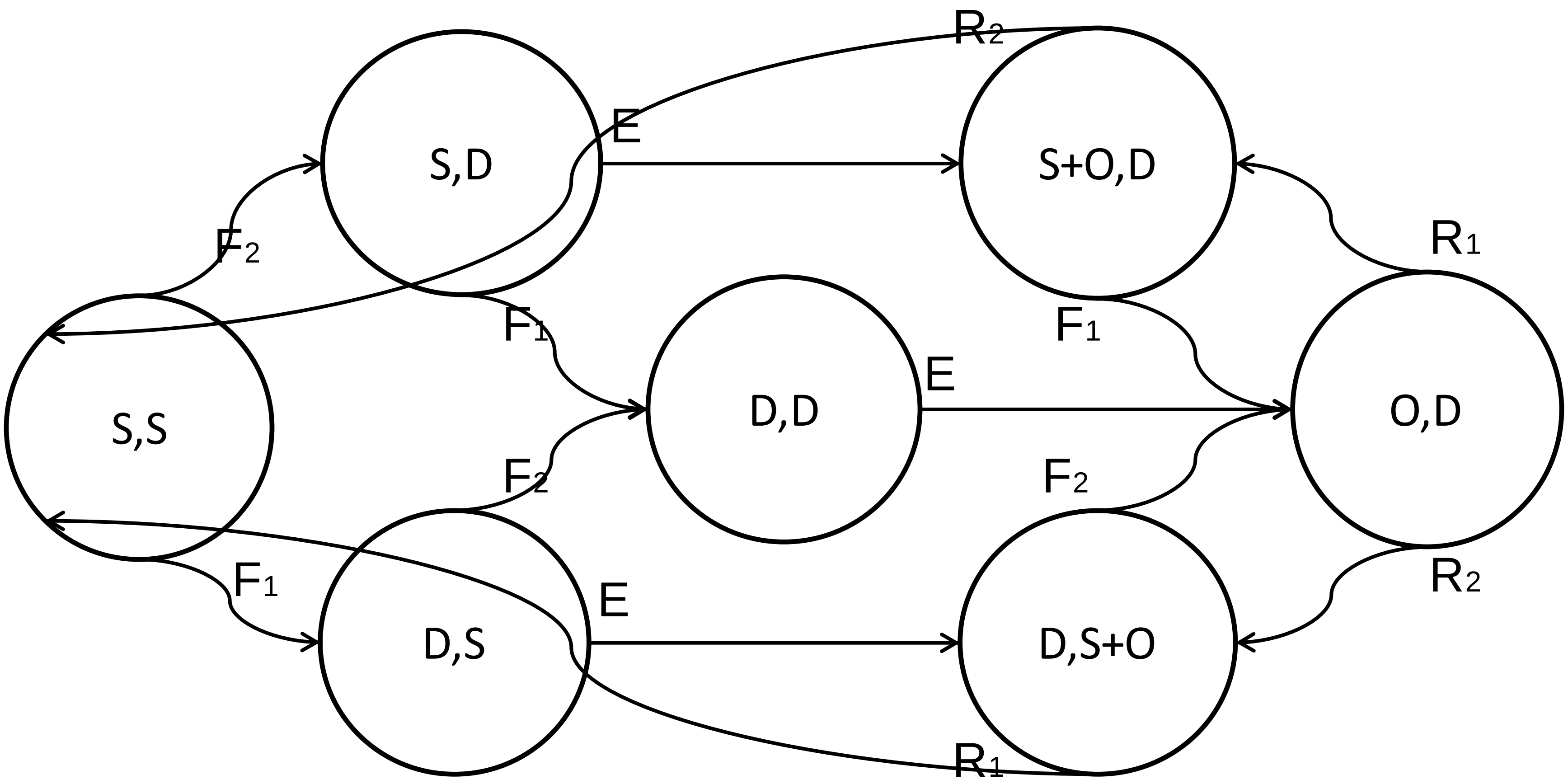
- Provisioning of spot and on-demand resources

