Model Based Control for Multi-Cloud Applications

May 18th, 2013

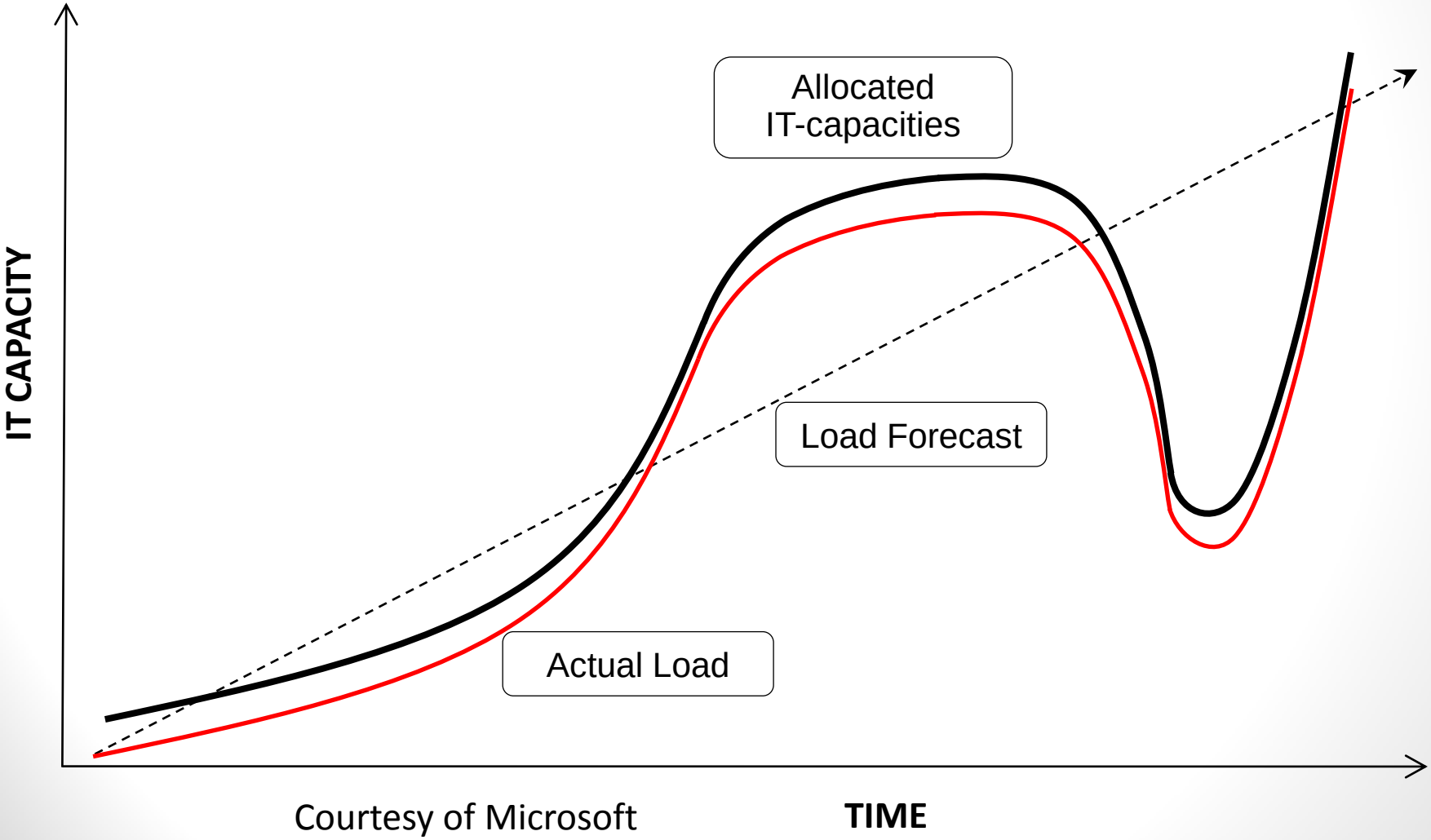
Authors:

Marco Miglierina
Giovanni P. Gibilisco
Danilo Ardagna
Elisabetta Di Nitto

Speaker:

Giovanni P. Gibilisco

Cloud and elasticity



Quality of service on the Cloud

- No native mechanisms to guarantee the Quality of Service required by specific application domains
- Claims: 99.95% of availability (Amazon, Azure)
- Actual observations[1]:
  - From users' perspective:

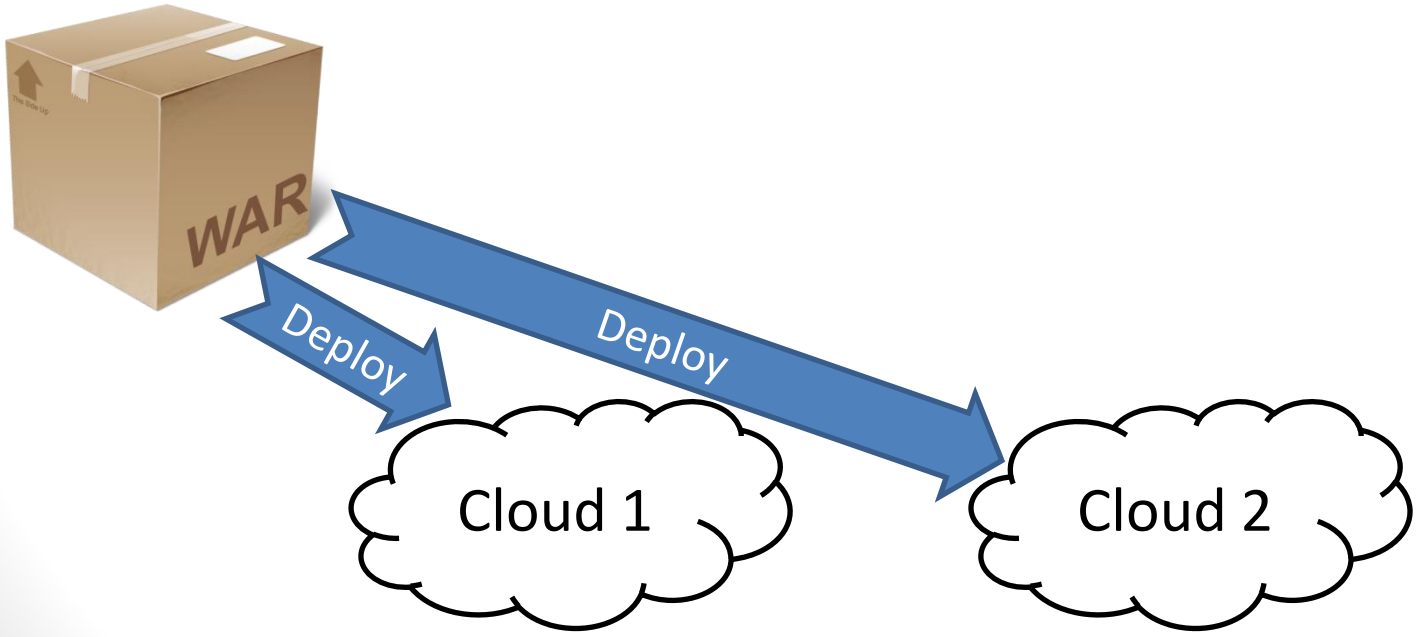
Provider	Availability
EC2 EU	96.32%
Google App Engine	93.05%
Windows Azure	95.39%

