University of Stavanger

Broker Overlay for Decentralized Grid Management

Abdulrahman Azab

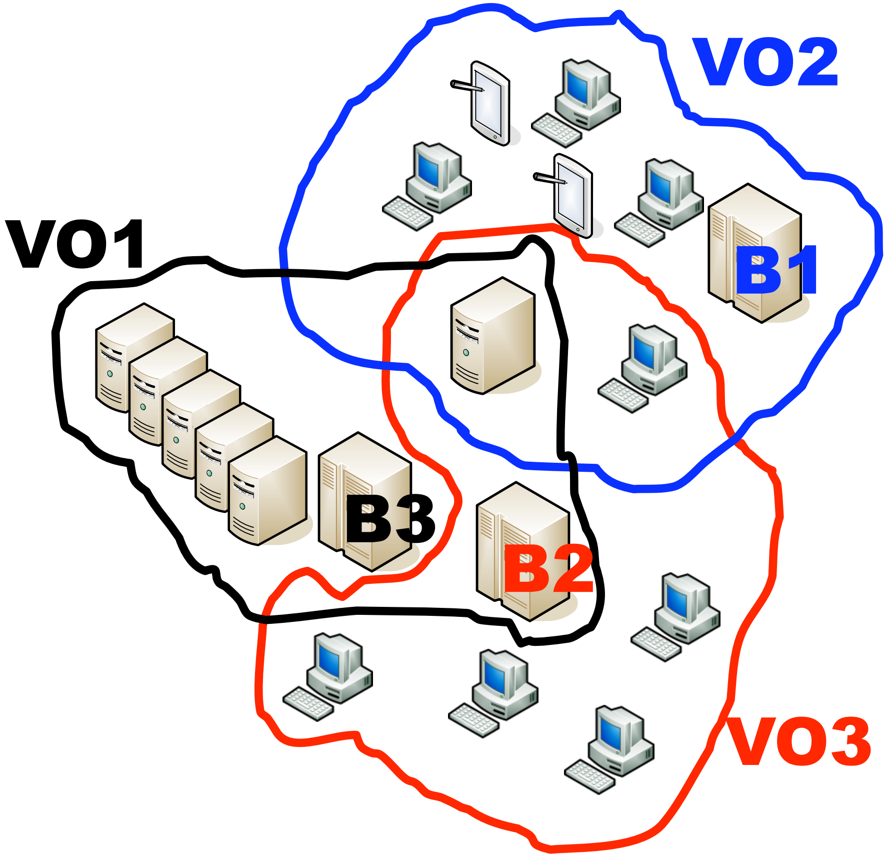
abdulrahman.azab@uis.no

University of Stavanger

What is Grid?

"Grid computing is concerned with coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations."

