José Ángel Bañares
Rafael Tolosana
Engineering & Architecture School - Zaragoza University, Spain
banares@unizar.es, rafaelt@unizar.es

Omer F. Rana
School of Computer Science & Informatics - Cardiff University, UK
o.f.rana@cs.cardiff.ac.uk

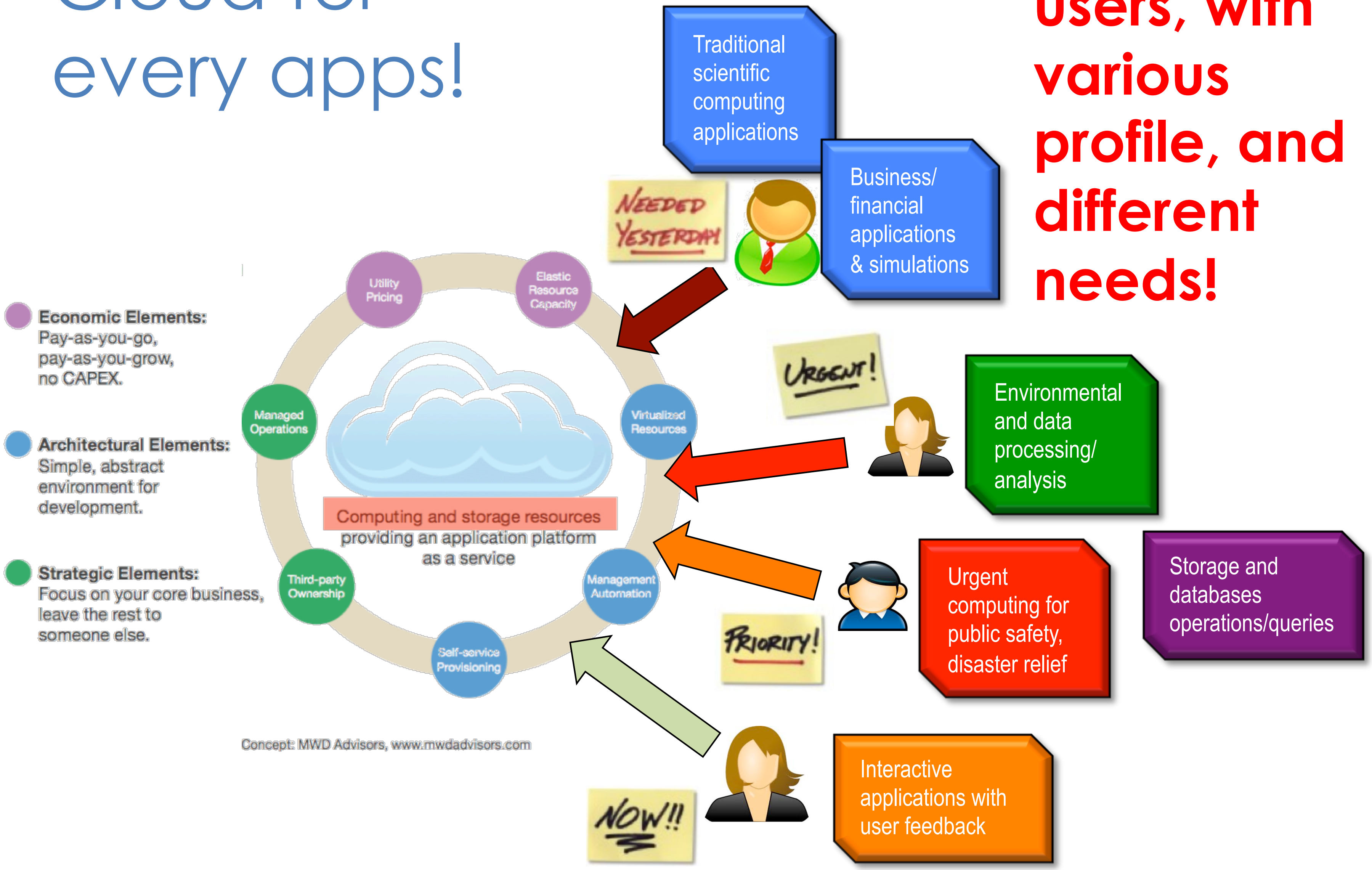
Congduc Pham
Laboratoire informatique- Université de Pau, France
congduc.pham@univ-pau.fr

# Revenue creation for rate adaptive stream management in multi-tenancy environments

GECON 2013
10th International Conference on Economics of Grids, Clouds, Systems and Services
September 18-20, 2013 Zaragoza, Spain

# Cloud for every apps!

**Many users, with various profile, and different needs!**



**Economic Elements:** Pay-as-you-go, pay-as-you-grow, no CAPEX.

**Architectural Elements:** Simple, abstract environment for development.

**Strategic Elements:** Focus on your core business, leave the rest to someone else.

Computing and storage resources providing an application platform as a service

Concept: MWD Advisors, www.mwdadvisors.com

Traditional scientific computing applications

Business/ financial applications & simulations

Environmental and data processing/ analysis

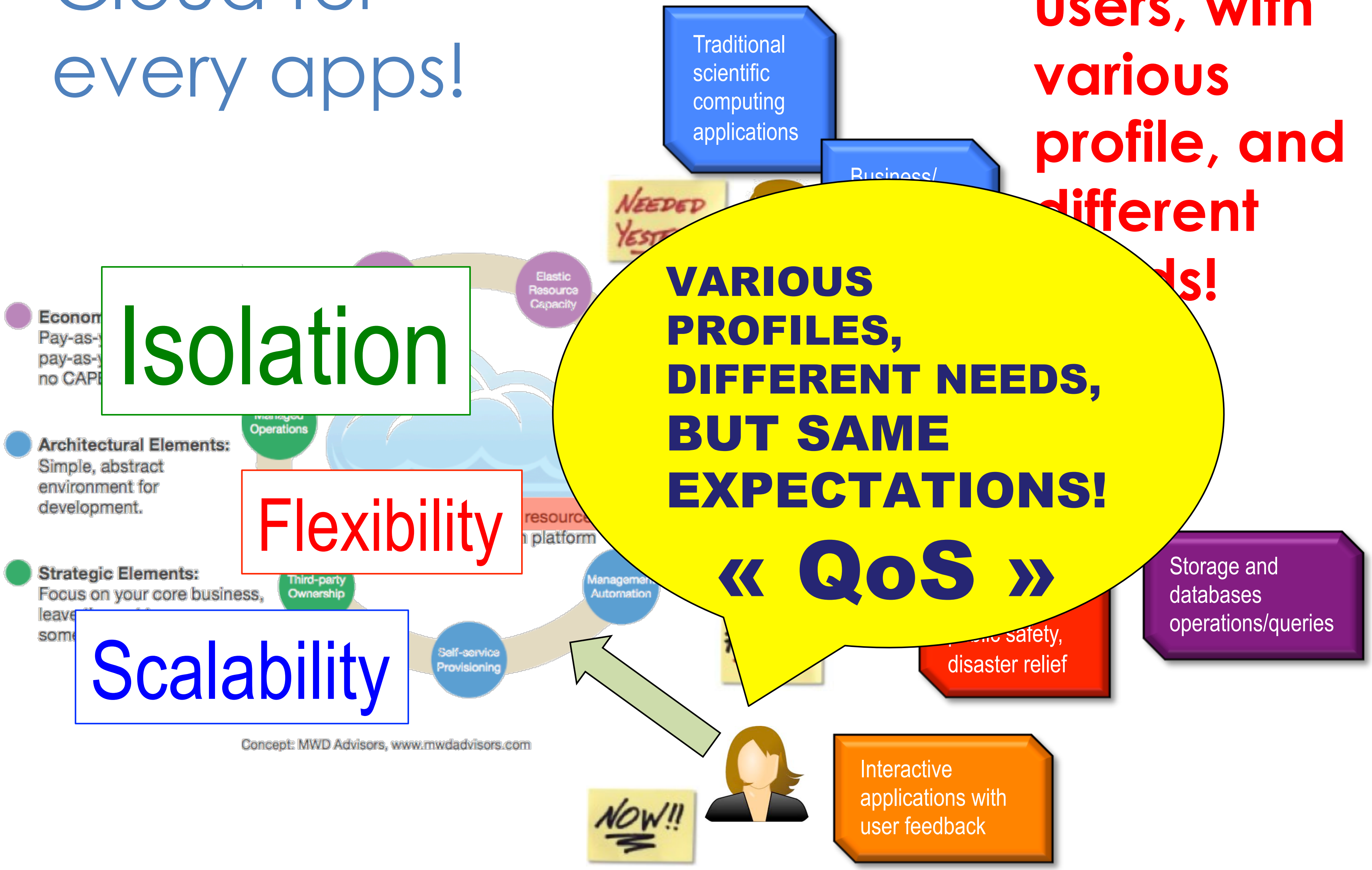
Urgent computing for public safety, disaster relief

Storage and databases operations/queries

Interactive applications with user feedback

# Cloud for every apps!

Many users, with various profile, and different needs!



Traditional scientific computing applications
Business/
NEEDED YEST
Elastic Resource Capacity
Econom
Pay-as-y
pay-as-y
no CAPE
Isolation
Managed Operations
Architectural Elements:
Simple, abstract environment for development.
Flexibility
resource
platform
Strategic Elements:
Focus on your core business,
leave
some
Third-party Ownership
Management Automation
Scalability
Self-service Provisioning
ic safety, disaster relief
Storage and databases operations/queries
NOW!!
Interactive applications with user feedback

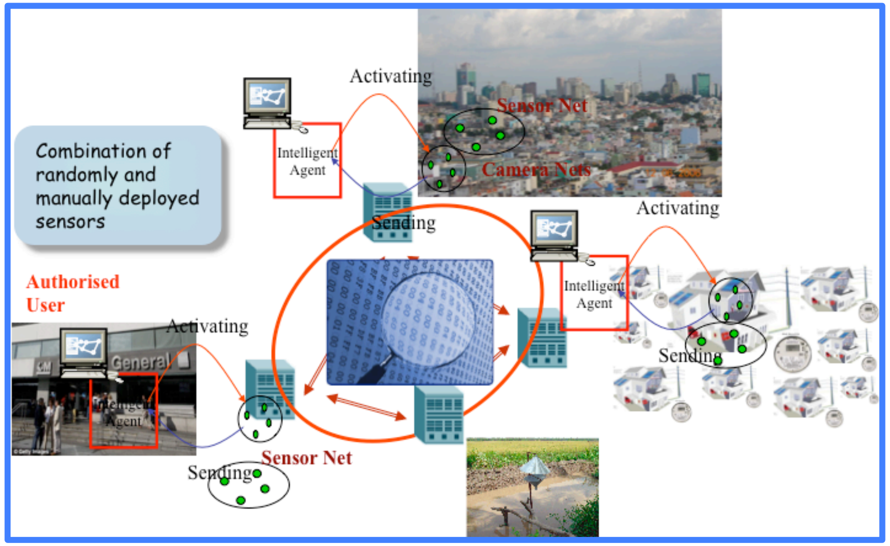
Concept: MWD Advisors, www.mwdadvisors.com

**VARIOUS PROFILES, DIFFERENT NEEDS, BUT SAME EXPECTATIONS!**

**« QoS »**

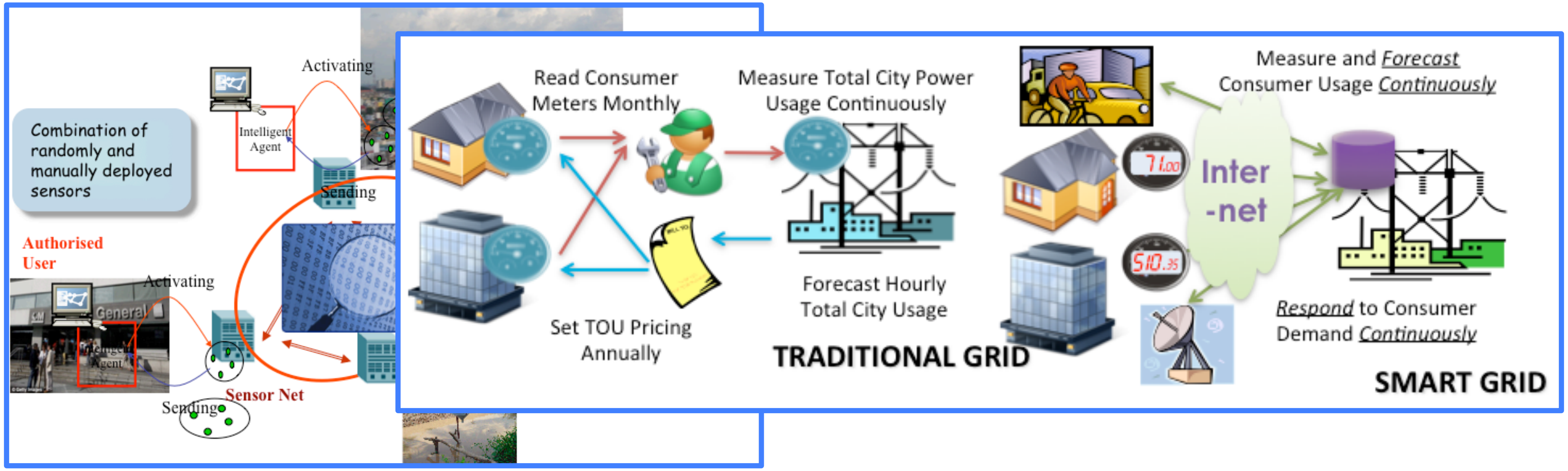
# Focus on data streams management

- Applications send **continuous/long-lived** data flows
- Data sent to a cloud **must** be processed in a **timely** manner but…
- … data rates can be **sporadic**.