  - Outages: Amazon[2] (Apr 2011) , Google[3] (May 2011), Azure[4] (Feb 2012)

1. Bitcurrent, "Cloud Performance from the End User", http://www.bitcurrent.com/, Tech. Rep., 2011.
2. http://aws.amazon.com/message/65648/
3. http://gmailblog.blogspot.it/2011/02/gmail-back-soon-for-everyone.html
4. http://blogs.msdn.com/b/windowsazure/archive/2012/03/01/windows-azure-service-disruption-update.aspx

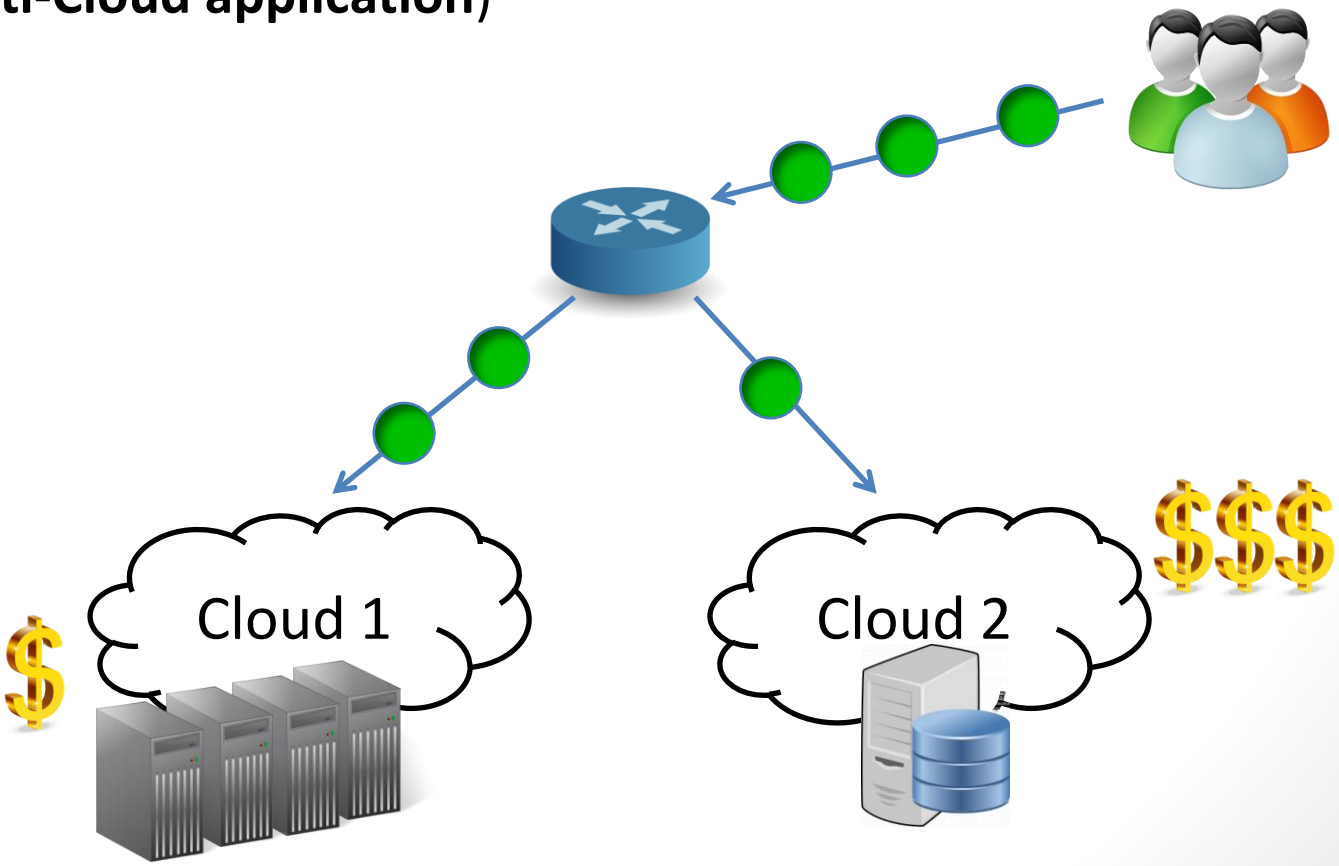
Goal

- High availability is usually obtained by **replication** of critical components
- Solution: exploit two or more Clouds as replication method (**multi-Cloud application**)