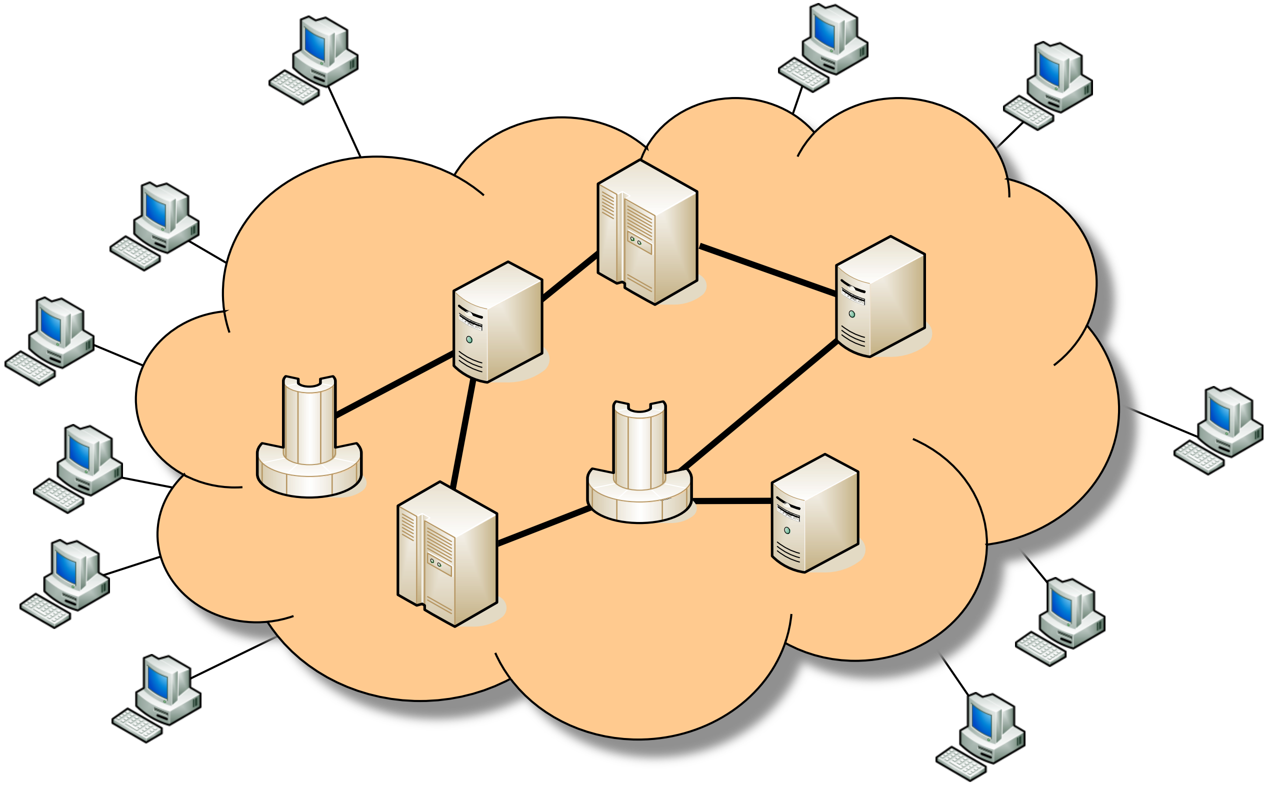
Ian Foster & Karl Kesselman , 2001.



What is Cloud?

“A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet”

Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu 2008

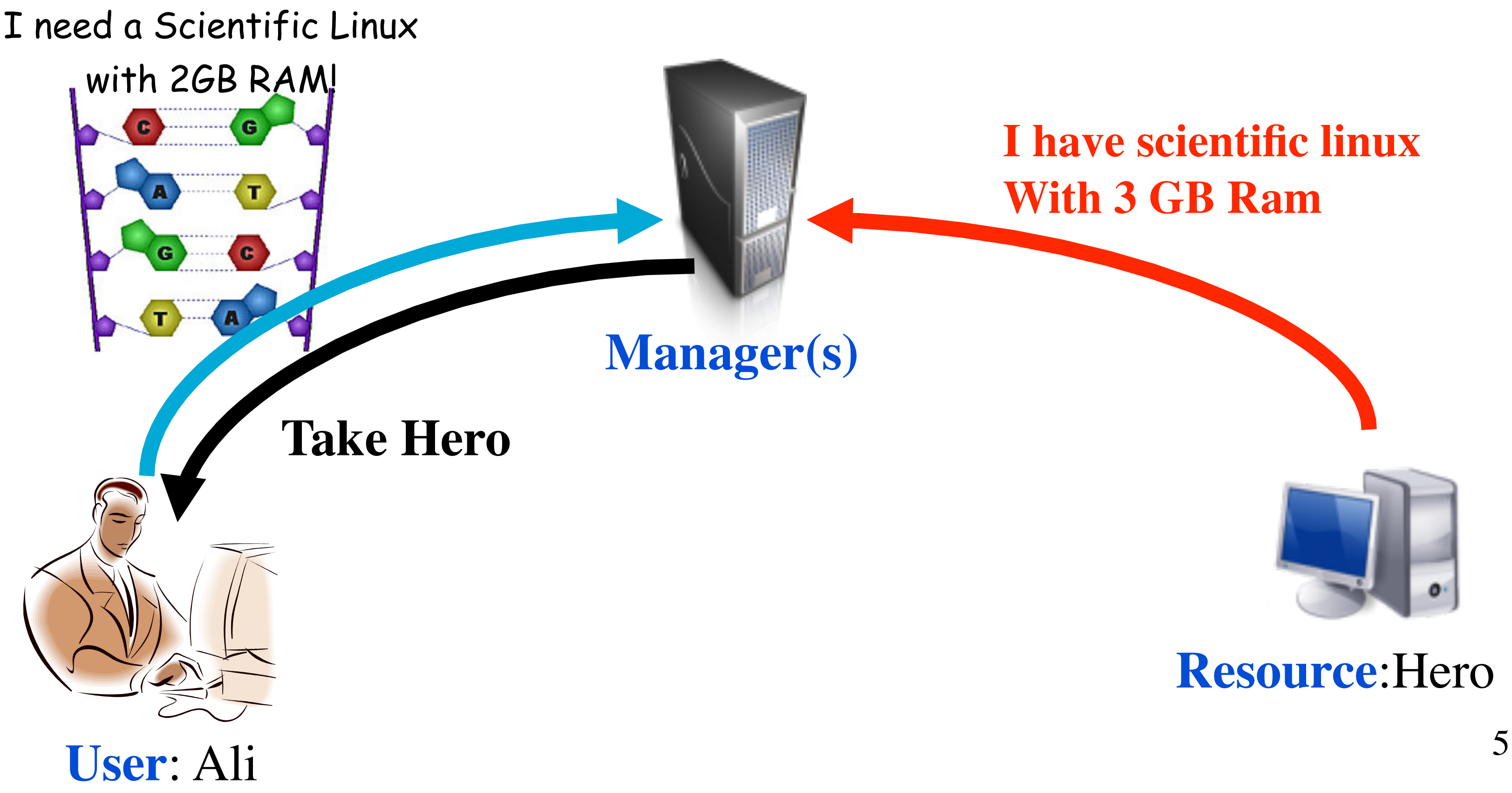


The Kiss Rule

Keep it simple, stupid!

Grid vs Cloud

- Grid



I need a Scientific Linux with 2GB RAM!

I have scientific linux With 3 GB Ram

Manager(s)