$m^S_{(i,j)}$	Number of spot instances in cloud i for component j
$m^O_{(i,j)}$	Number of on-demand instances in cloud i for component j

- Provisioning based on mean:
  - Solution based on simulated annealing

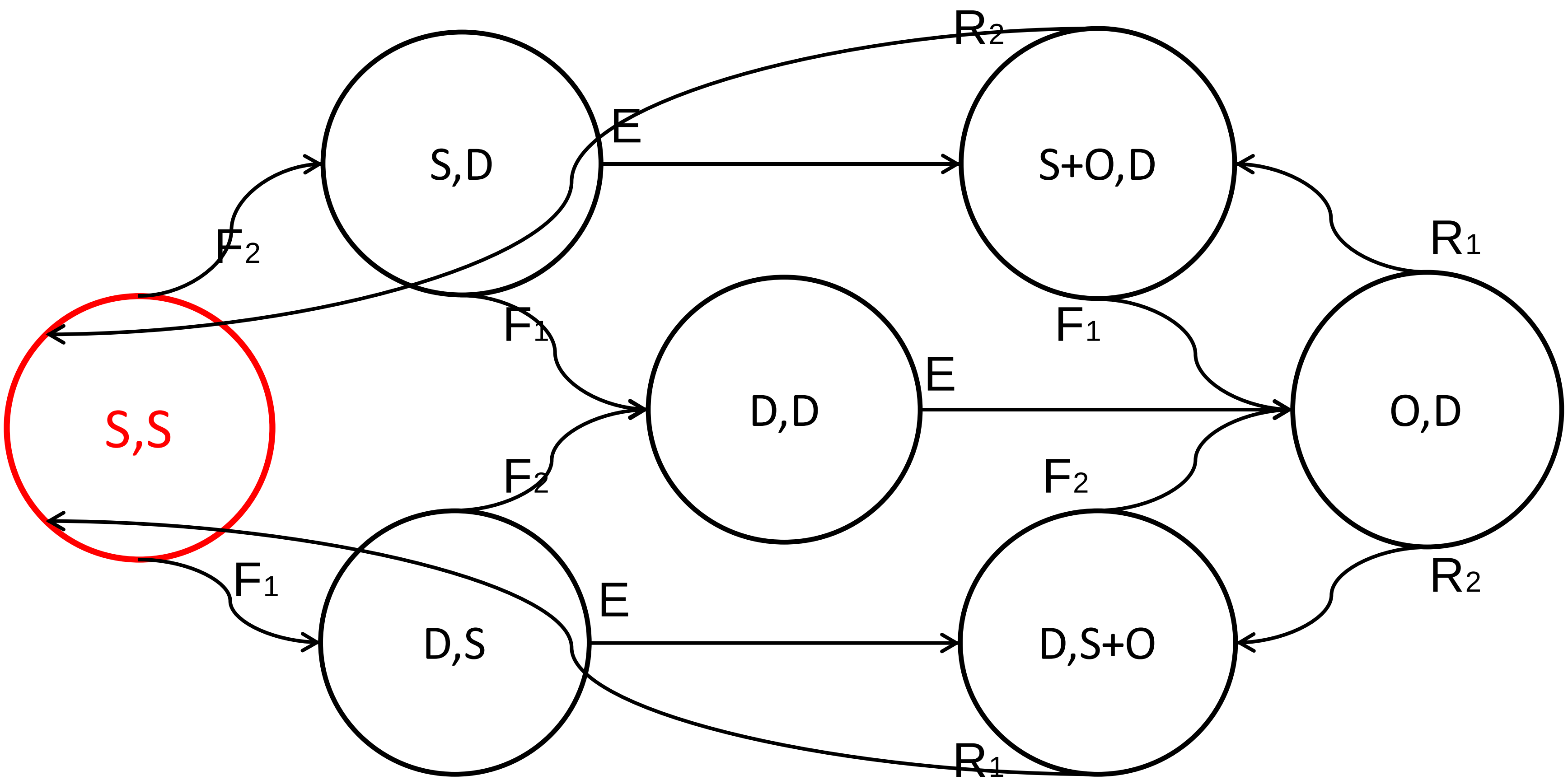
$$
\begin{aligned}
\min \quad & \sum_{j=1}^{2} \left(\sum_{i=1}^{2} C_i^S m^S_{(i,j)} + C^O m^O_{(i,j)} \right) \\
\text{s.t.} \quad & E[R] = \textsc{Line}(m^S, m^O), \\
& E[R] \le R^{\max}, \\
& m^S_{(i,j)} \in \mathbb{N}_+, \ i = 1,2, \ j = 1,2, \\
& m^O_{(i,j)} \in \mathbb{N}_+, \ j = 1,2,
\end{aligned}
$$