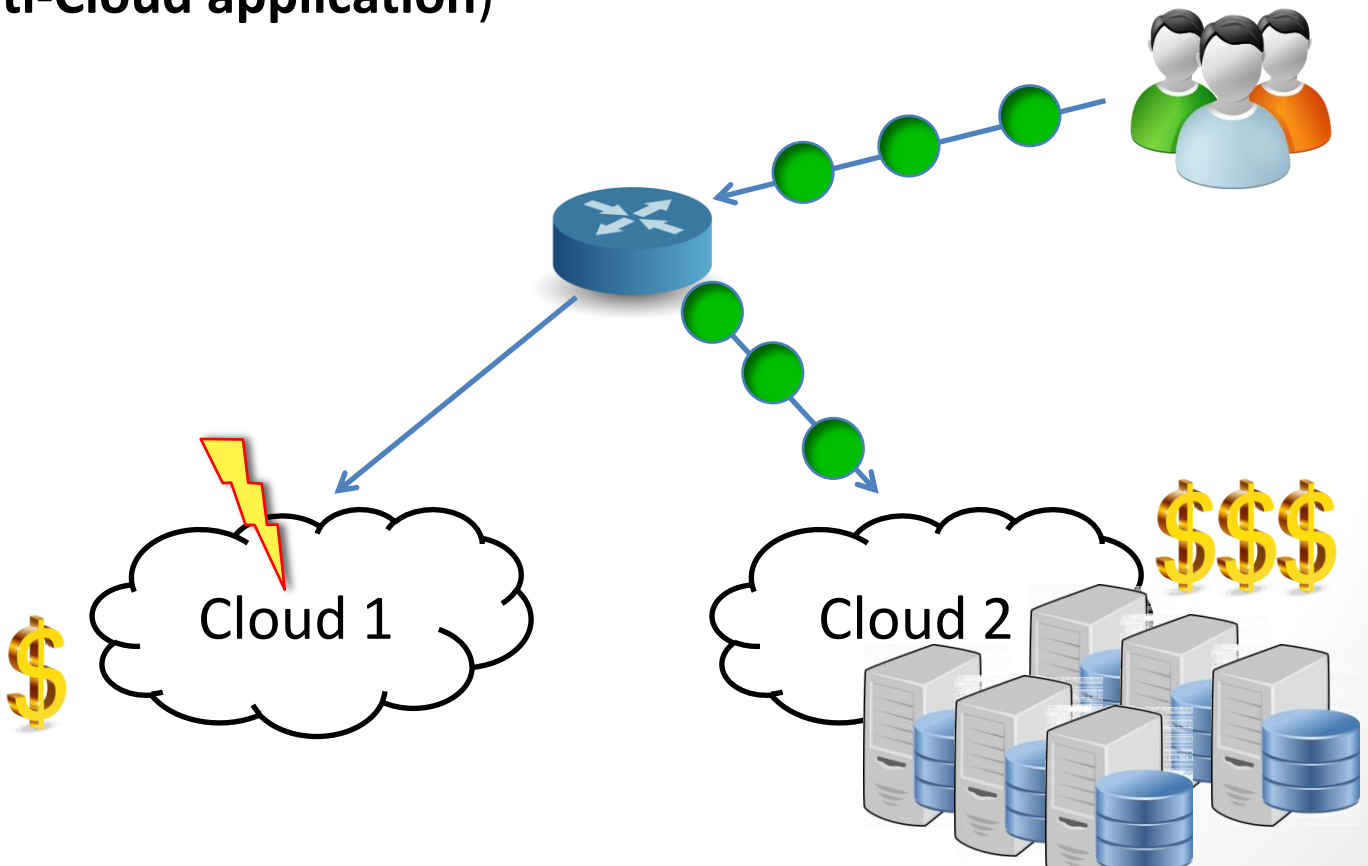
Goal

- High availability is usually obtained by **replication** of critical components
- Solution: exploit two or more Clouds as replication method (**multi-Cloud application**)



Goal

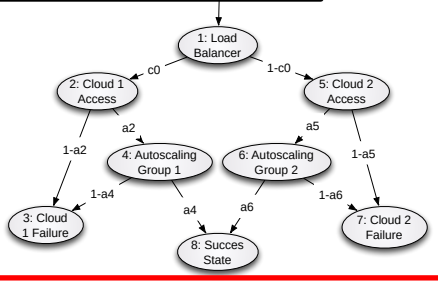
- High availability is usually obtained by **replication** of critical components
- Solution: exploit two or more Clouds as replication method (**multi-Cloud application**)