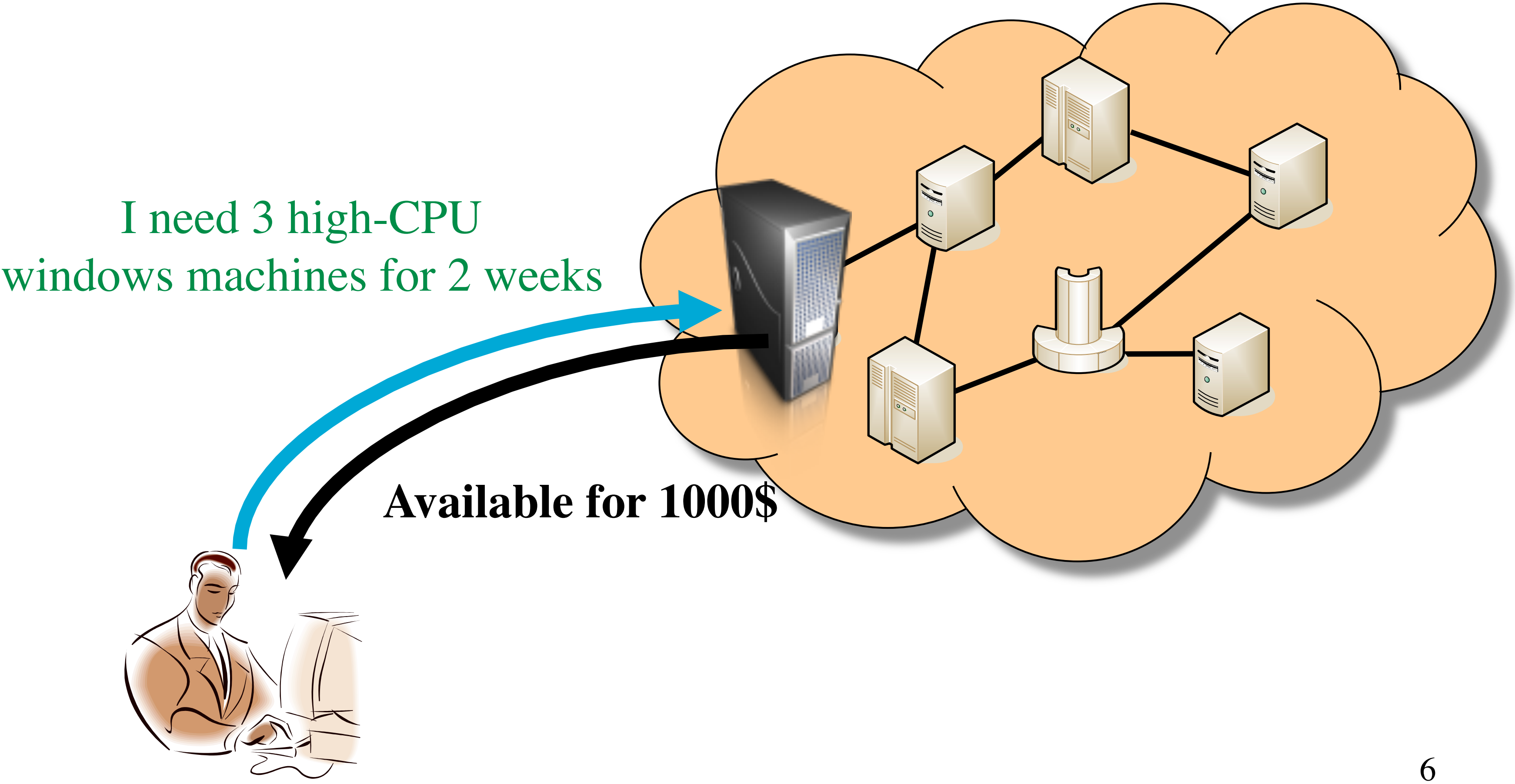
Take Hero

User: Ali

Resource:Hero

Grid vs Cloud

- Cloud



I need 3 high-CPU windows machines for 2 weeks

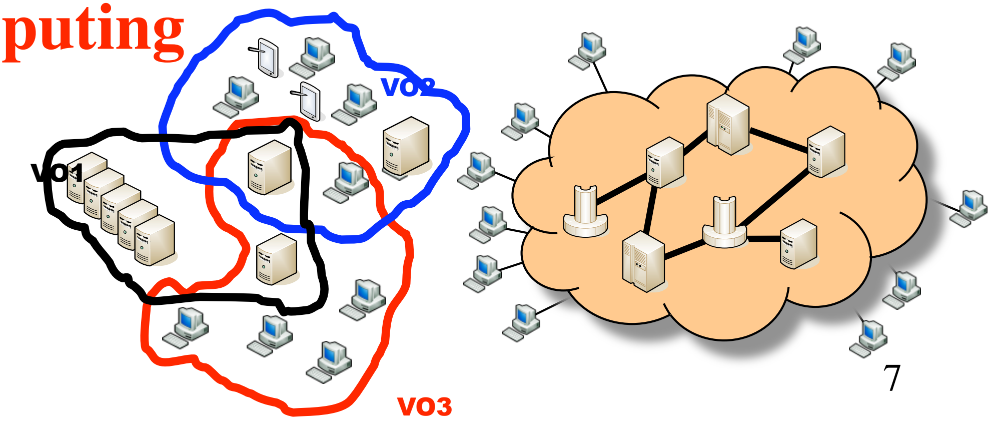
Available for 1000$

Computational Grid vs. Computational Cloud

	Computational Grid	Computational Cloud
Provided service	Computational power	
Amount of concurrent requests	Limited	Massive
Transparency	Not required	Required
Scalability	Limited	High

I don’t care.