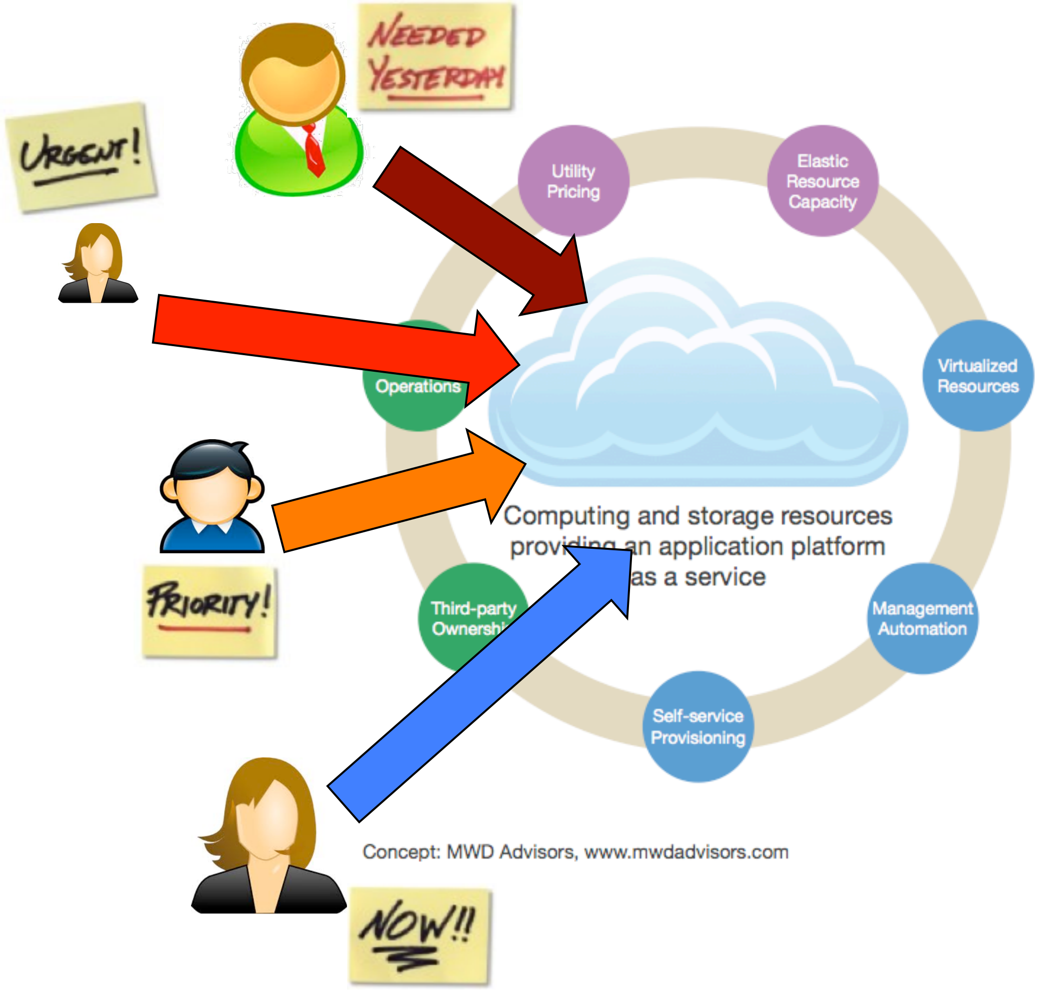


# Focus on data streams management

- Applications send **continuous/long-lived** data flows
- Data sent to a cloud **must** be processed in a **timely** manner but…
- … data rates can be **sporadic**.


Combination of randomly and manually deployed sensors
Authorised User
Activating
Intelligent Agent
Sending
Activating
General
Intelligent Agent
Sensor Net
Sending
Read Consumer Meters Monthly
Measure Total City Power Usage Continuously
Forecast Hourly Total City Usage
Set TOU Pricing Annually
TRADITIONAL GRID
Measure and Forecast Consumer Usage Continuously
71.00
510.35
Inter -net
Respond to Consumer Demand Continuously
SMART GRID

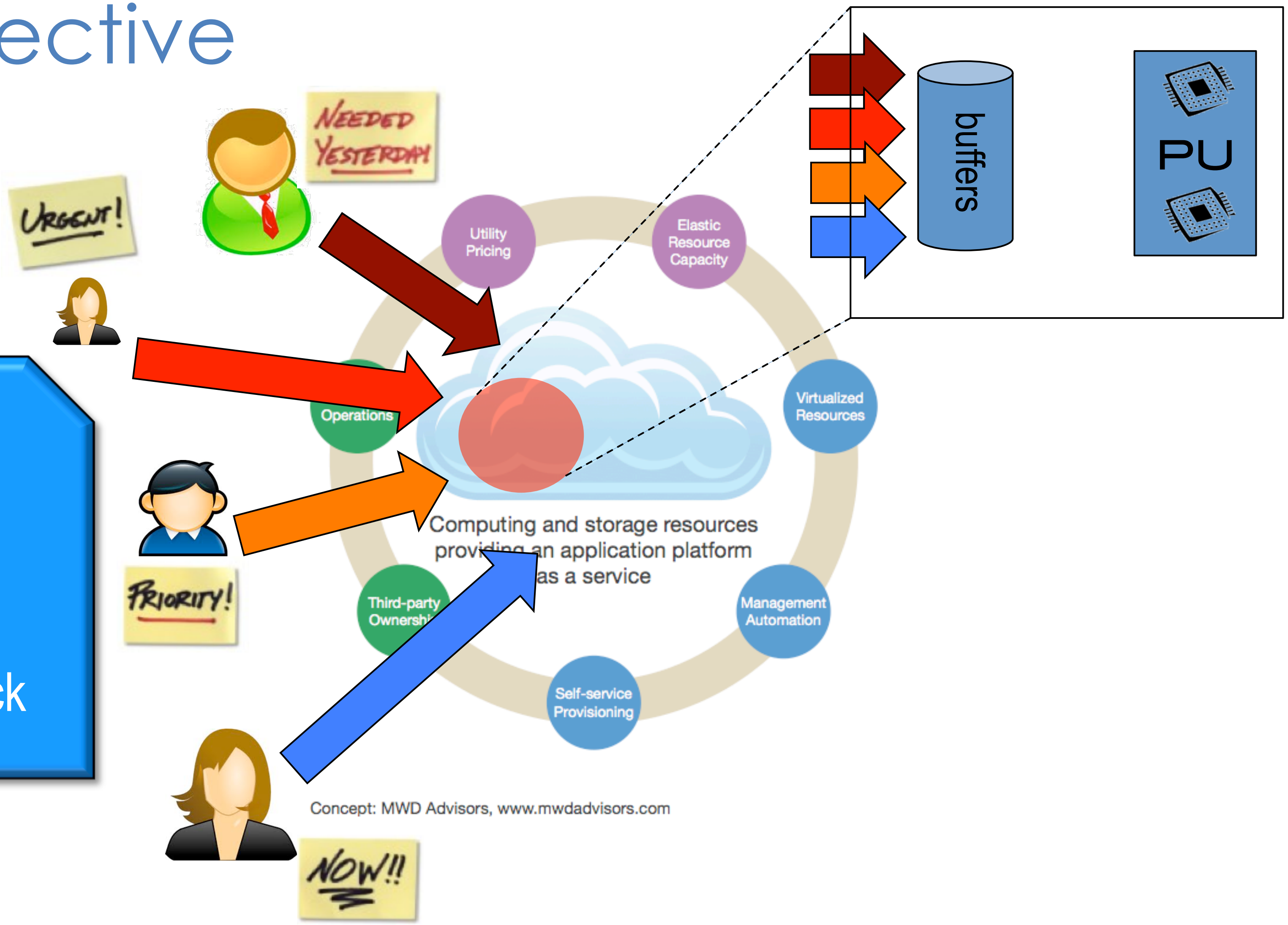
# From a cloud provider perspective


NEEDED YESTERDAY!
URGENT!
PRIORITY!
NOW!!
Utility Pricing
Elastic Resource Capacity
Virtualized Resources
Management Automation
Self-service Provisioning
Third-party Ownership
Operations
Computing and storage resources providing an application platform as a service

Concept: MWD Advisors, www.mwdadvisors.com

# From a cloud provider perspective

Network bandwidth is assumed to NOT BEING the bottleneck

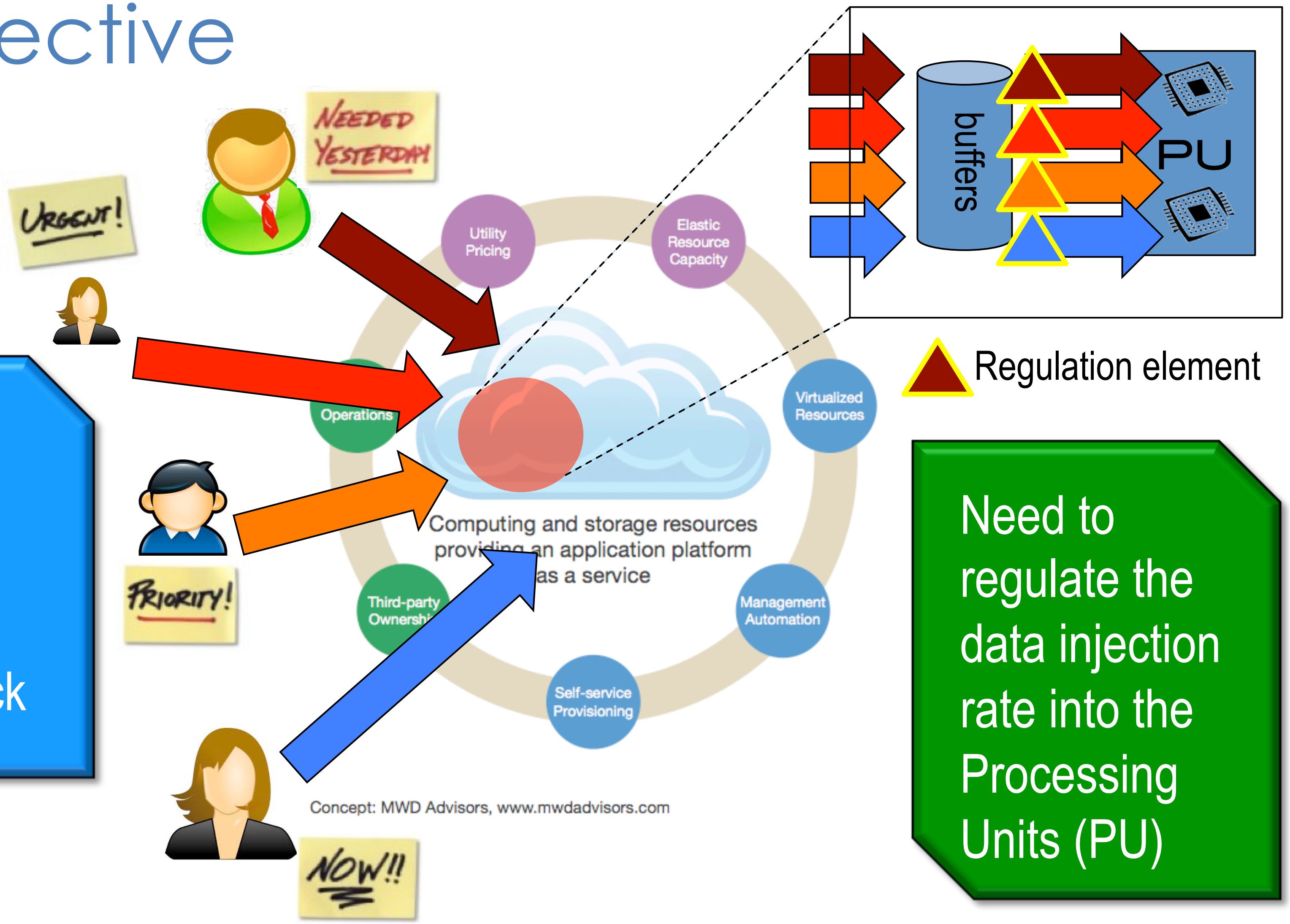


PU ARE the limiting resources!