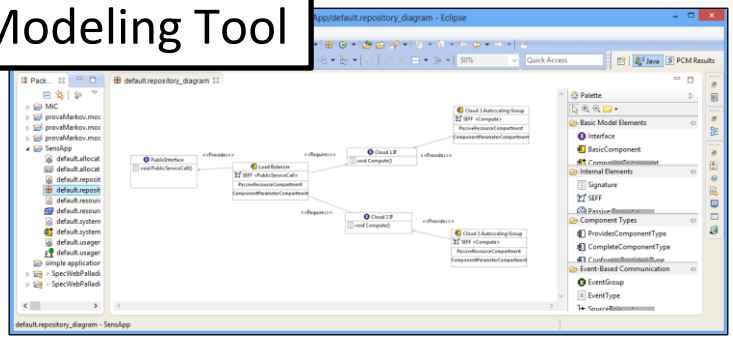
Our solution

Design Time

Multi-Cloud Model

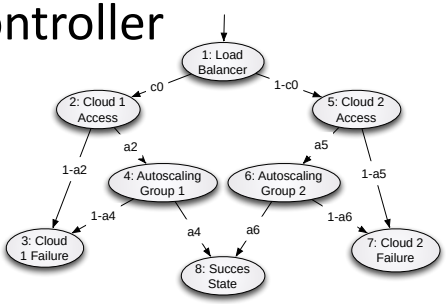


Modeling Tool



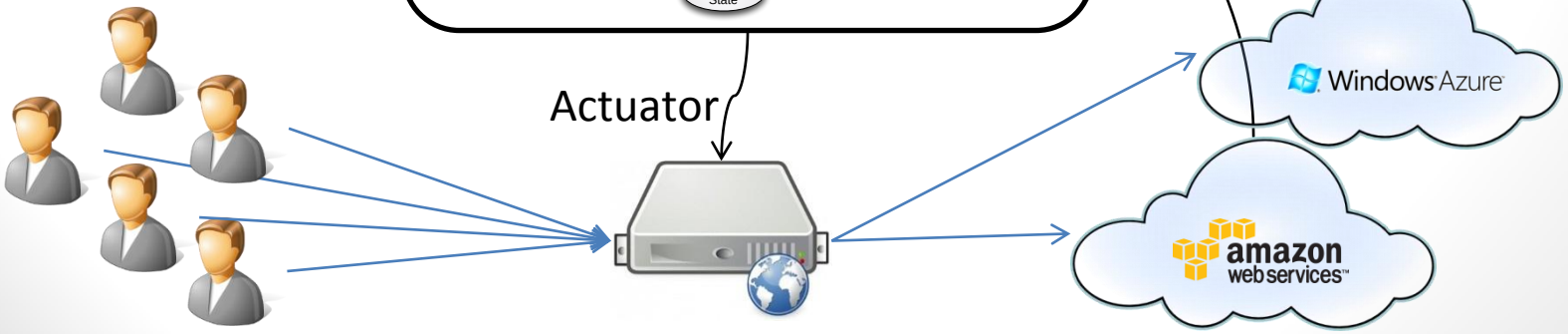
Run Time

Controller

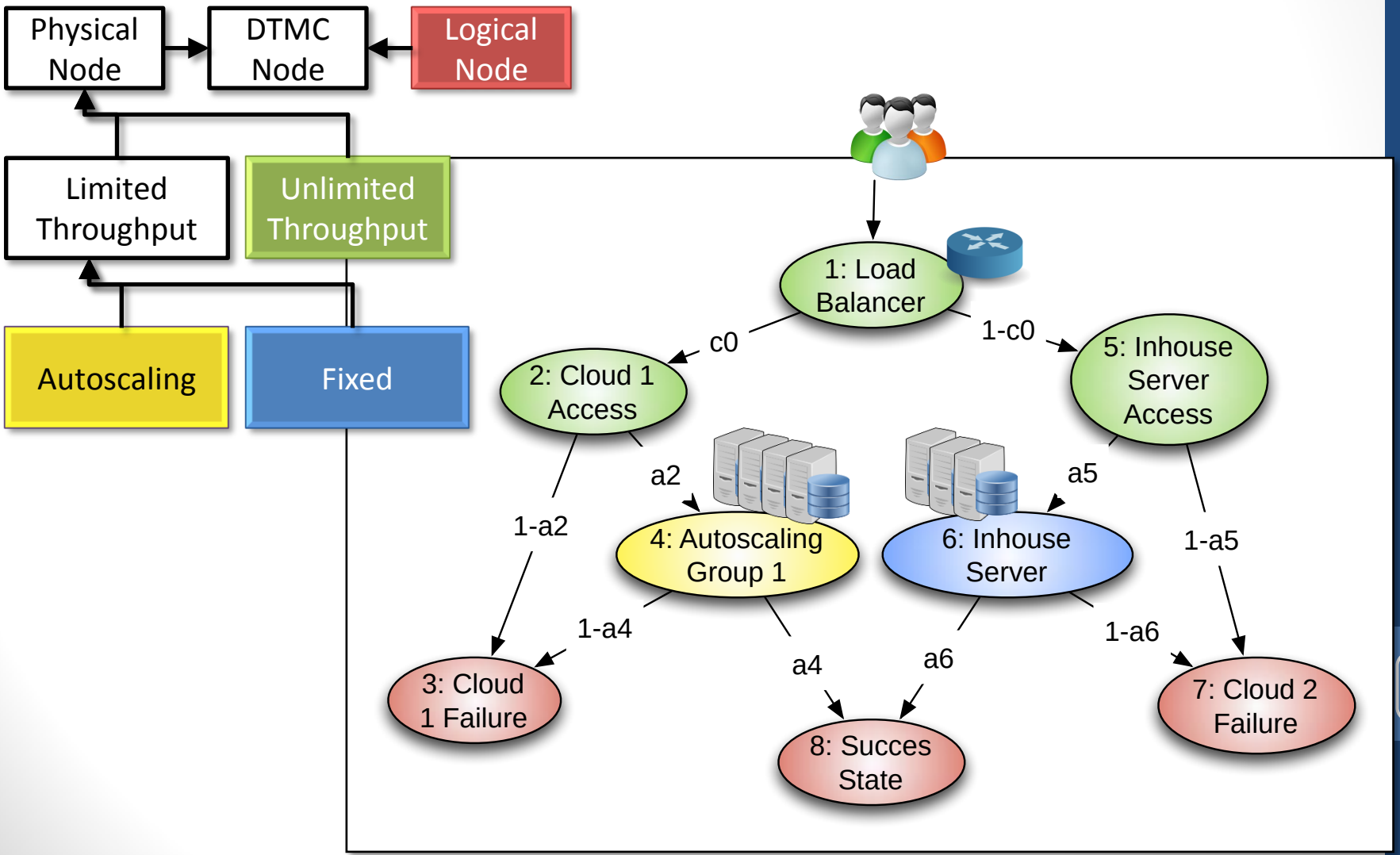


Monitors

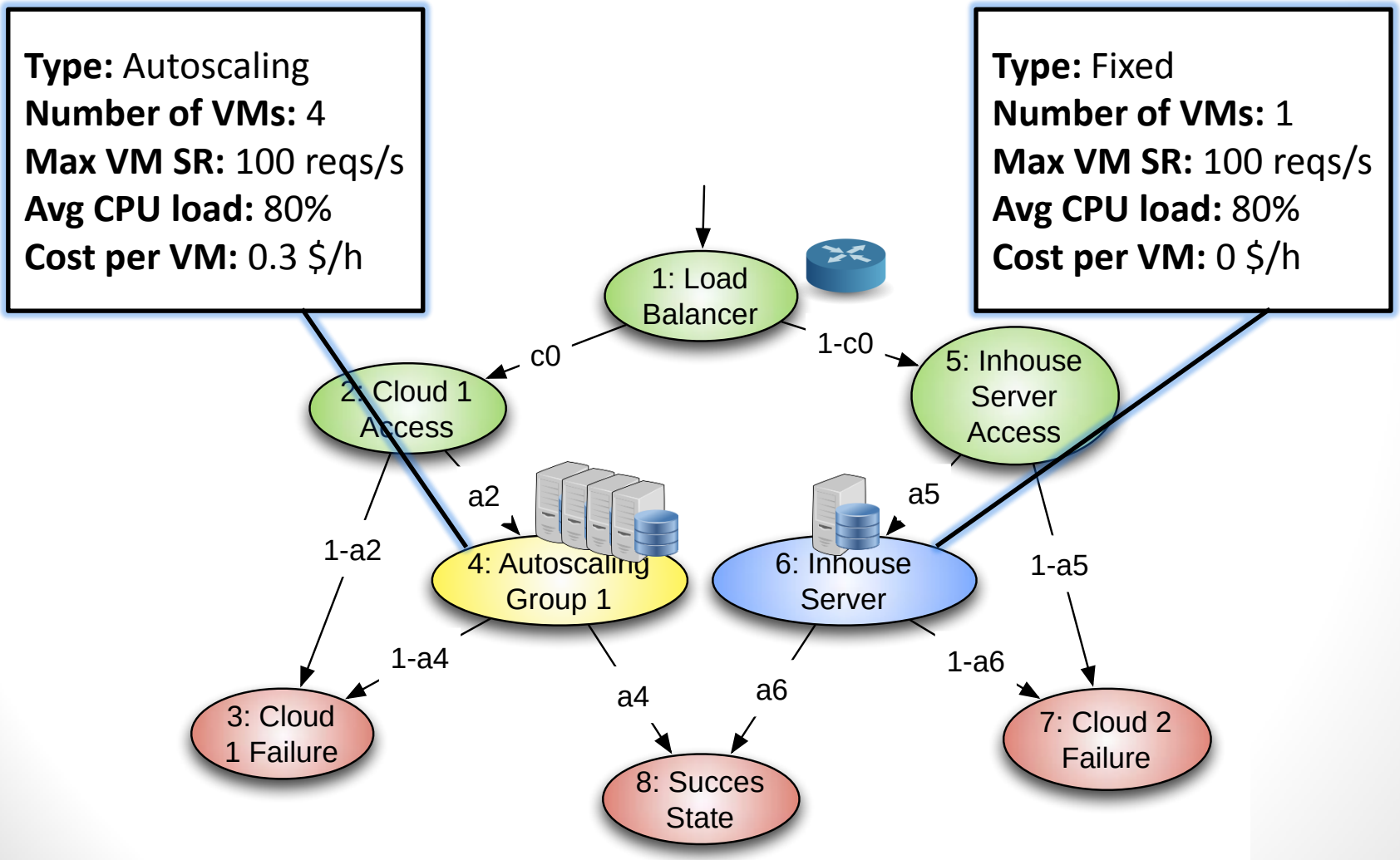
Actuator



Modeling multi-Cloud applications



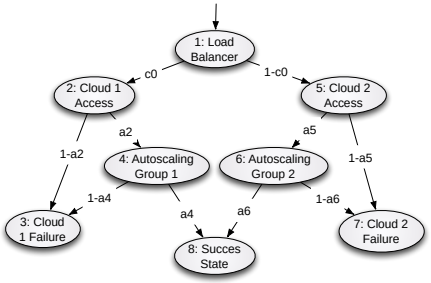
Modeling multi-Cloud applications



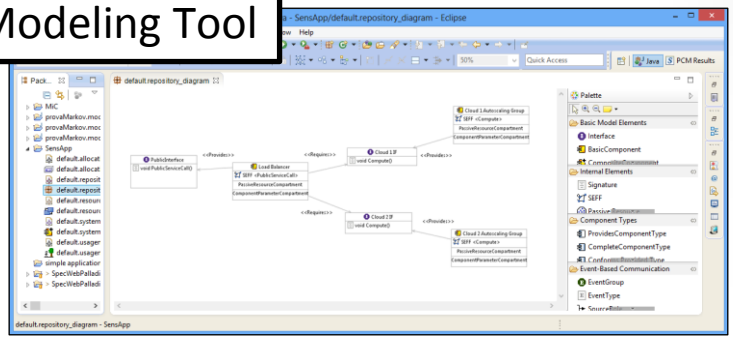