Both are Distributed computing



Challenges

- Many, but we consider:

1. Stability with scalability

2. System transparency

Stability with Scalability

- Stability

Maintaining throughput under failures

- Scalability

Ability to add more nodes

- Stability with scalability

Maintaining throughput under failure with bigger Environment

-Achieve load balancing
-Avoid job starvation

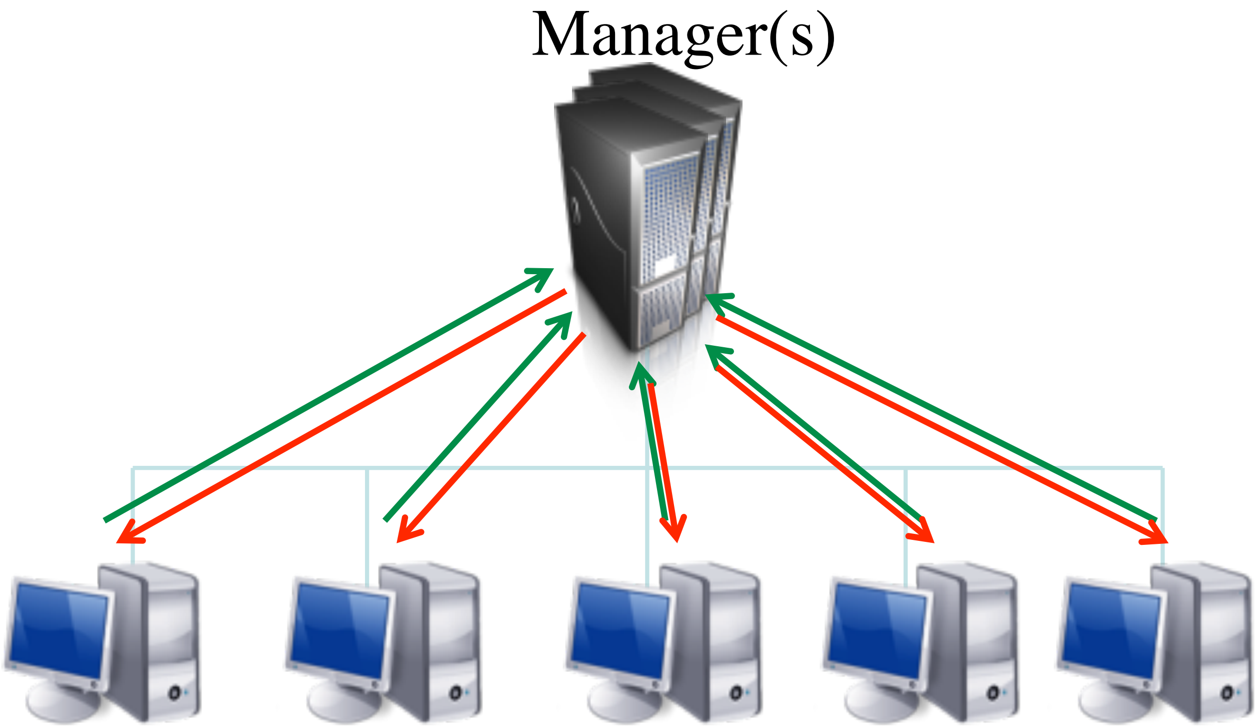
How?

- Optimized machine organization
- Efficient job scheduling
- Efficient fault tolerance