Concept: MWD Advisors, www.mwdadvisors.com

# From a cloud provider perspective

Network bandwidth is assumed to NOT BEING the bottleneck



Concept: MWD Advisors, www.mwdadvisors.com

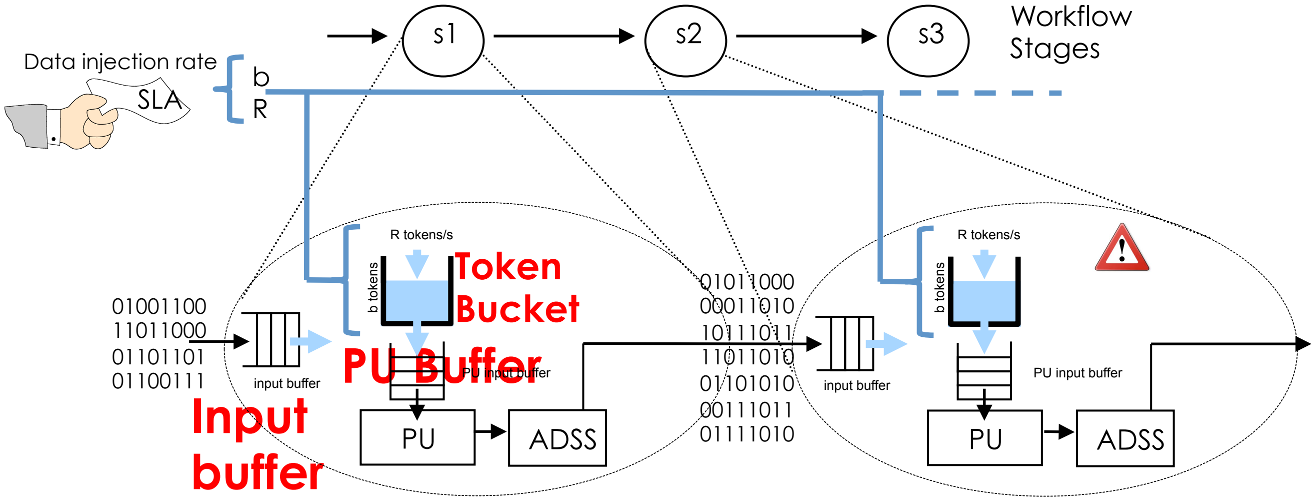
PU ARE the limiting resources!

Regulation element

Need to regulate the data injection rate into the Processing Units (PU)

R. Tolosana, J. Banares, C. Pham, O. Rana, "Enforcing QoS in Scientific Workflow Systems Enacted Over Cloud Infrastructures", Journal of Computer and System Science (JCSS), 78(5), Elsevier.

# System Architecture

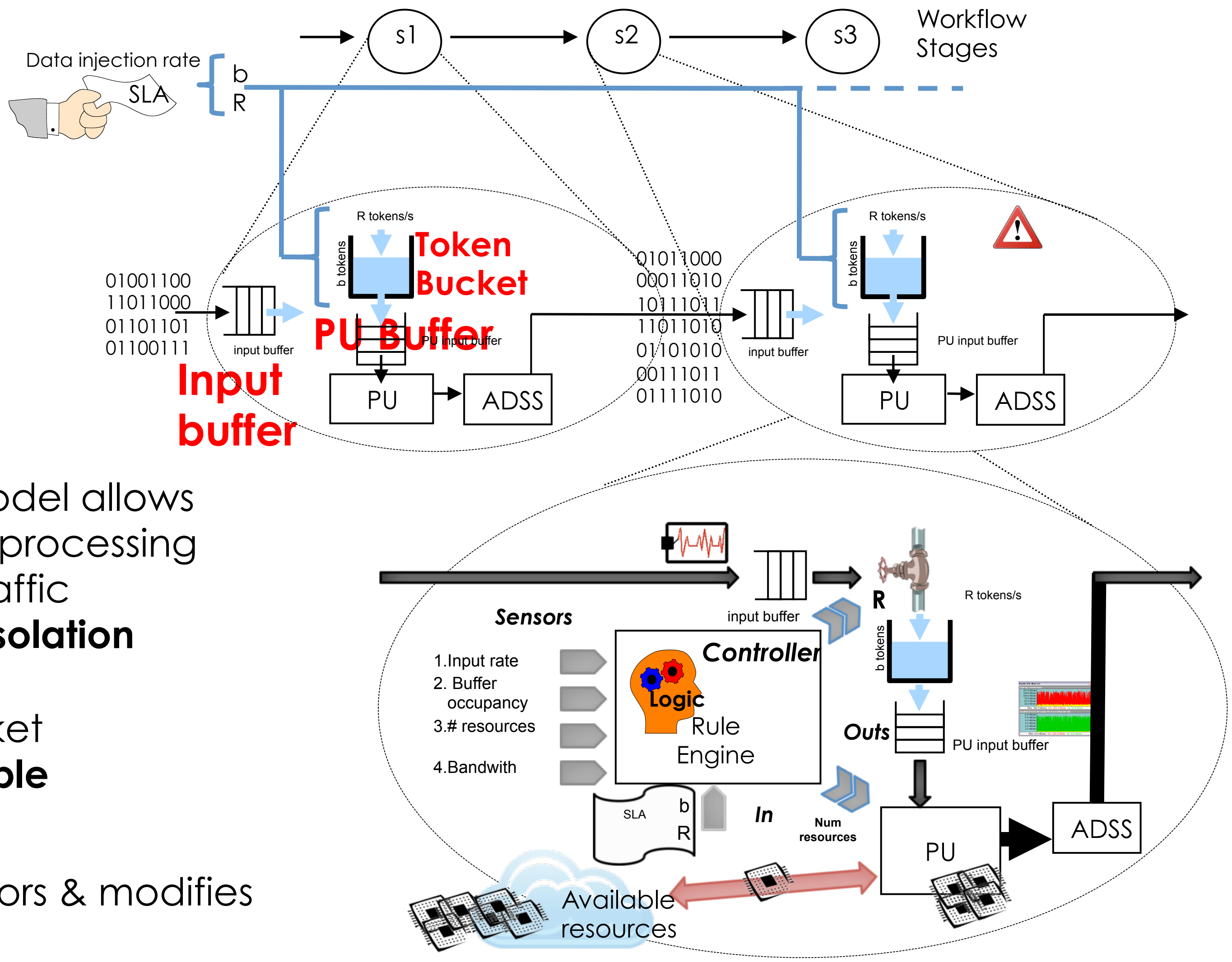


**Token Bucket** model allows for variable rate processing with bounded traffic envelop ➔**flow isolation**

Each token bucket provides us **tunable** parameters: b,R

R. Tolosana, J. Banares, C. Pham, O. Rana, "Enforcing QoS in Scientific Workflow Systems Enacted Over Cloud Infrastructures", Journal of Computer and System Science (JCSS), 78(5), Elsevier.

# System Architecture



**Token Bucket** model allows for variable rate processing with bounded traffic envelop →**flow isolation**

Each token bucket provides us **tunable** parameters: b,R

**Controller**: monitors & modifies behaviour

# Last year GECON'12

## Adaptive infrastructure for shared clouds

- **Multiple** concurrent data streams with **SLA**
- **Token bucket behaviour** is regulated by **b, R** parameters, flow **isolation** at each node
- **Rule-based SLAs** can specify **behaviours** allowing the **controller** to take different **actions** when a threshold is reached
    - **Load-shedding:** drop data stored by the token bucket buffer
    - **Modify** the mean injection rate **R**

## Basic revenue model

- **User classes, e.g. :** Gold, Silver, Bronze,…
- **Revenue:** price charge to n clients for m operations: $\sum_{i=1}^{n}\sum_{j=1}^{m} Pr(O_{ij})$
- **Cost:** for performing such operations: $c(O_{ij})$
- **Penalisation:** in case of QoS violation for client: $PSLA_{ij}$

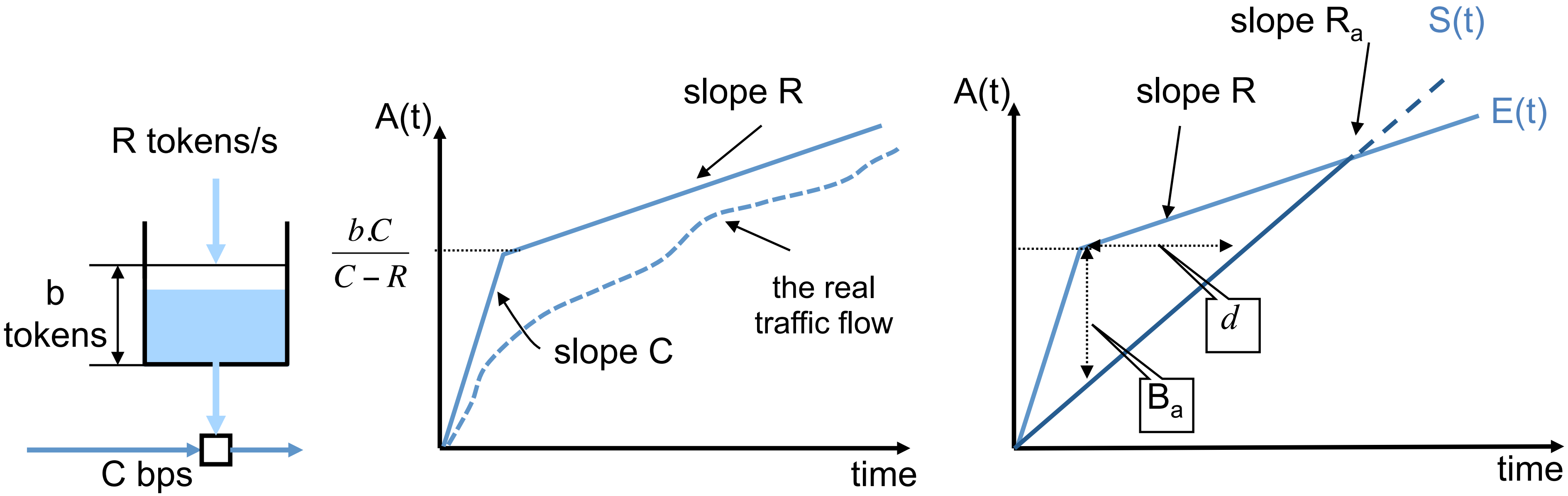
## Maximizing provider revenue

$$\sum_{i=1}^{n}\sum_{j=1}^{m} Pr(O_{ij}) - \sum_{i=1}^{n}\sum_{j=1}^{m} min(c(O_{ij}), PSLA_{ij})$$

# Unified resource mngt & more elaborated revenue model

- Allocation of new resources may have **inadequate time scale** or be **very costly**
- QoS requirements are often defined using the **worst case scenario** or **statistically**
- **Business policies** can be used to improve revenue and to provide more flexibility in SLA definition
- Cloud providers can take advantage of locally unused resources that are cheaper than allocating new resources → **redistribute unused resources**
- Revenue & penalisation depends on user class, i.e. **Gold** users incur both **high revenue** and **high penalty** while **Bronze** can have **low revenue** and **no penalty**→**redistribute pre-allocated ressources from less-prioritized users**