Our solution

Design Time

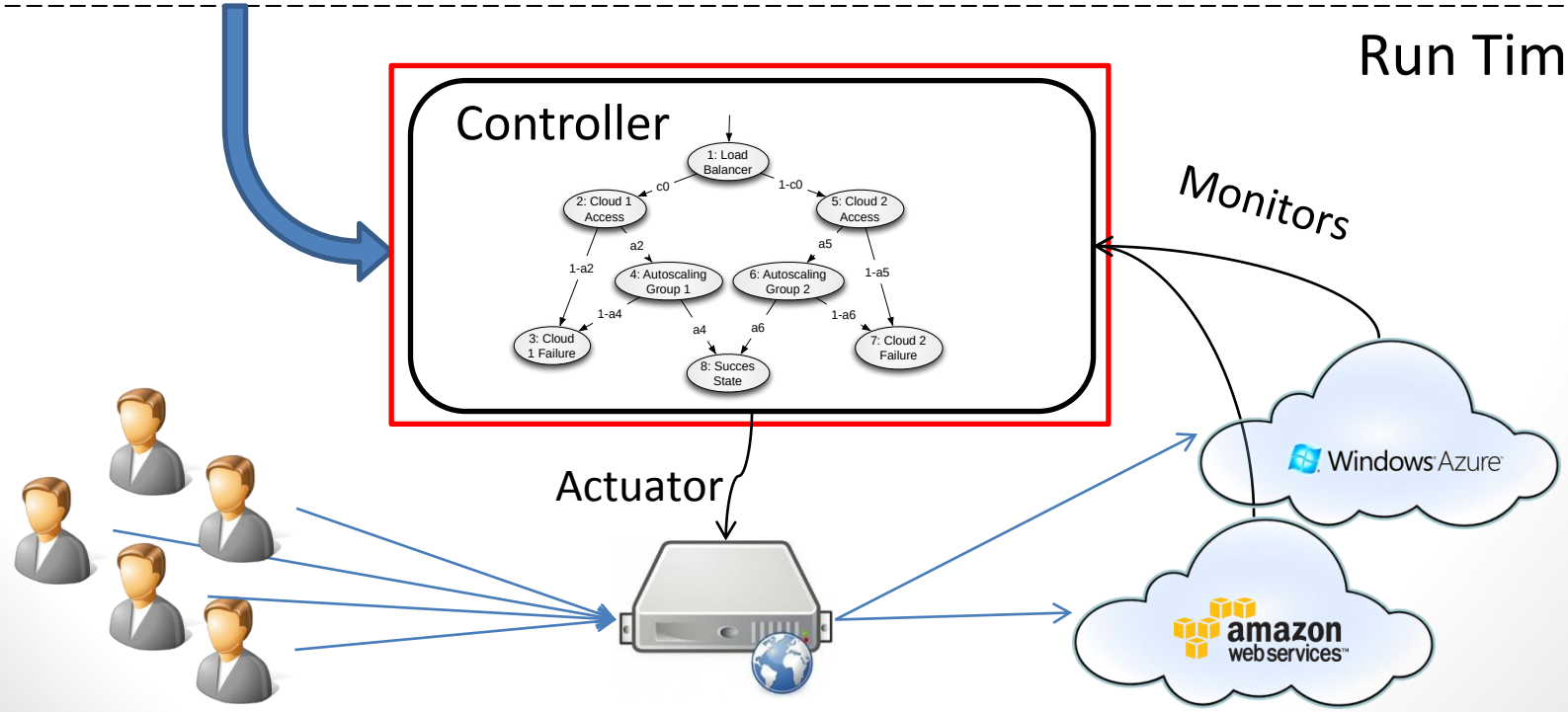
Multi-Cloud Model



Modeling Tool

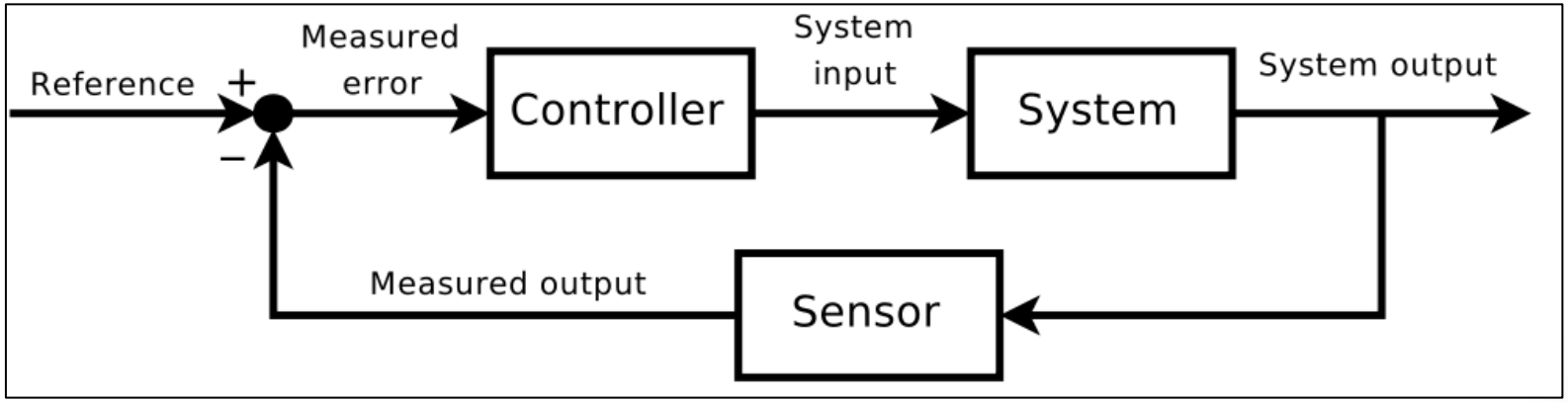


Run Time



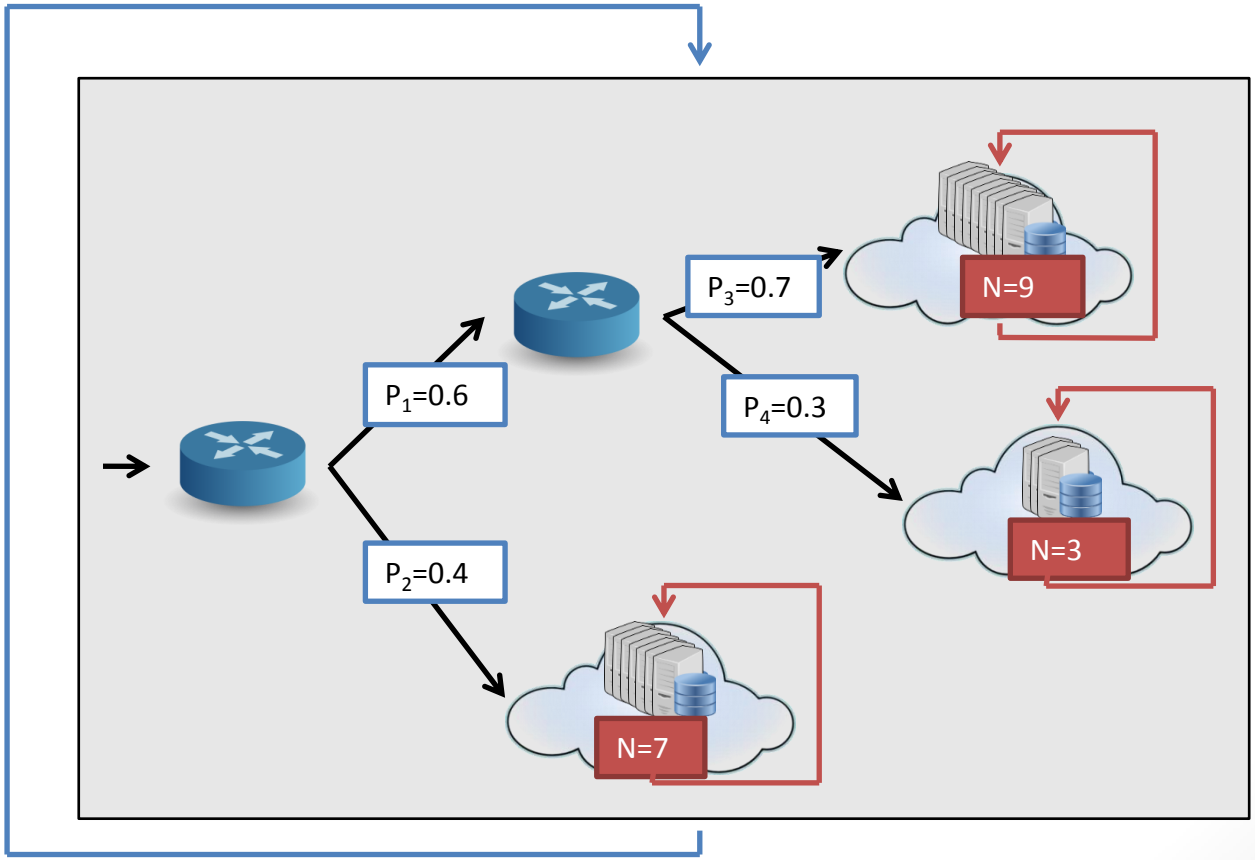
Building the controller

- **Objectives**
  - Guarantee the required availability
  - Minimize costs

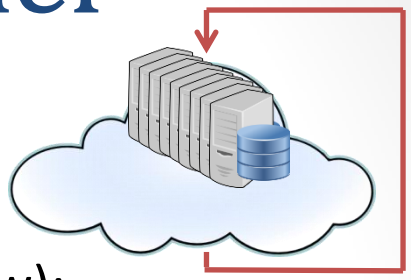


Building the controller

- **Objectives**
  - Guarantee the required availability
  - Minimize costs