Machine organization

- Flat

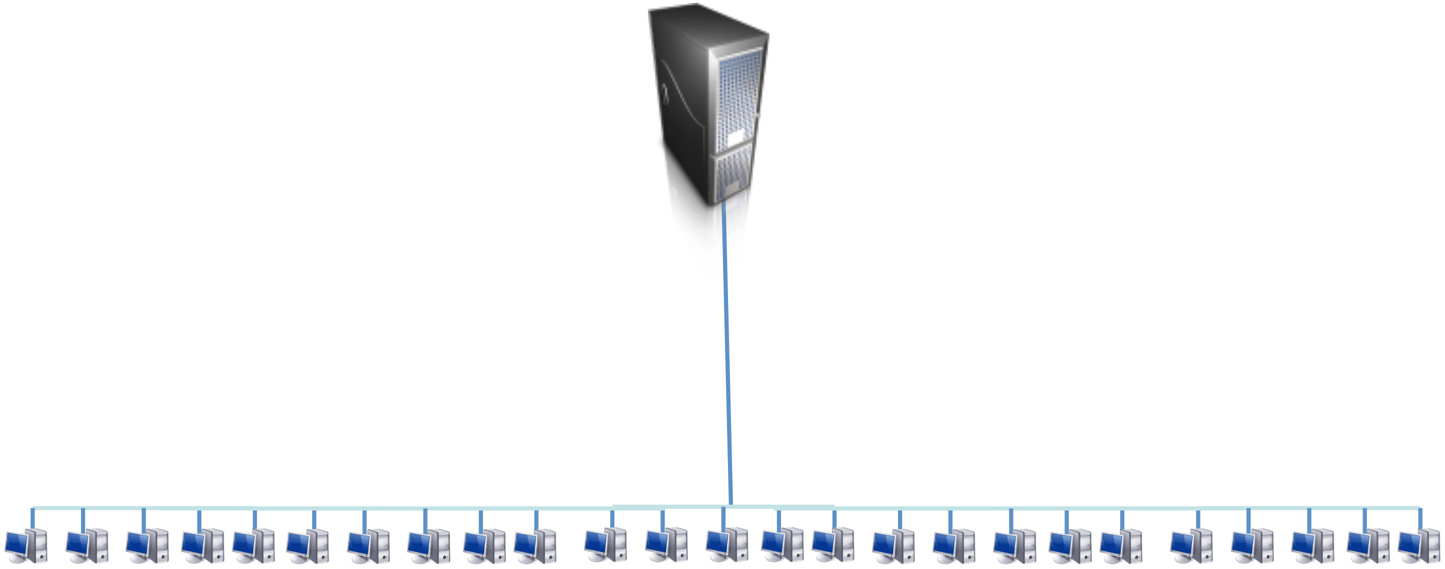
  (gLite, Condor, Globus,…)