Spot VM Reliability Model



Spot VM Reliability Model

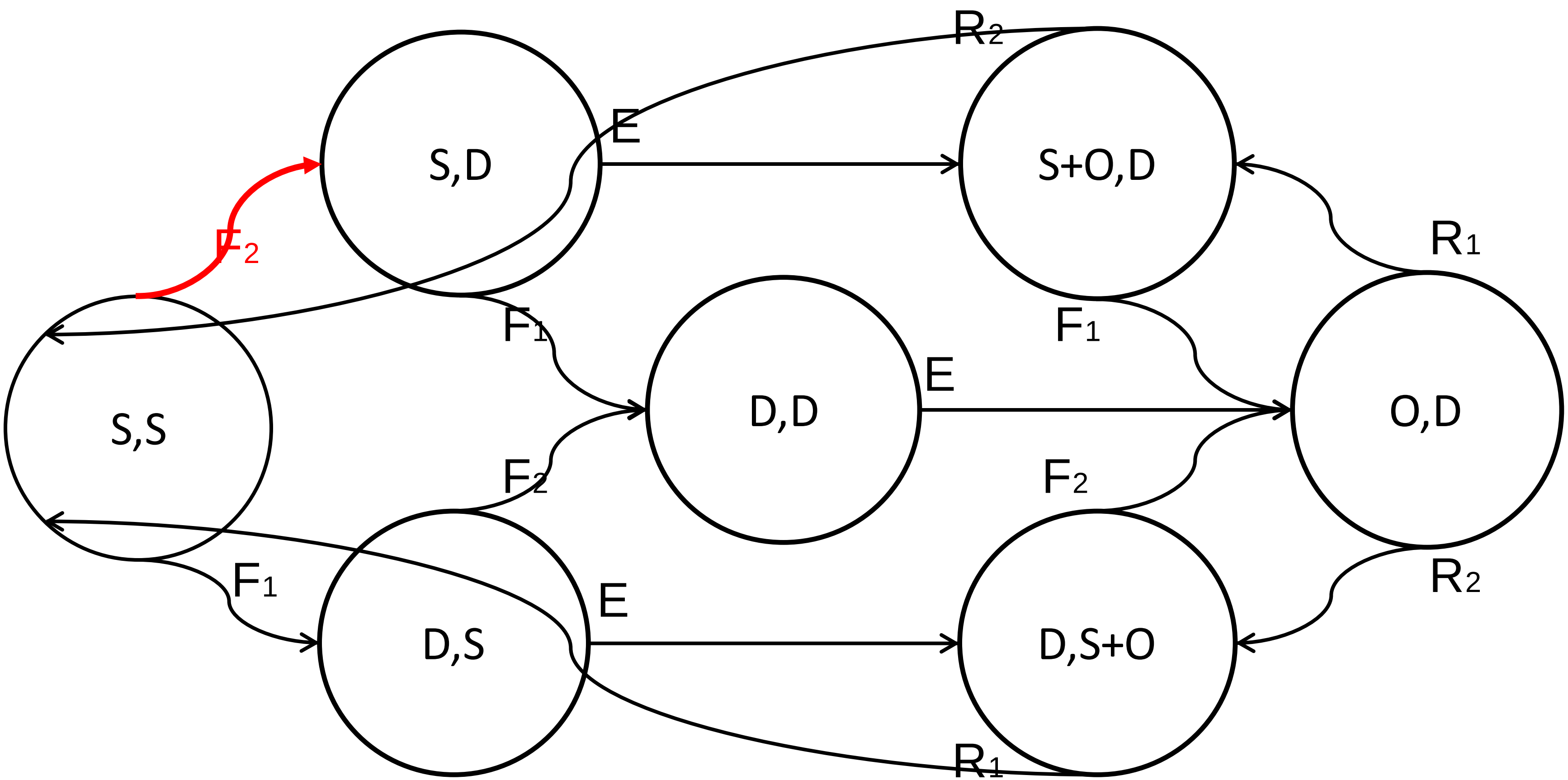
2 spot instances in 2 clouds



Imperial College London