The Autoscaling Controller



- **Reference:** average CPU usage – *u*
- **Control variable:** number of VMs – *n*
- **Monitored data** at the node (over a time window):
  - Arrival Rate – *AR*
  - VM Max Service Rate – *sr*

$$\bar{n} = \frac{AR}{sr \cdot u}$$

Desired Number of VMs

Current number of VMs

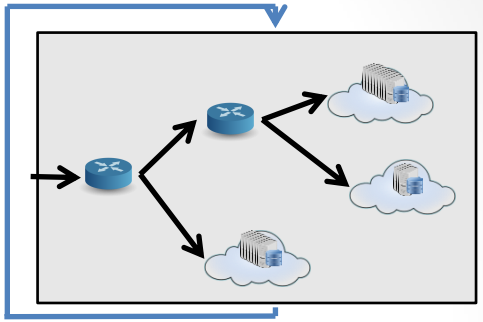
$$n(k+1) = \alpha n(k) + (1-\alpha)\bar{n}$$

Number of VMs at the next step

Convergence factor in (0,1)

The Load Balancer Controller

- **Reference:** System availability – v
- **Control variable:** traffic distribution probabilities – c_i
- **Monitored data** (over time window):
  - Incoming requests to node i - IN_i
  - Successful requests to node I – OUT_i
  - VM Max Service Rate – sr
  - Arrival Rate – AR
  - VM cost per second



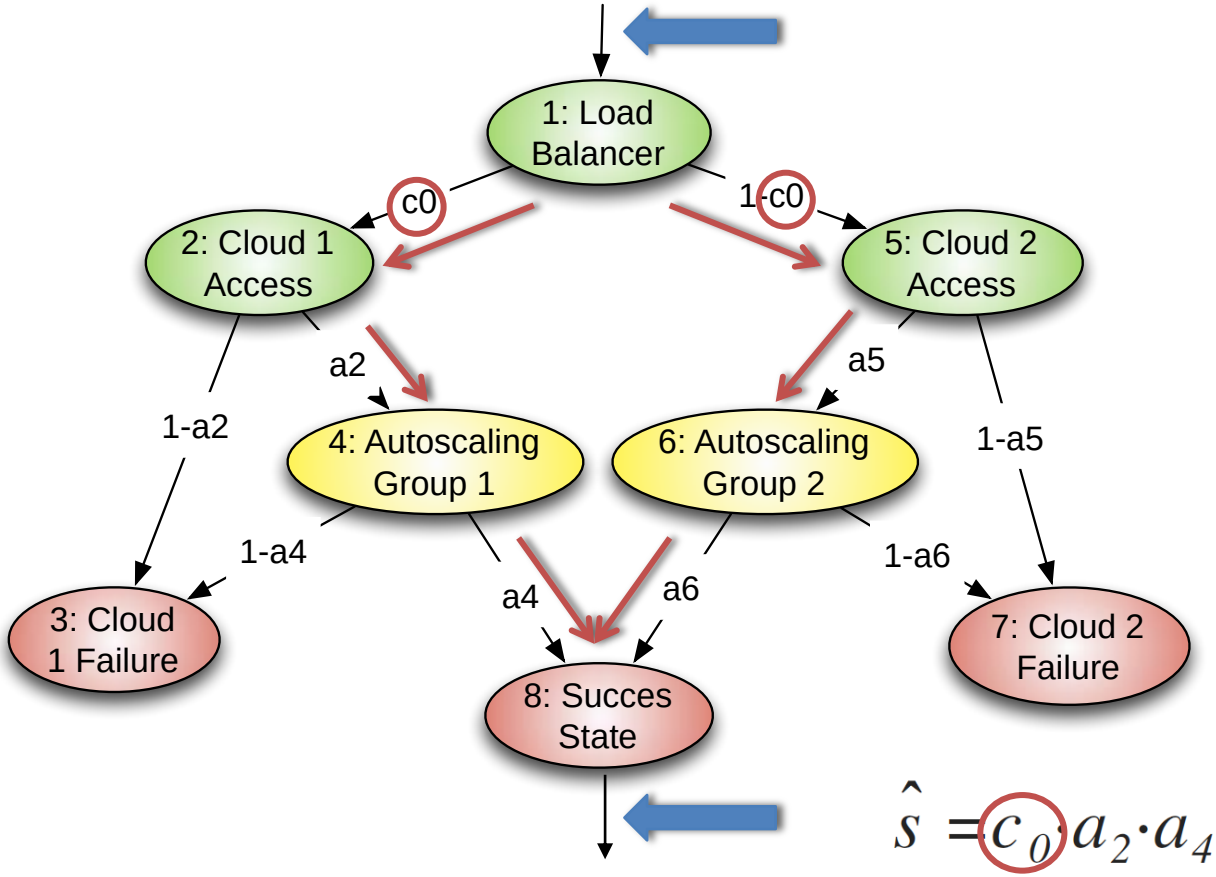
$$v(k+1) - \hat{s}(k+1|k) \leq \beta \cdot \max(0, v(k) - s(k))$$