Machine organization

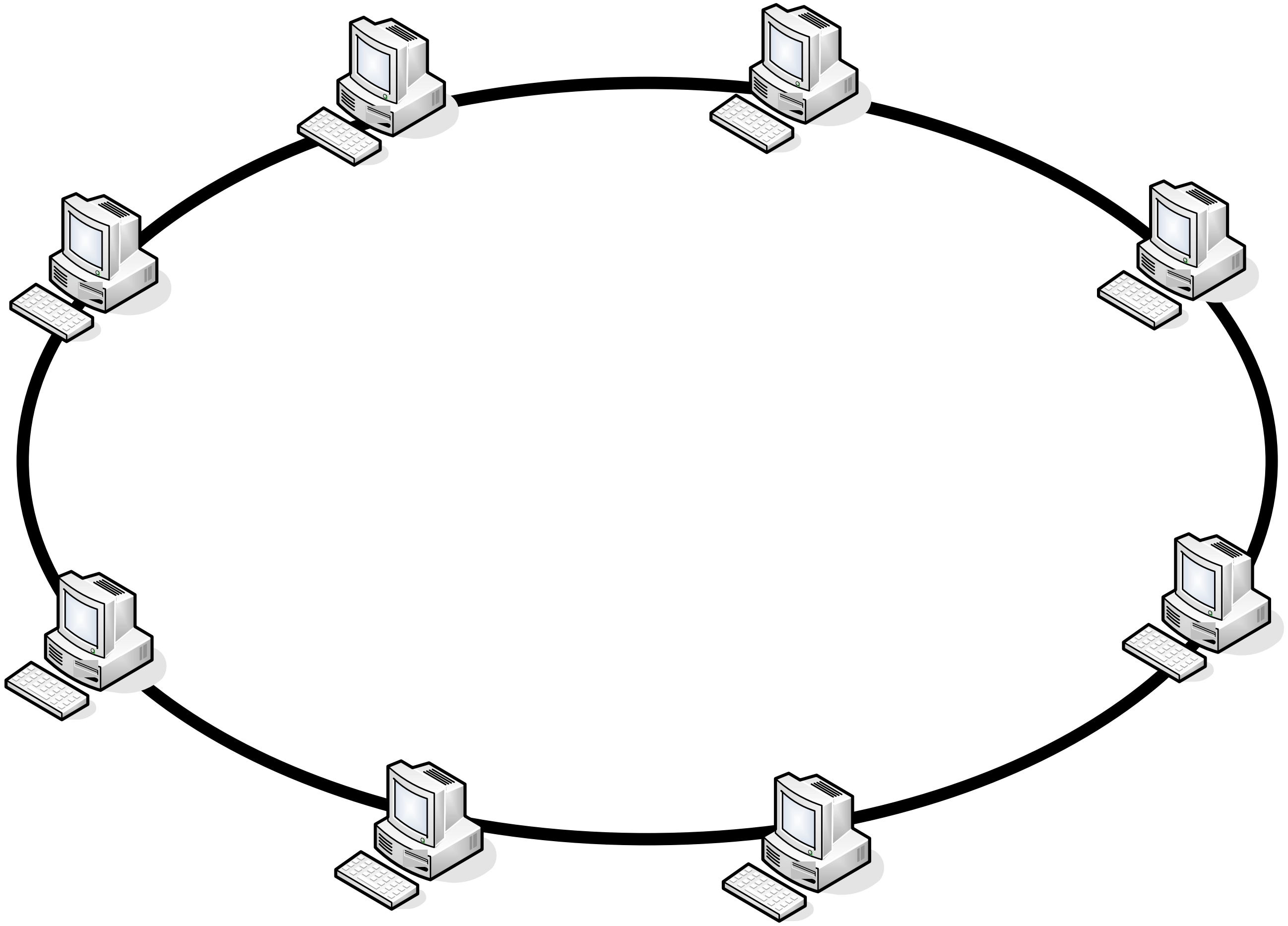
- Flat

  (gLite, Condor, Globus,...)