# Token Bucket for traffic shaping

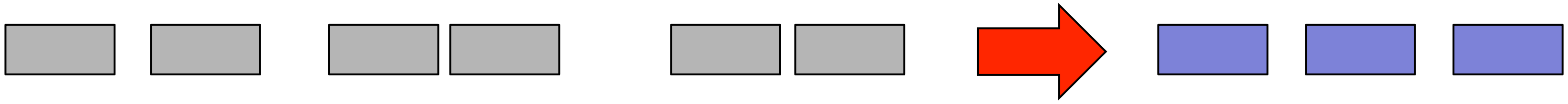


A(t): Amount of data arriving up to time t

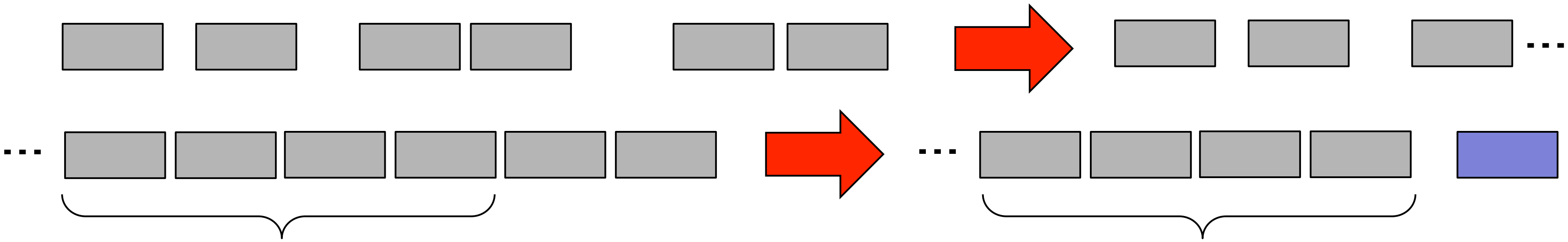
- Two key parameters of interest:
  - R: Also called the **committed information rate** (CIR), it specifies how much data can be sent or forwarded per unit time on average
  - b: it specifies for **each burst** how much data can be sent within a given time without creating scheduling concerns. Tokens in excess are normally dropped.

# Shaping in image

- Constant

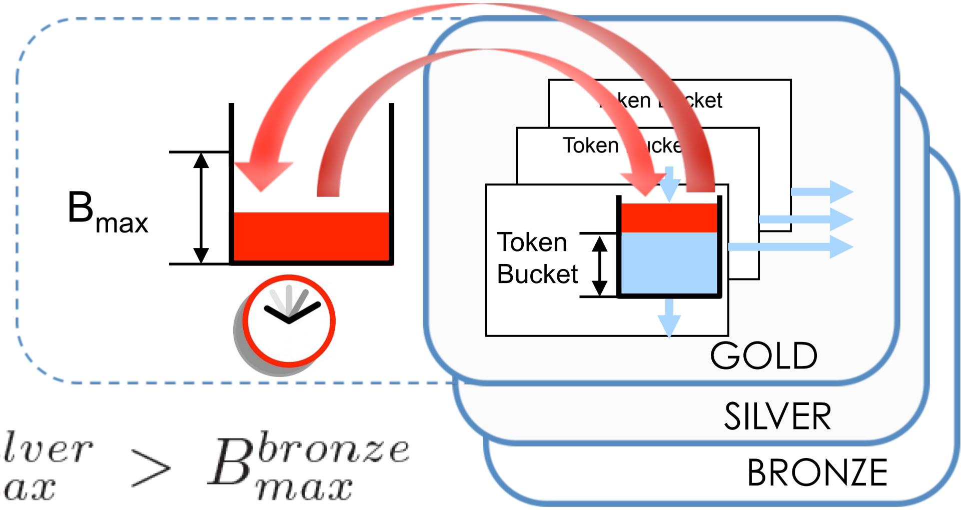


- Dynamic



Dynamic shaping allows for variable data injection rate (variable sensing rate for instance). Need however to bound the amount of provided service per time period T.

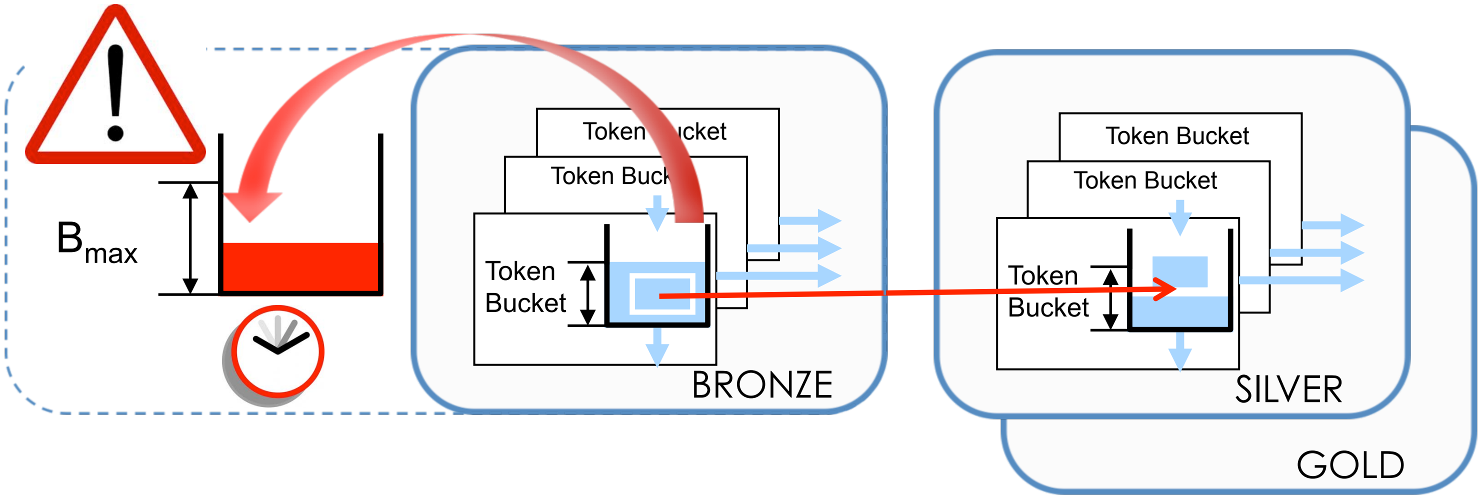
# Unified resource mngt with TB ➔Redistribute unused resources



$$B_{max}^{gold} > B_{max}^{silver} > B_{max}^{bronze}$$

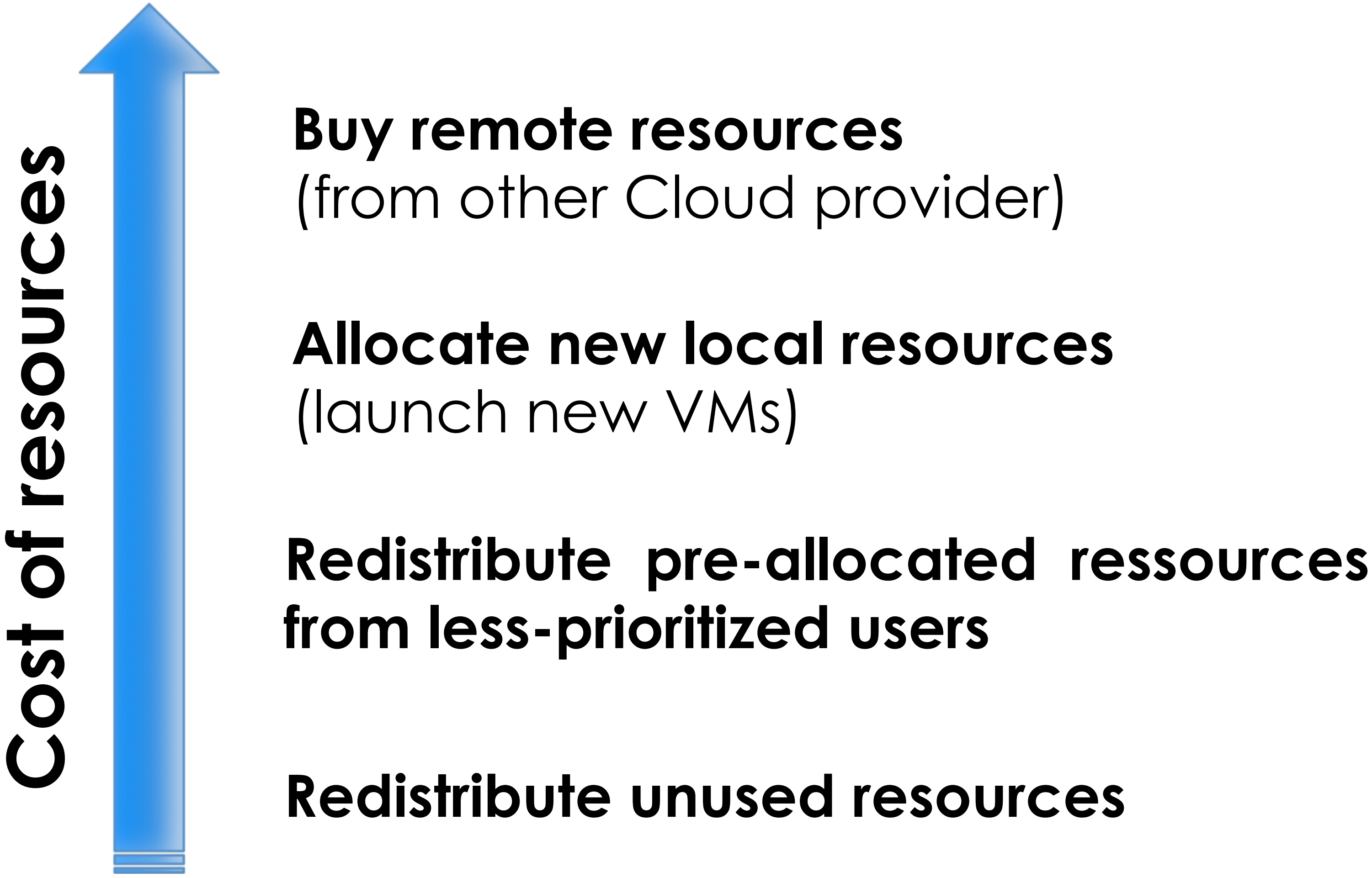
- Under-utilization of resource in a flow will produce tokens in excess
- Within a service class,
  - Tokens in excess of all flows are collected and stored up to $B_{max}$ tokens
  - Token's lifetime is limited to a few control intervals to limit inconsistency

# Unified resource mngt with TB ➔Take resources from lower classes



- Silver class has higher revenue and higher penalty than Bronze class for example: shortage of resource in Silver class **is more costly**
- Taking resources from Bronze to Silver is more **revenue-efficient**
  - Tokens are taken directly from a Bronze flow's token bucket
  - Can put a limit to the number of tokens the system can take
- **Safer** than taking tokens from the Bronze unused token bucket