Spot VM Reliability Model

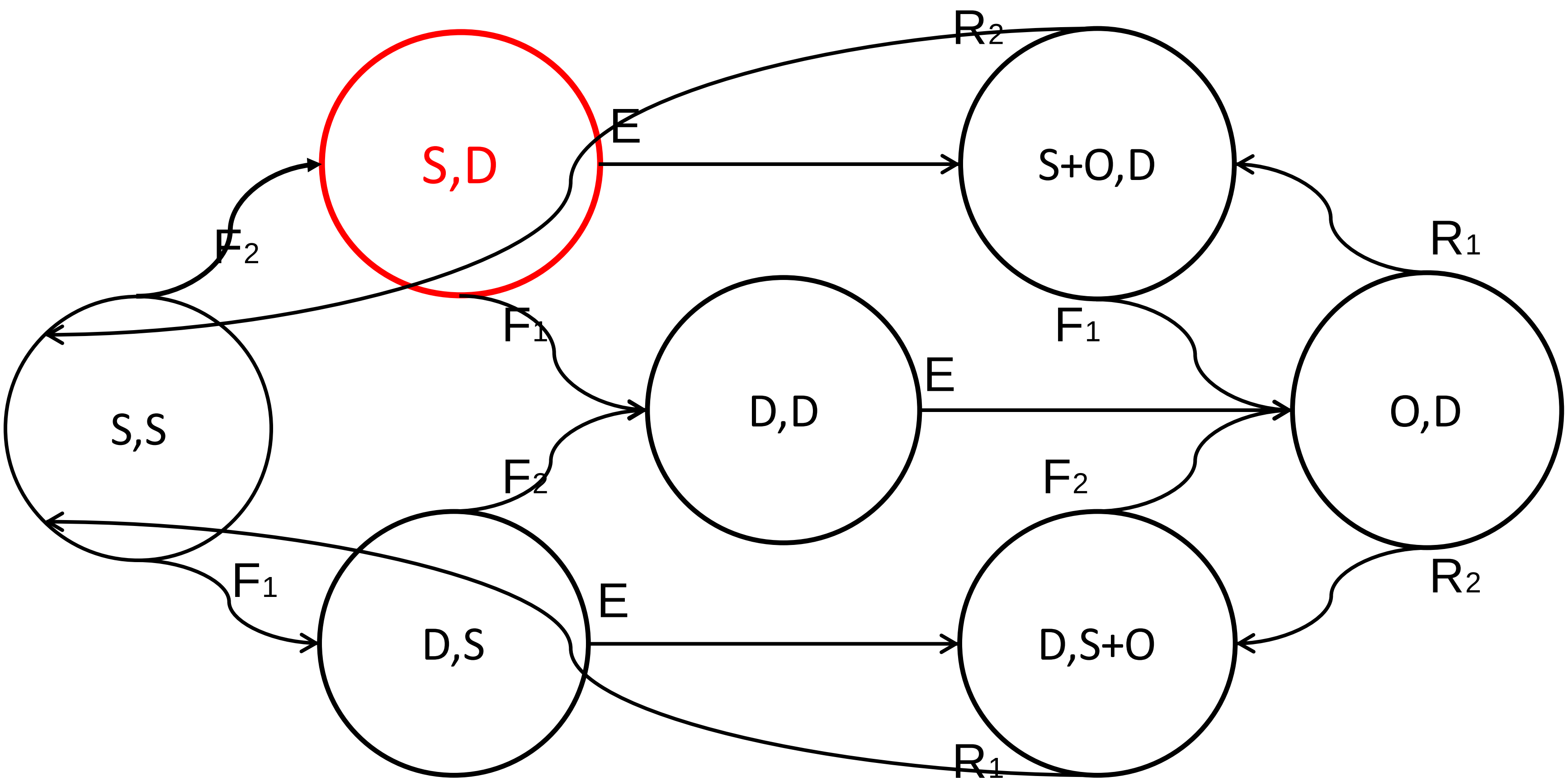
Spot instances in cloud 2 are lost



Imperial College London

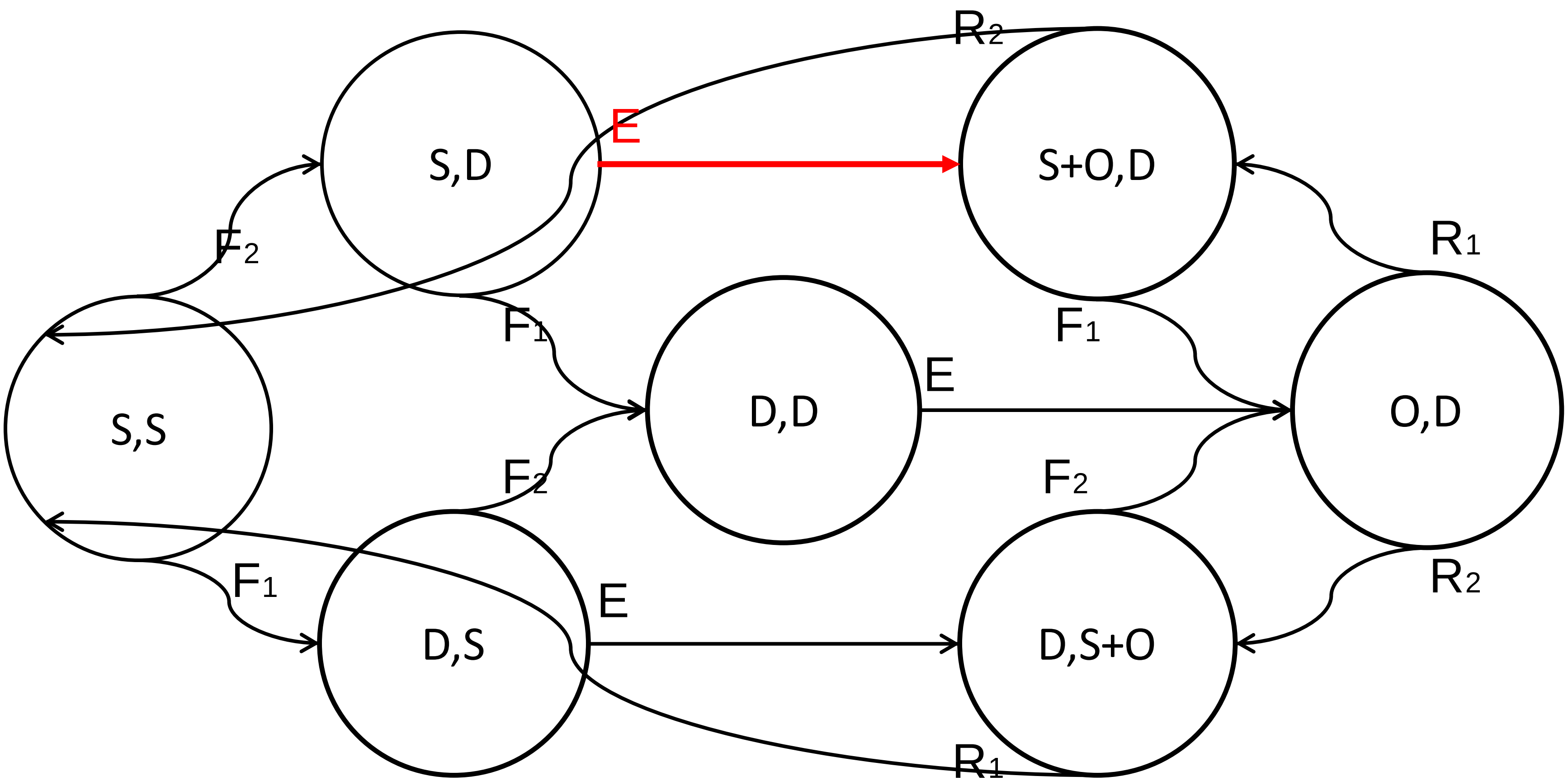