Estimated system availability

Convergence factor in (0,1)

Measured system availability

The Load Balancer Controller



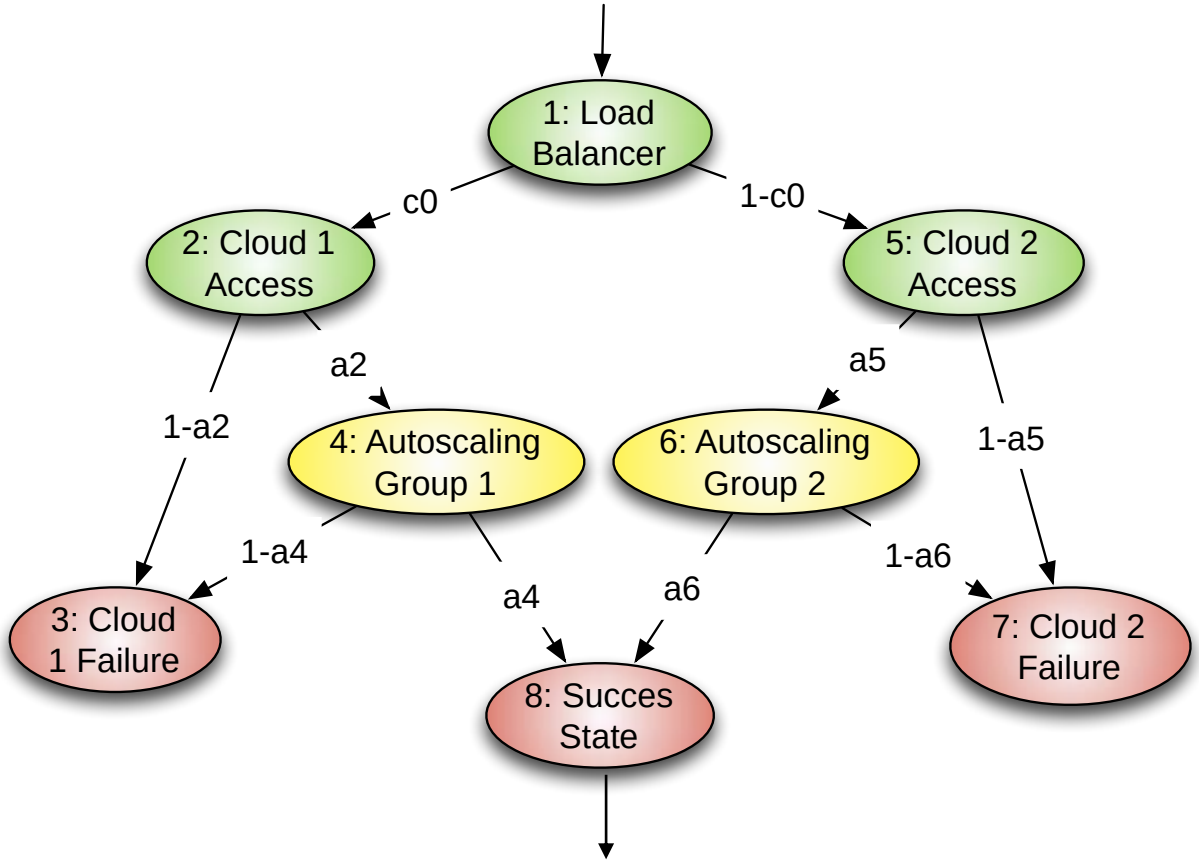
$$s = \frac{Out_8}{In_1}$$

$$\hat{s} = c_0 \cdot a_2 \cdot a_4 + (1 - c_0) \cdot a_5 \cdot a_6$$

$$a_i = \frac{Out_i}{In_i}$$

$$v(k+1) - \hat{s}\,(k+1|k) \leq \beta \cdot \max(0, v(k) - s(k))$$

The Load Balancer Controller



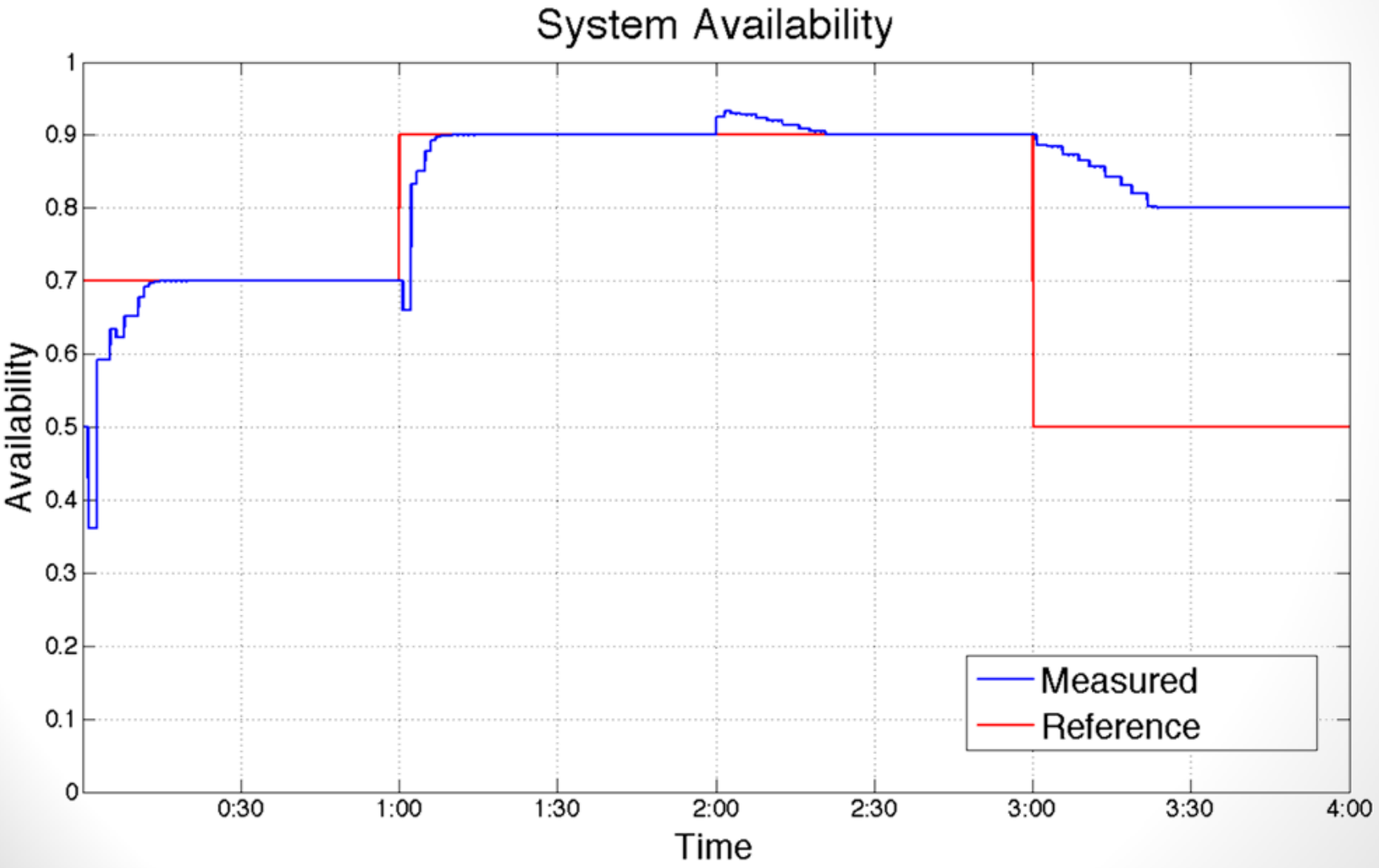
- Solution is chosen so to minimize an objective function $J(c_i)$
  - It is built so to allow cost minimization by preferring the most convenient Cloud, and to discourage nodes overloading

Evaluation

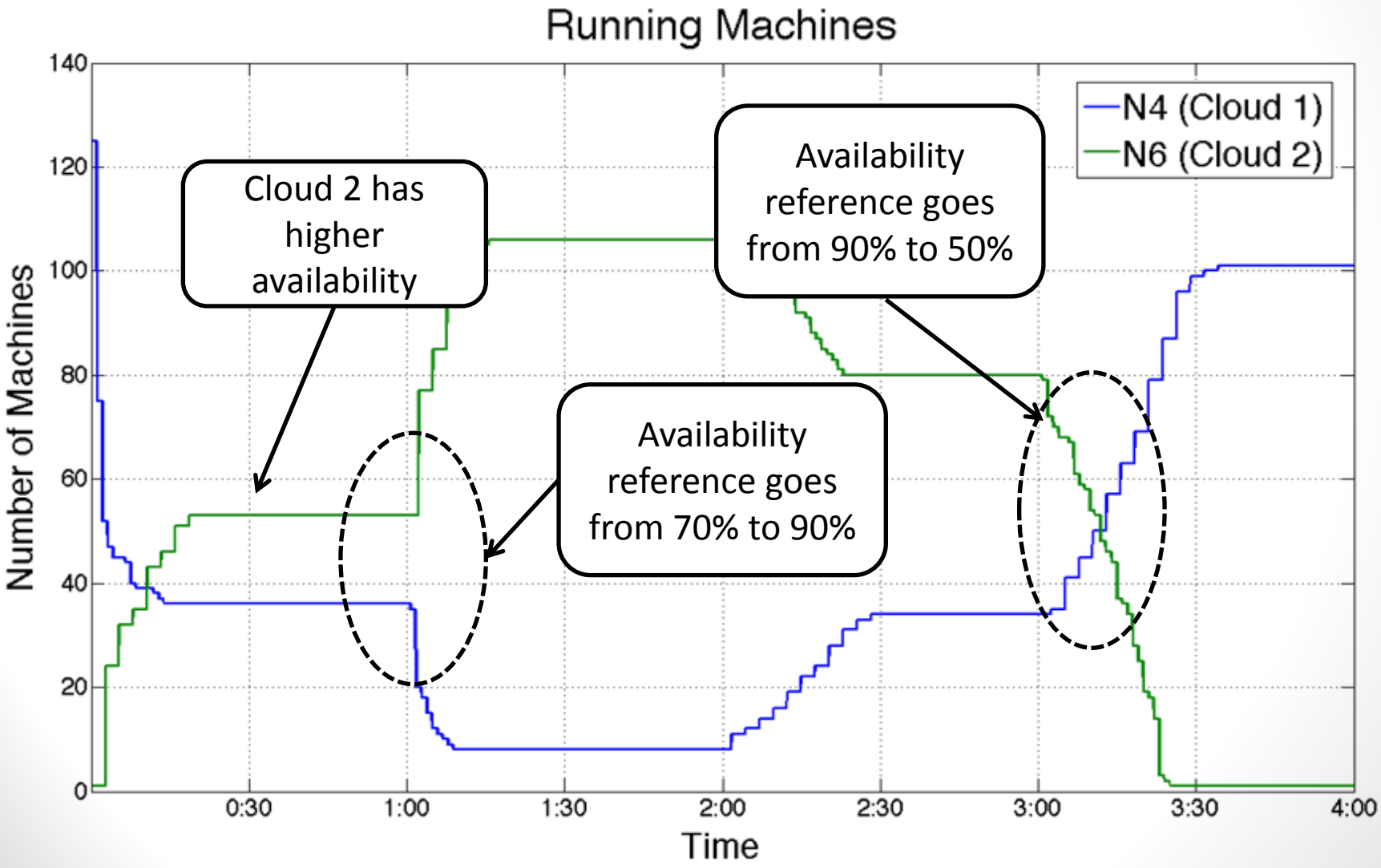
- **Objective**
  - Test how the controller is able to **track** the reference system availability
  - Test how the controller reacts to **sudden changes** in the environment, such as Cloud outages or performance degradations
- **Experiment setup**
  - For the evaluation we used Matlab
  - We implemented our controller
  - The environment and the different scenarios were simulated
  - One of the tested scenarios are now presented

Evaluation

Results



Evaluation
Results



Conclusions

- We definined an **adaptive approach** able to guarantee **availability requirements**, managing cloud to cloud migration and in-cloud autoscaling policies, **minimizing costs**

- The **controller** is able to track the reference system availability and to react to changes in the environment

Future Work

- Analyze convergence parameters (α and β) and CPU reference (u) setting, **studying optimality** of this choice
- The approach should provide **more realistic** descriptions and features of the current Cloud offer (e.g. pay by the hour)
- The proposed approach should be tested on **real Cloud infrastructures**

Any question?