# More elaborated revenue model



Cost of resources

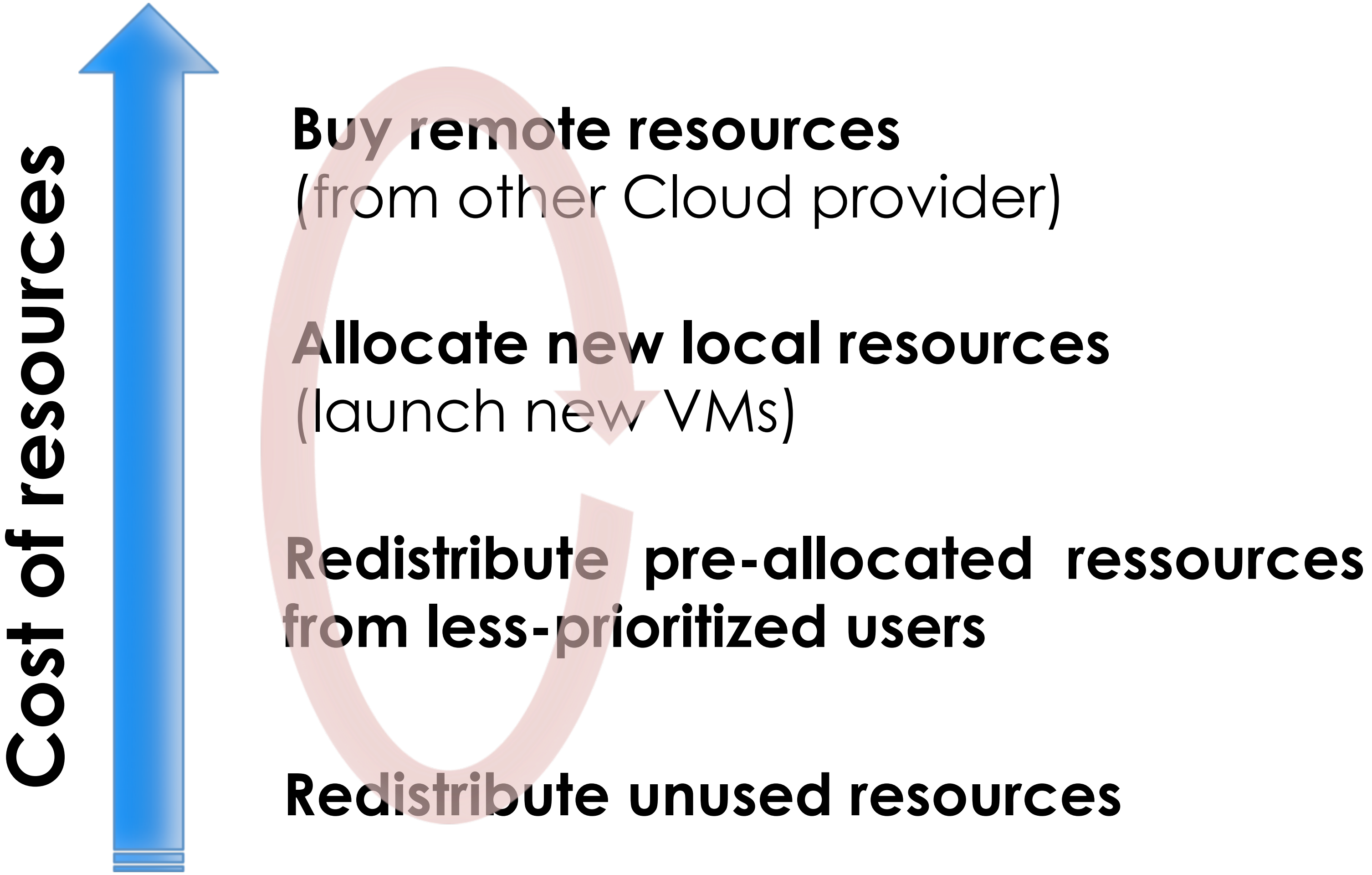
**Buy remote resources**
(from other Cloud provider)

**Allocate new local resources**
(launch new VMs)

**Redistribute pre-allocated ressources from less-prioritized users**

**Redistribute unused resources**

# More elaborated revenue model



Cost of resources

**Buy remote resources**
(from other Cloud provider)

**Allocate new local resources**
(launch new VMs)

**Redistribute pre-allocated ressources from less-prioritized users**

**Redistribute unused resources**

# Instant revenue & global revenue

$$Instant\ Revenue = \sum_{i=1}^{n}(CostPU_{client} - CostPU_{provider}) * \#PU \\ - \sum_{i=1}^{n} \#penalties_i * CostPenalties_i \\ - \Delta\#PU * CostPU_{provider}$$

- **Global revenue** is the accumulated *InstantRevenue* over time.
- With the unified resource management proposition, we can
  - **Reduce the number of penalties** by using unused resources
  - **Reduce Δ#PU** by taking resources from less priority flows
- Using a single performance criterion, i.e. global revenue, simplifies the optimization problem