Spot VM Reliability Model

Spot instances in cloud 2 are lost



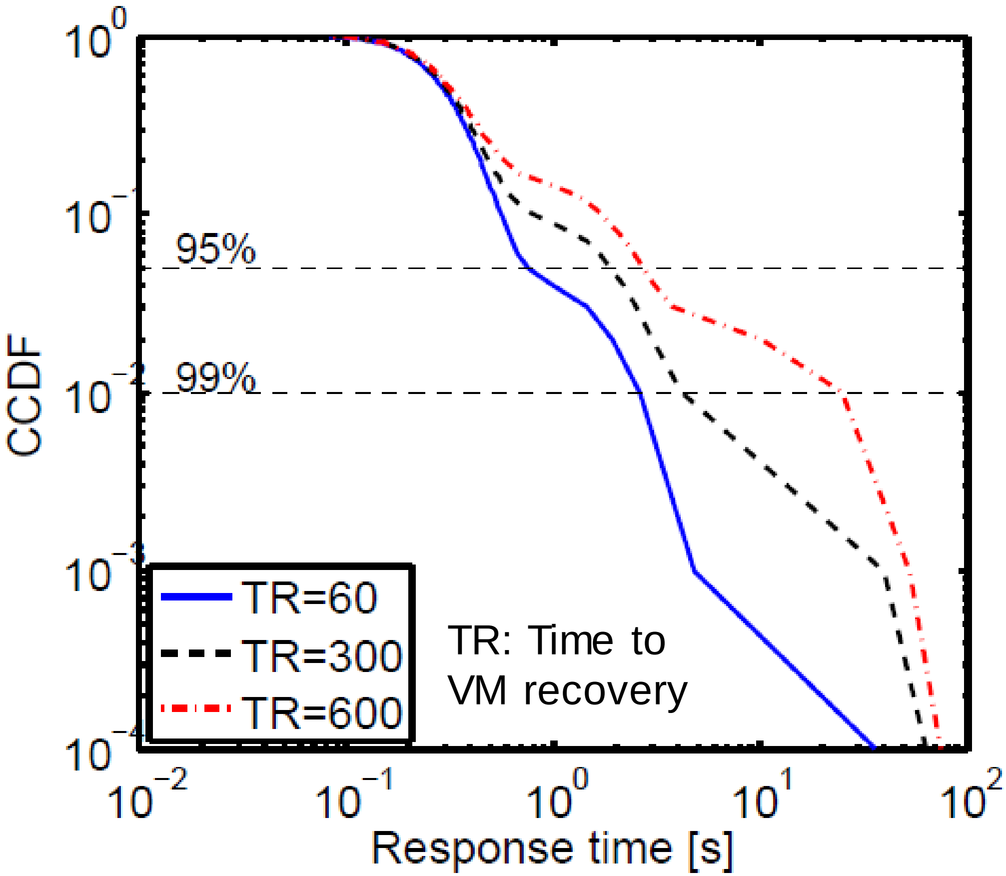
Spot VM Reliability Model

"Emergency" on-demand instances started on cloud 1



Spot VM Reliability Model

- Response time sensitivity to time to recover a spot VM



- Provisioning based on RT percentiles:

$$R_x = \textsc{Line}(m^S, m^O)$$

$$R_x \leq R_x^{\max}$$

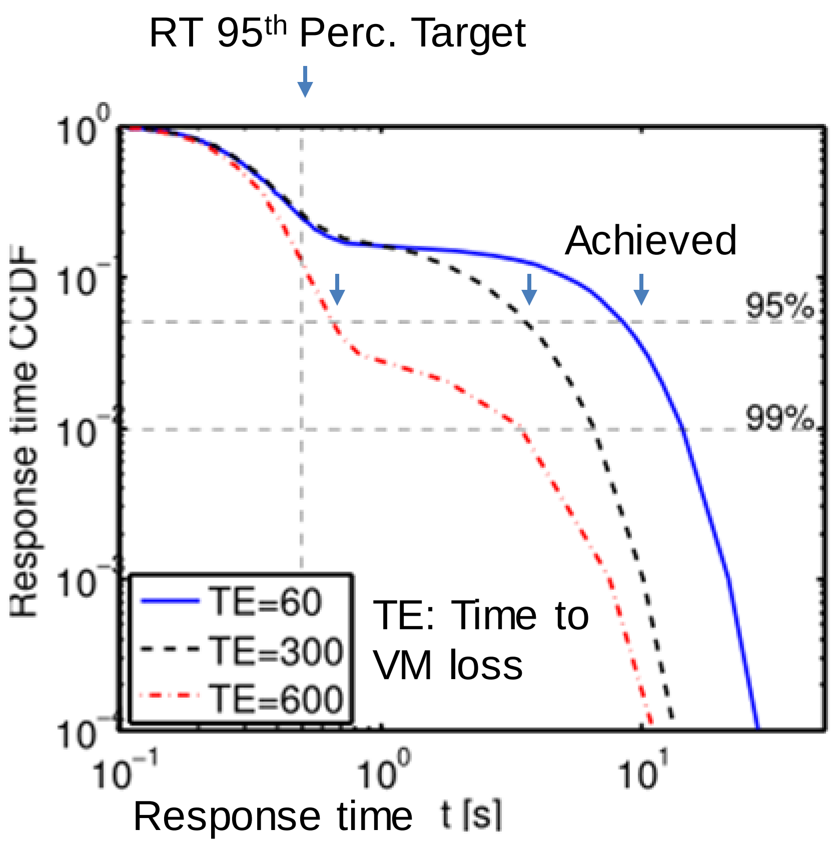
Response time percentile

- 1-2 orders of magnitude better than Markov ineq. & Chevyshev ineq.