# Simulation settings

1 data chunk = 1 token

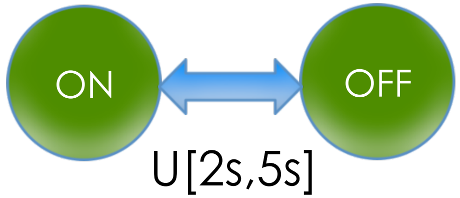
4

Data injection rate

SLA

b=10 tokens
R=20 tokens/s

Poisson



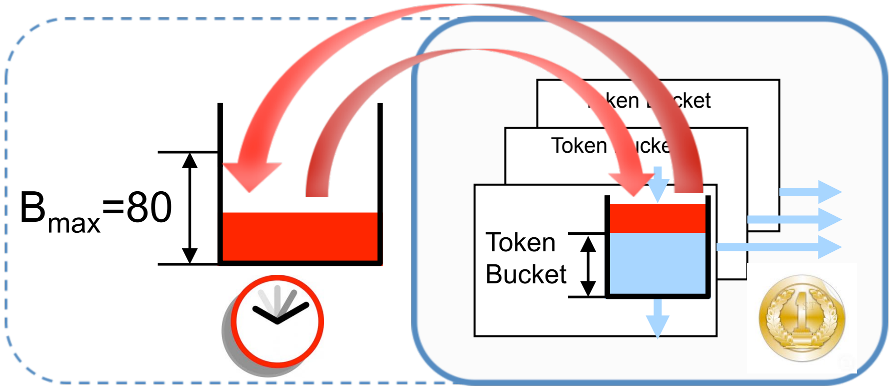
N

10 data chunk/s

20u/s

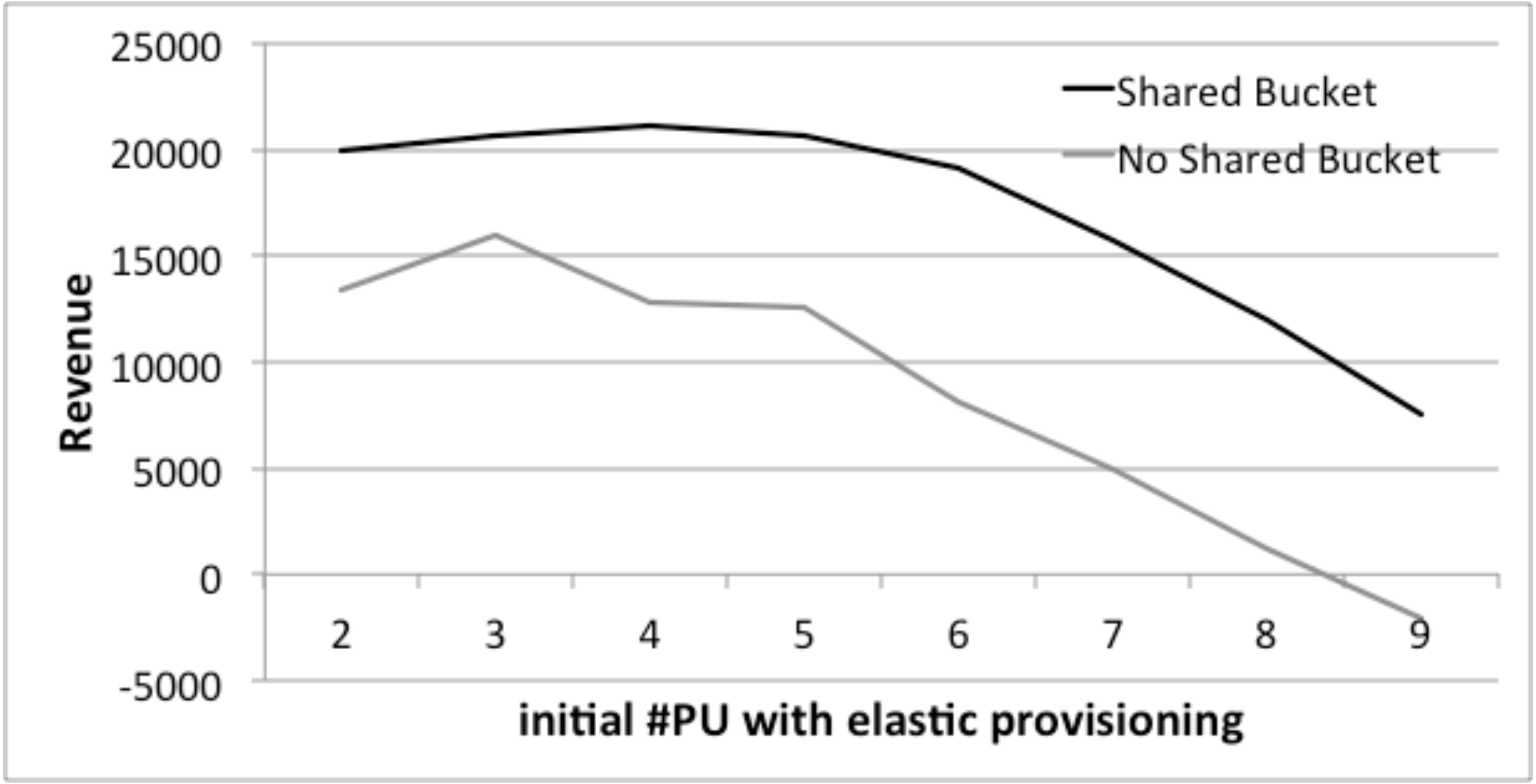
15u/s

600u/s

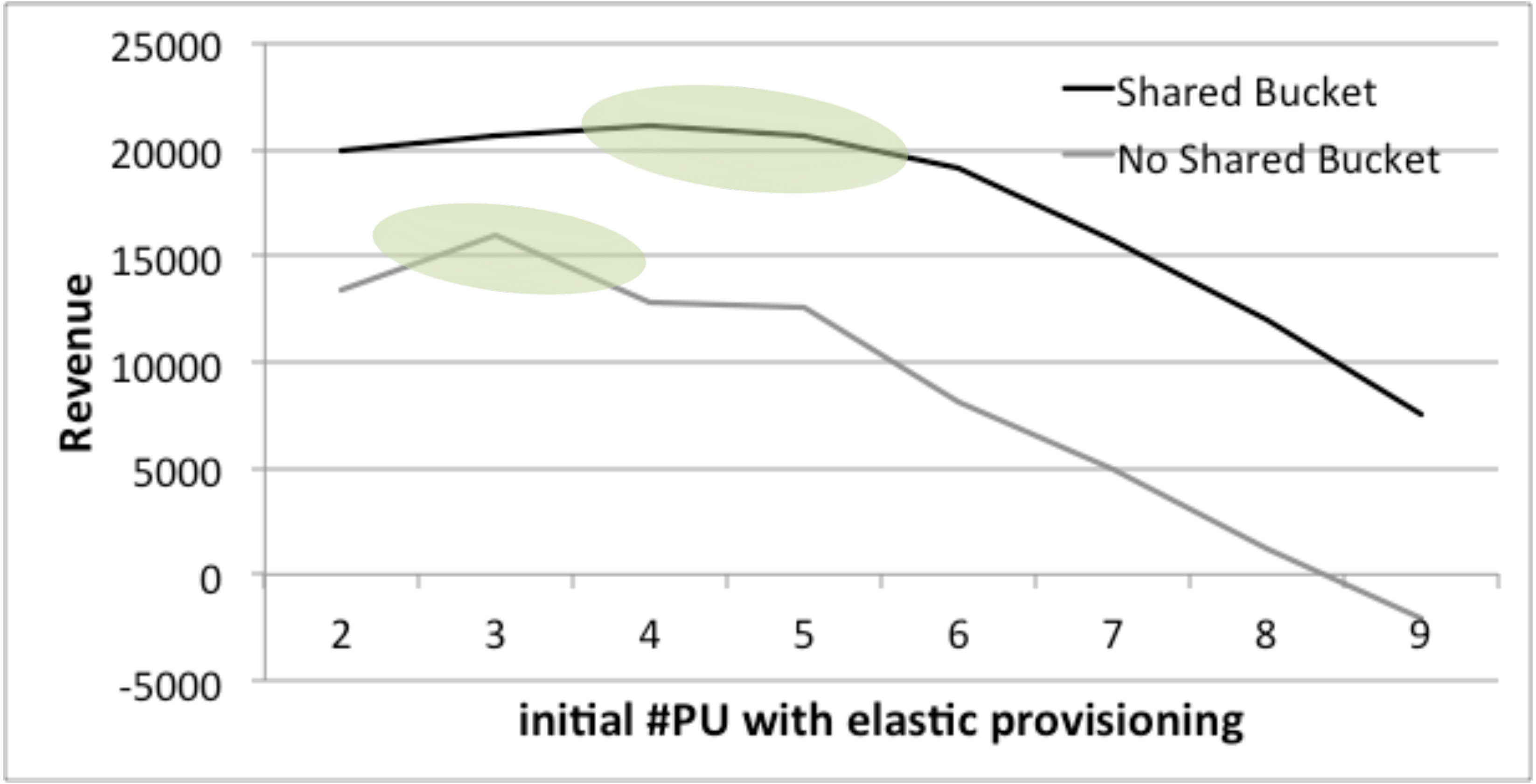


On average, 4 PU are required to process the 4 Gold flows

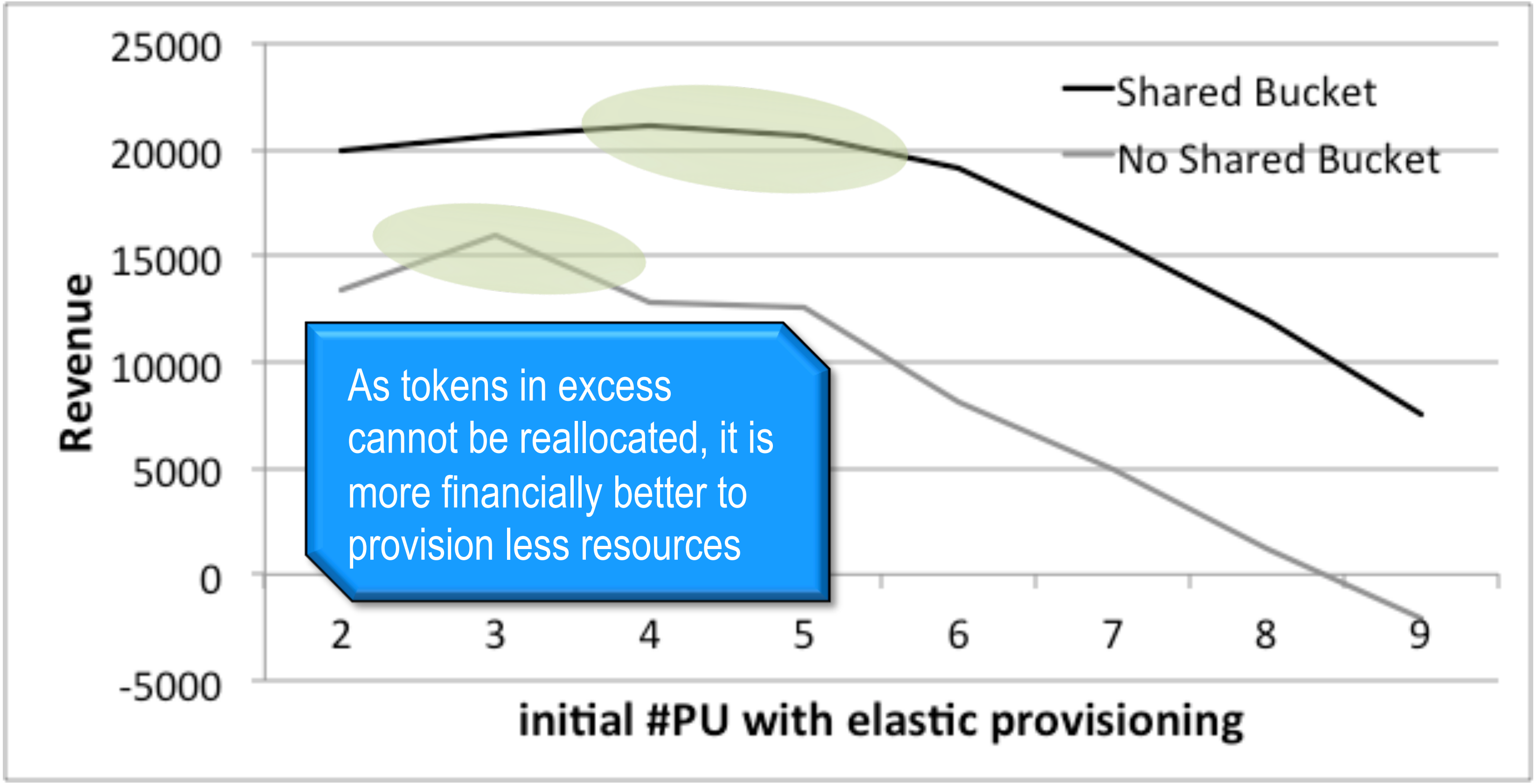
# Simulation results - 300s with elastic provisioning



# Simulation results - 300s with elastic provisioning

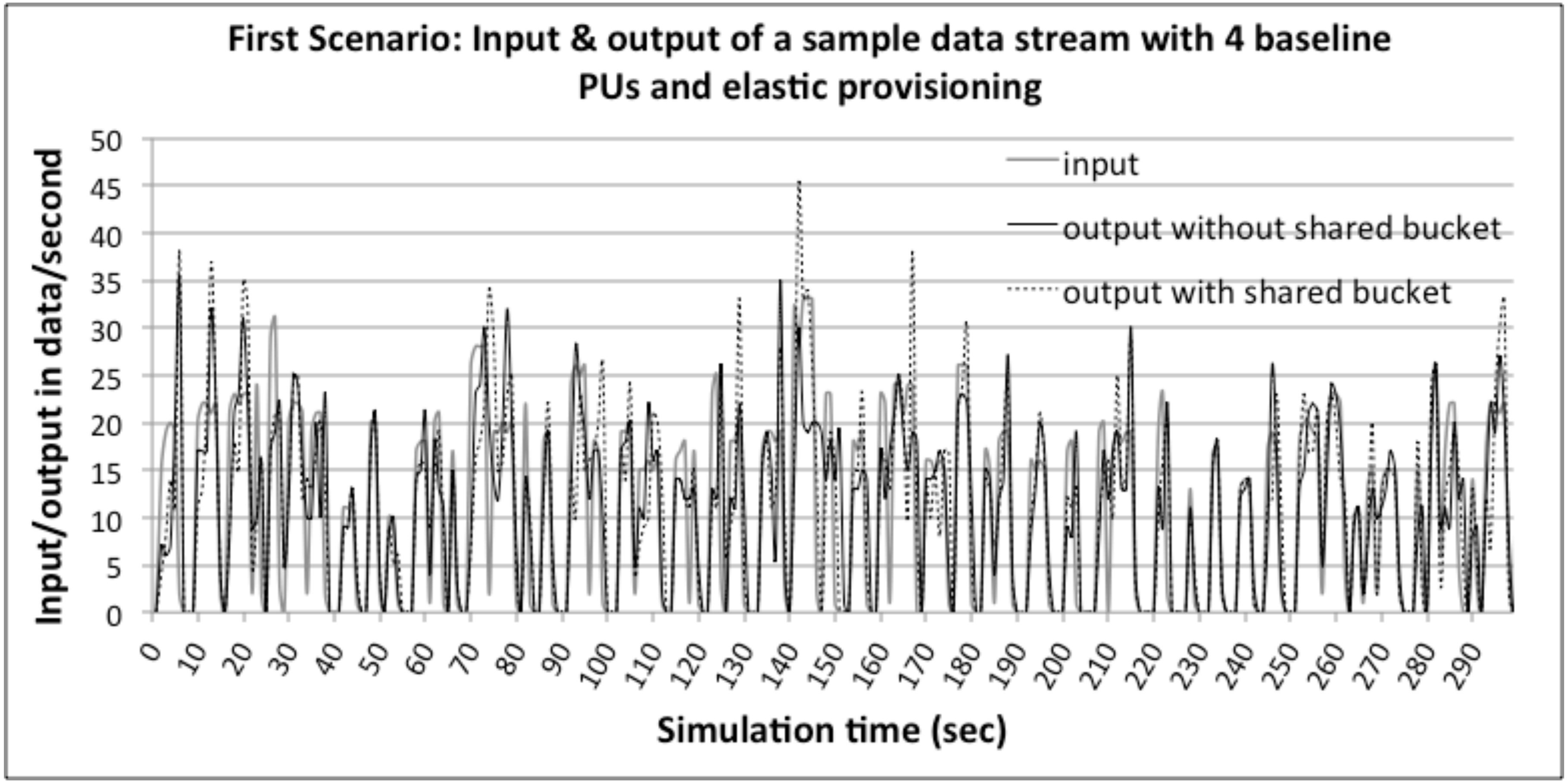


# Simulation results - 300s with elastic provisioning



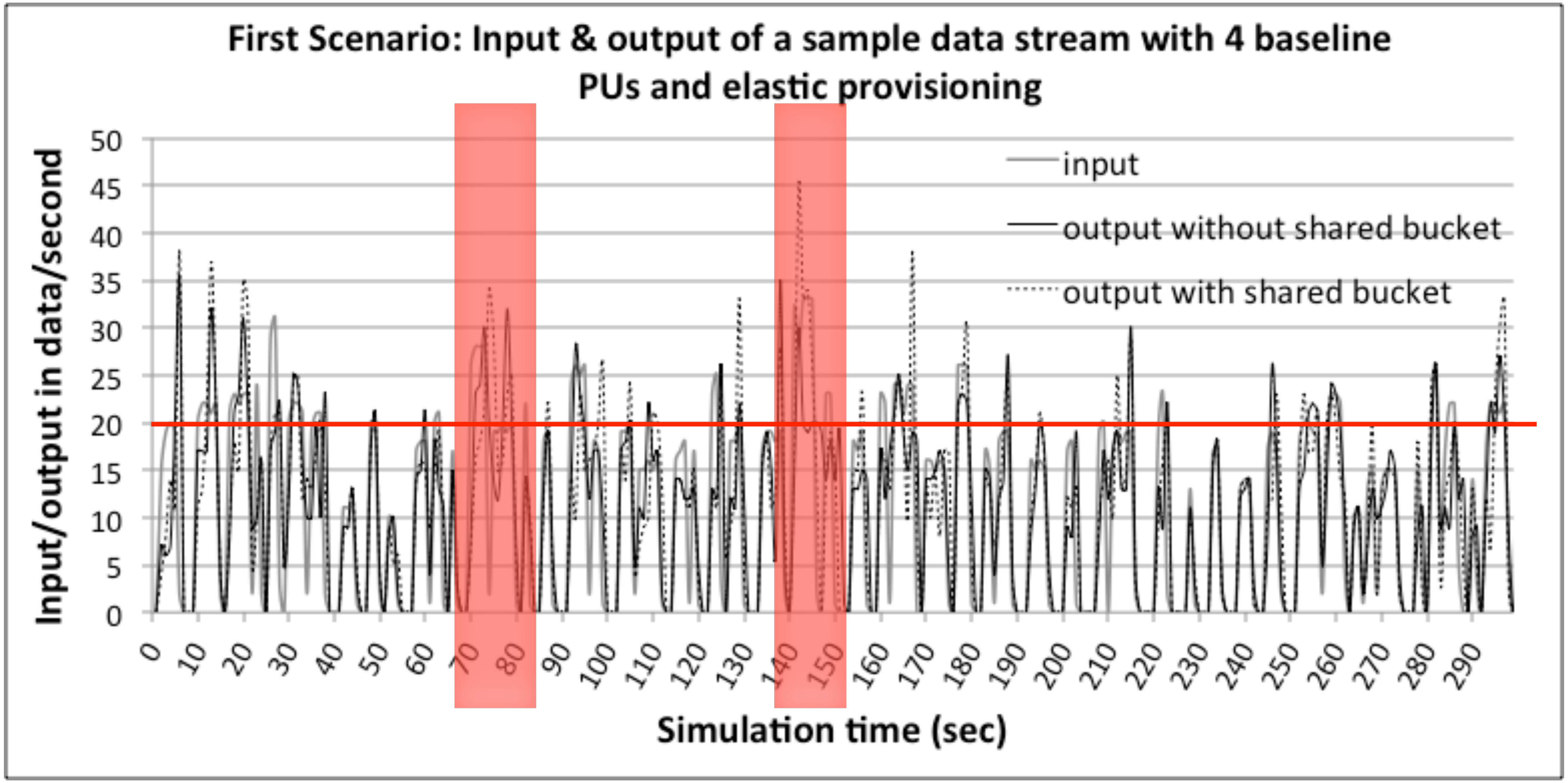
As tokens in excess cannot be reallocated, it is more financially better to provision less resources

# Simulation results - throughput 4PU

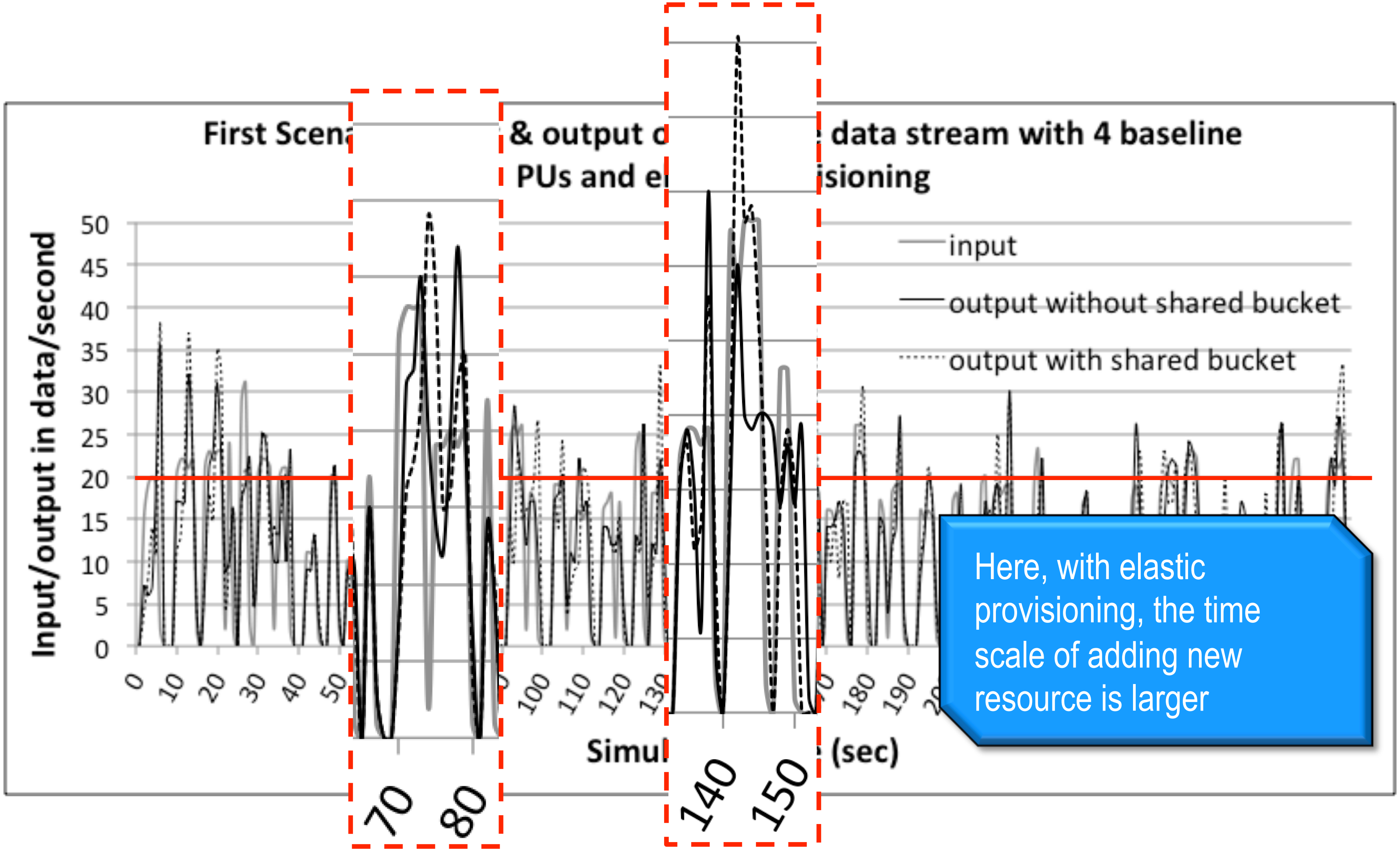


Universidad Zaragoza

# Simulation results - throughput 4PU

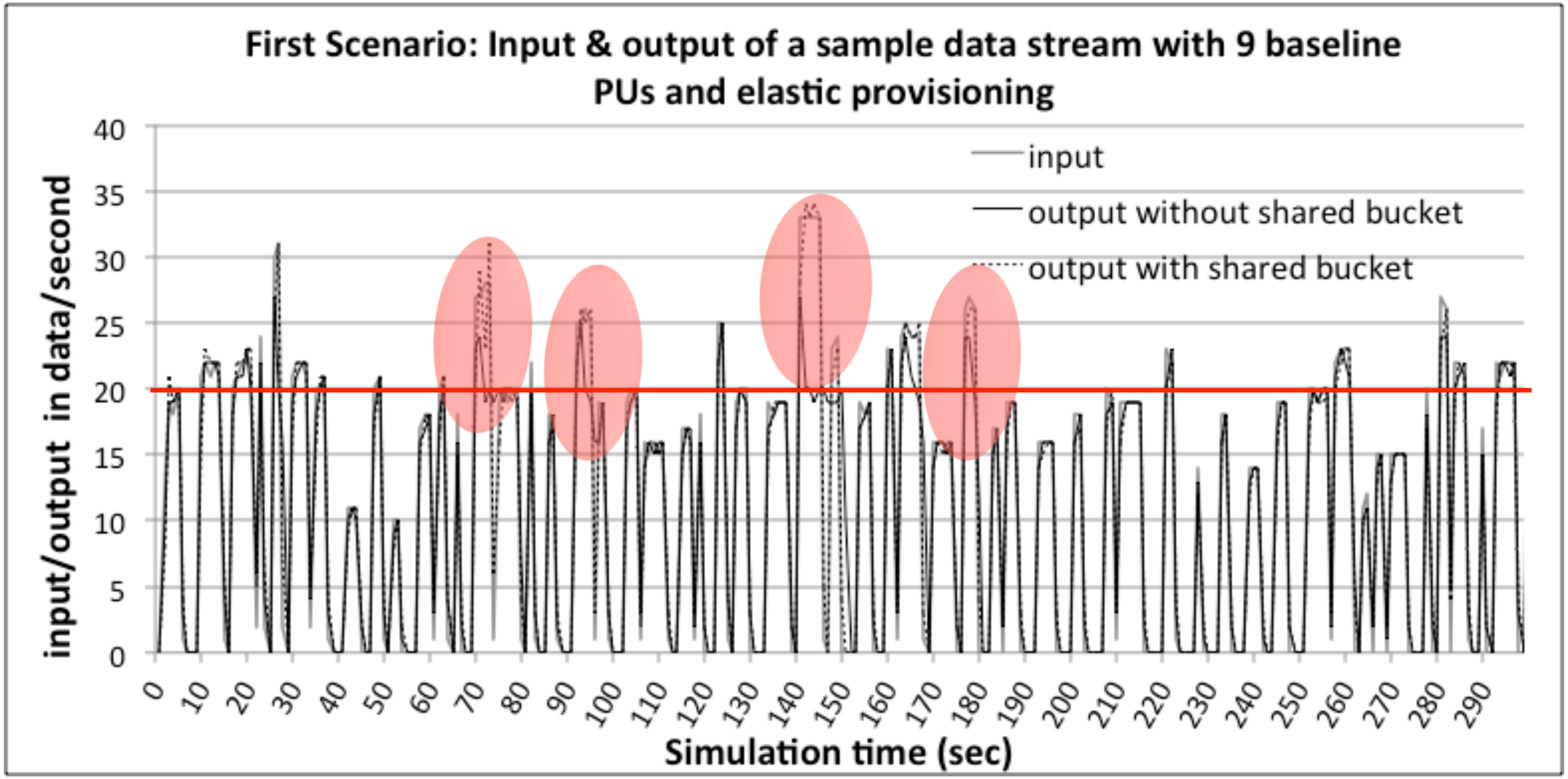


# Simulation results - throughput 4PU

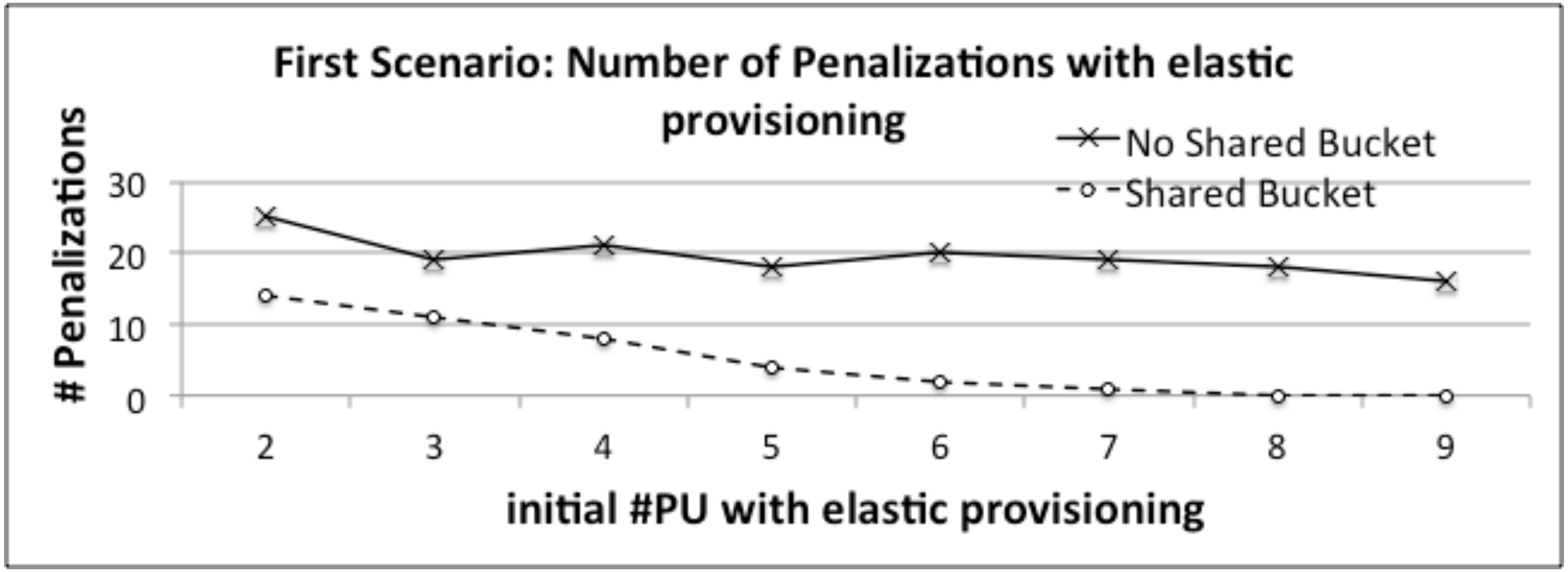


Here, with elastic provisioning, the time scale of adding new resource is larger

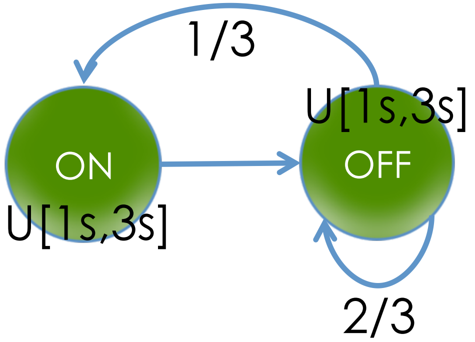
# Simulation results - throughput 9PU

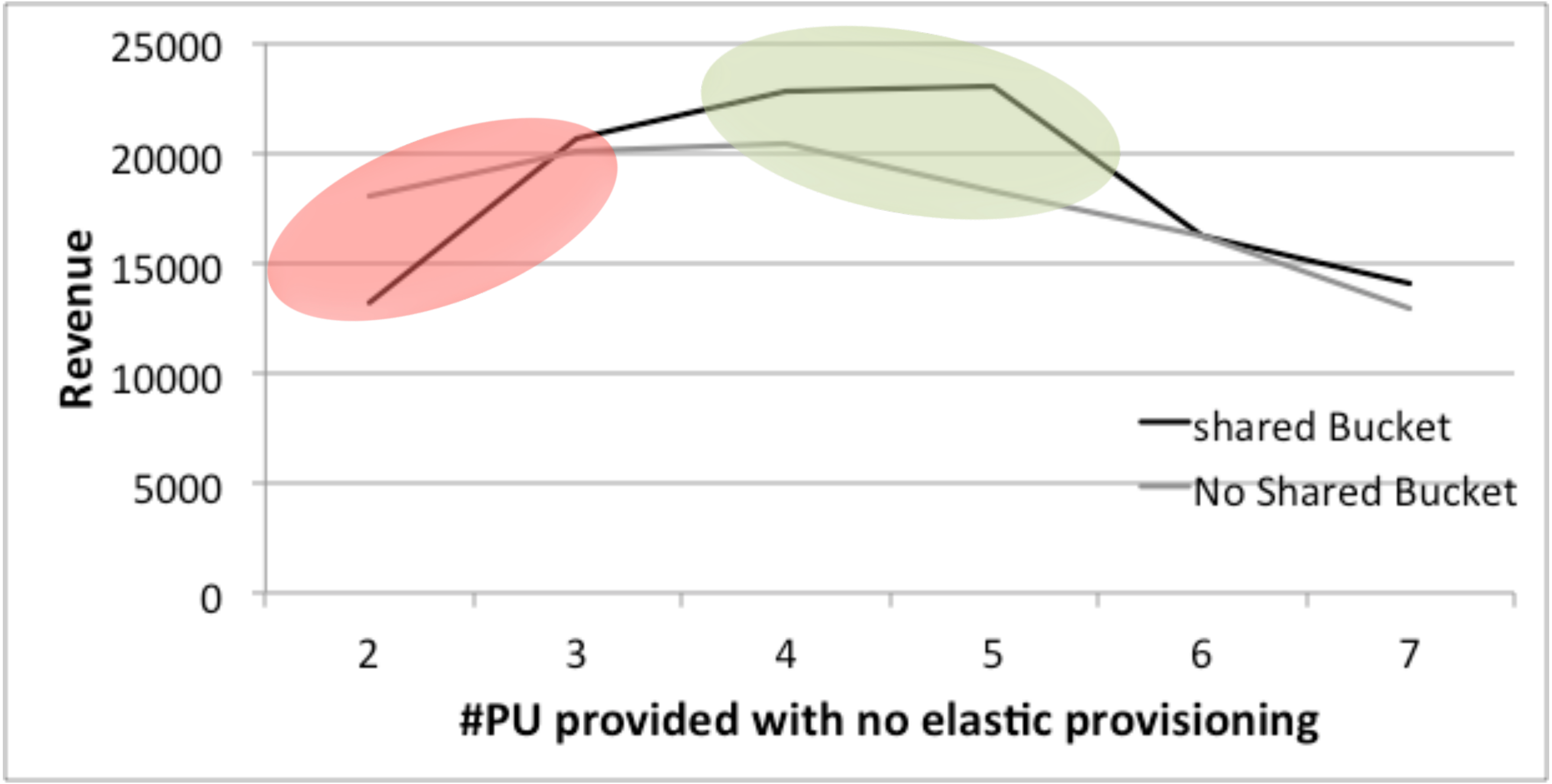


# Simulation results
# number of penalisations

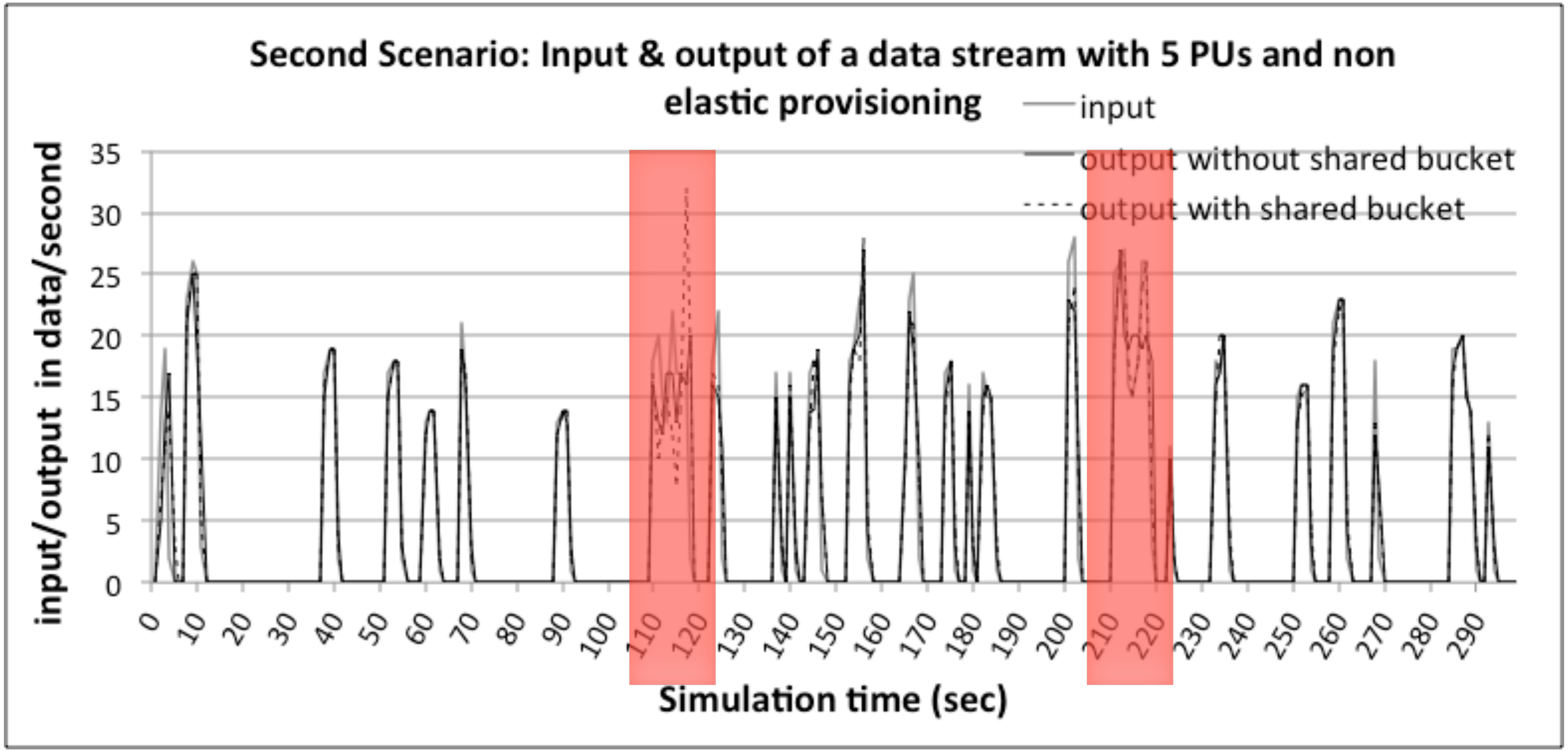


# Simulation results - 300s without elastic provisioning

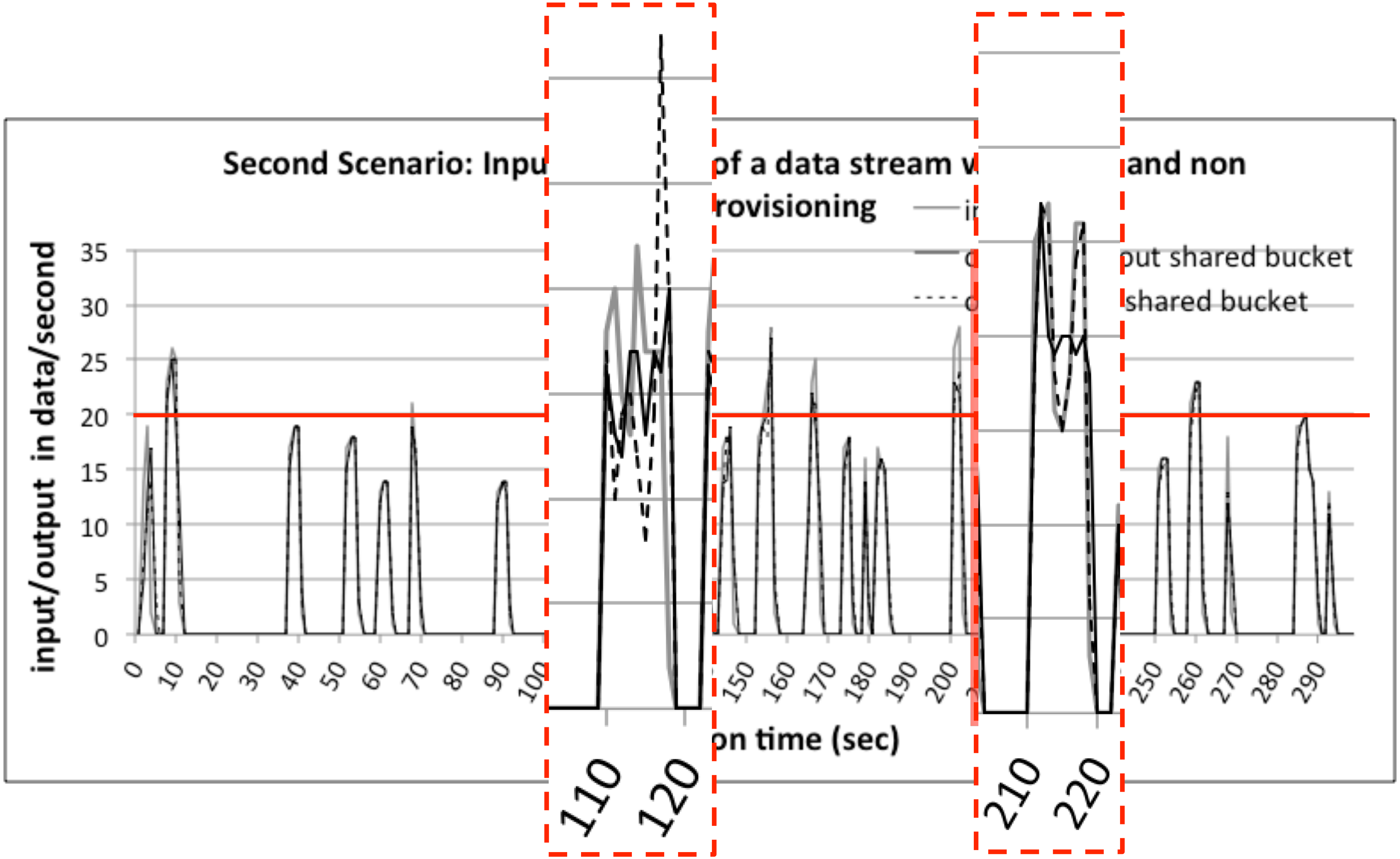




# Simulation results – throughput



# Simulation results – throughput



# On-going works: implementing the architecture

- Models are implemented and simulated with **Renew**
- Renew can be used as the **control logic** and to **execute commands** for various purposes:
  - Execute any UNIX commands
  - Launch any script/executable (Python, Java, C,…)
- **OpenNebula** is used to deploy VMs
- Renew can call **OpenNebula command** to launch initial VMs, launch additional VMs on the same cloud (local resources), launch VMs on a remote cloud (remote resources), …
- Renew can execute **Java program** for sending, receiving, processing data chunks according to the control logic

# Conclusions

## Unified resource management with Token Bucket

- We extend our architecture with a unified and more flexible approach based to tokens management
- Redistribution of unused resources and redistribution of pre-allocated resources from low priority classes are **handled consistently** with the Token Bucket main regulation & isolation mechanism
- **Business rules** can complement optimization process to improve revenue

## Revenue-centric model

- Adding new resources (local or remote) as well as optimizing pre-allocated resources can be classified according to their cost
- Global revenue taking into account all possible resource-related actions can be maximized when using the proposed unified resource magement approach

José Ángel Bañares
Rafael Tolosana
Engineering & Architecture School - Zaragoza University, Spain
banares@unizar.es, rafaelt@unizar.es

Omer F. Rana
School of Computer Science & Informatics - Cardiff University, UK
o.f.rana@cs.cardiff.ac.uk

Congduc Pham
Laboratoire informatique- Université de Pau, France
congduc.pham@univ-pau.fr

# Revenue creation for rate adaptive stream management in multi-tenancy environments

GECON 2013