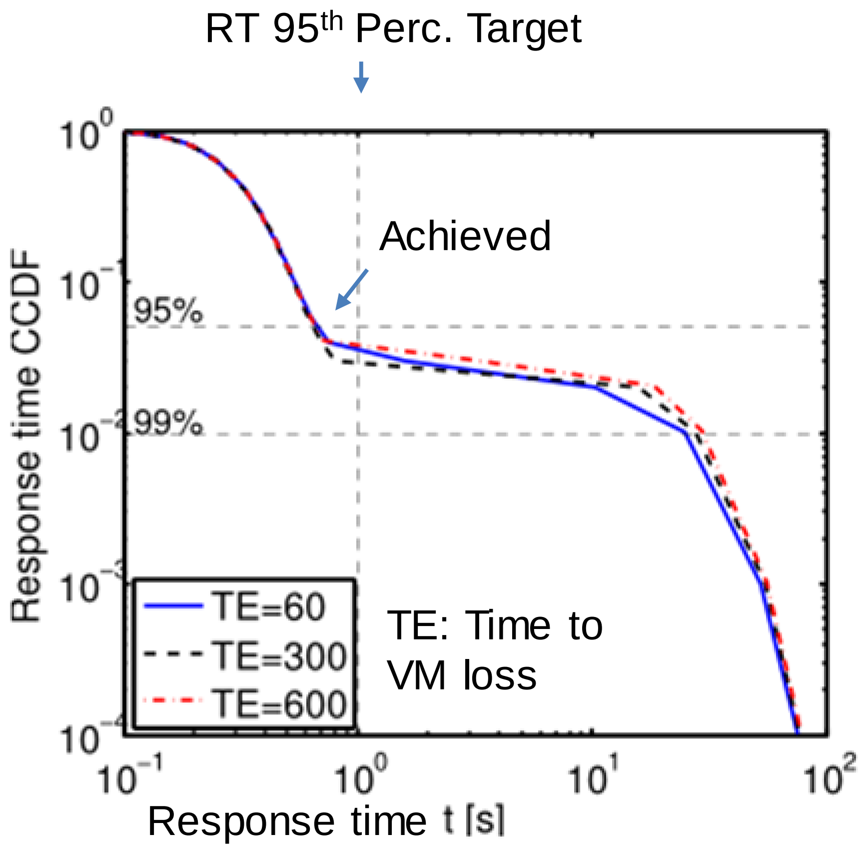
Imperial College London

Spot VM Reliability Model

- Response time sensitivity to time to recover a spot VM



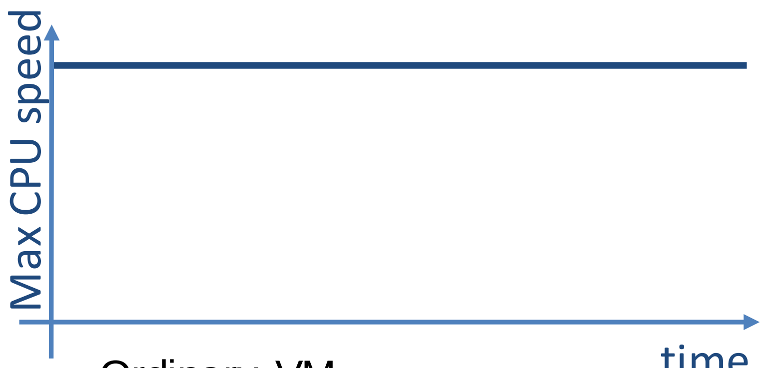
Provisioning based on mean



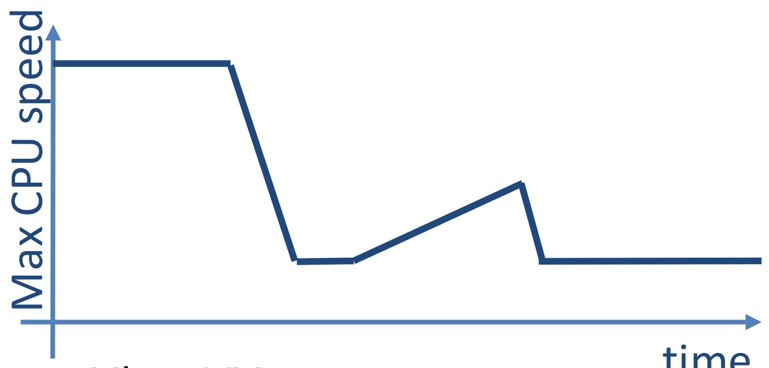
Provisioning based on 95th-percentile

Other Applications of LINE

- Optimal bidding prince for spot VMs (*Cluster Comp.*, 2016)
- Optimal cloud application refactoring (*IEEE CLOUD*, 2016)
- Inference of resource capacities in the cloud (*IEEE TSE*, 2014)
- Ongoing work: provisioning of micro-VMs
    - Environments can help model burstable performance



Ordinary VM



Micro VM

Imperial College London

Questions?

g.casale@imperial.ac.uk