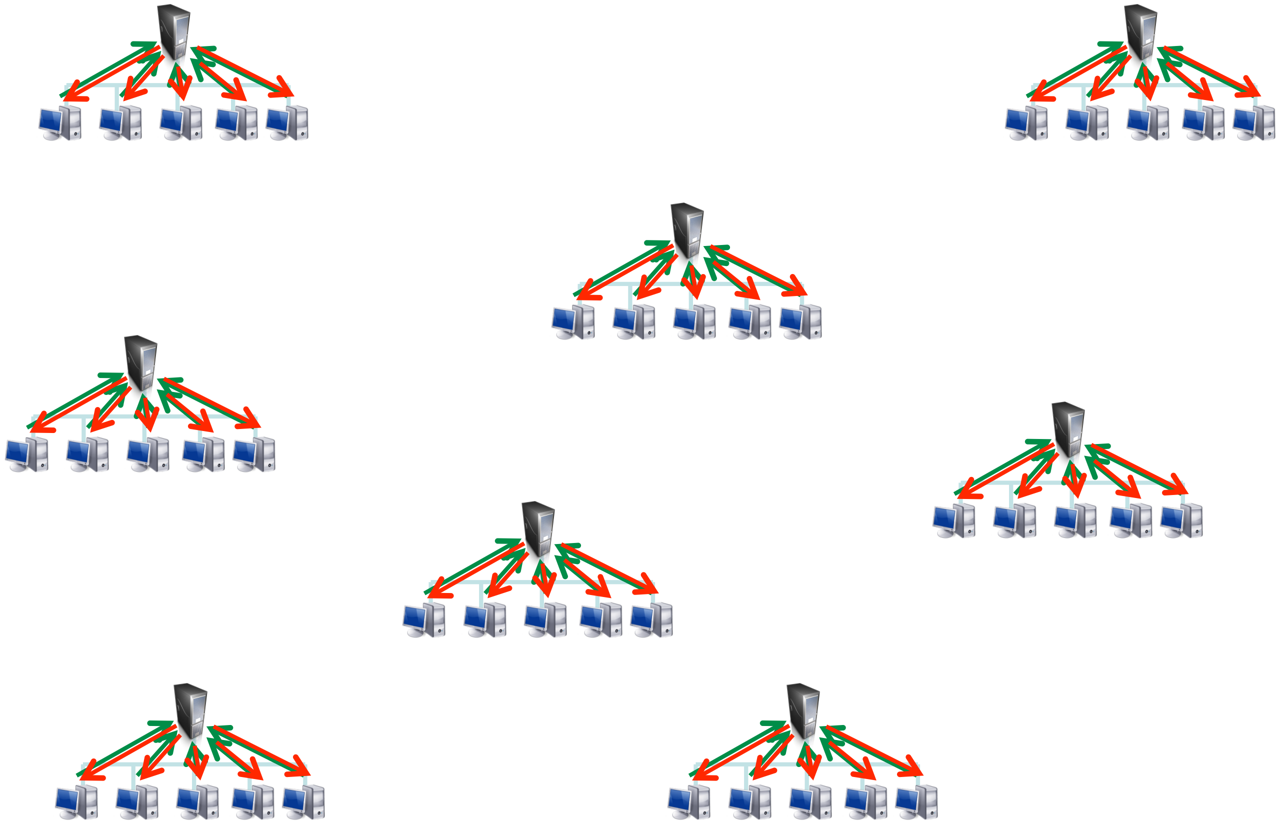
Machine organization

- Flat
  (NorduGrid, HIMAN, XtreemOS)



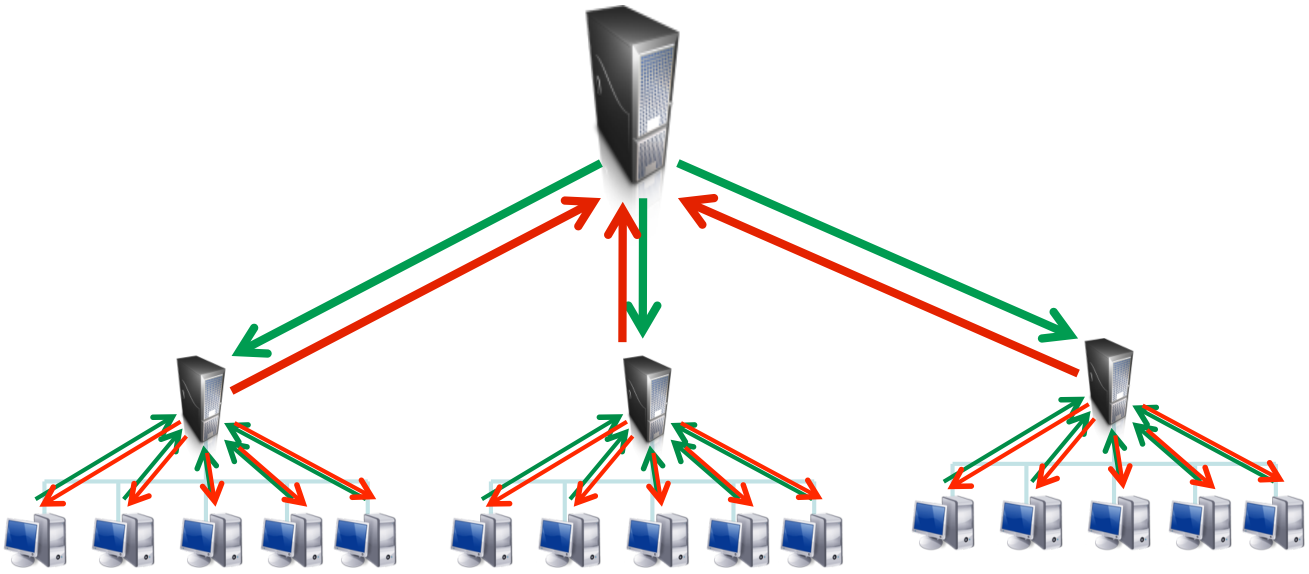
Machine organization

- Flat



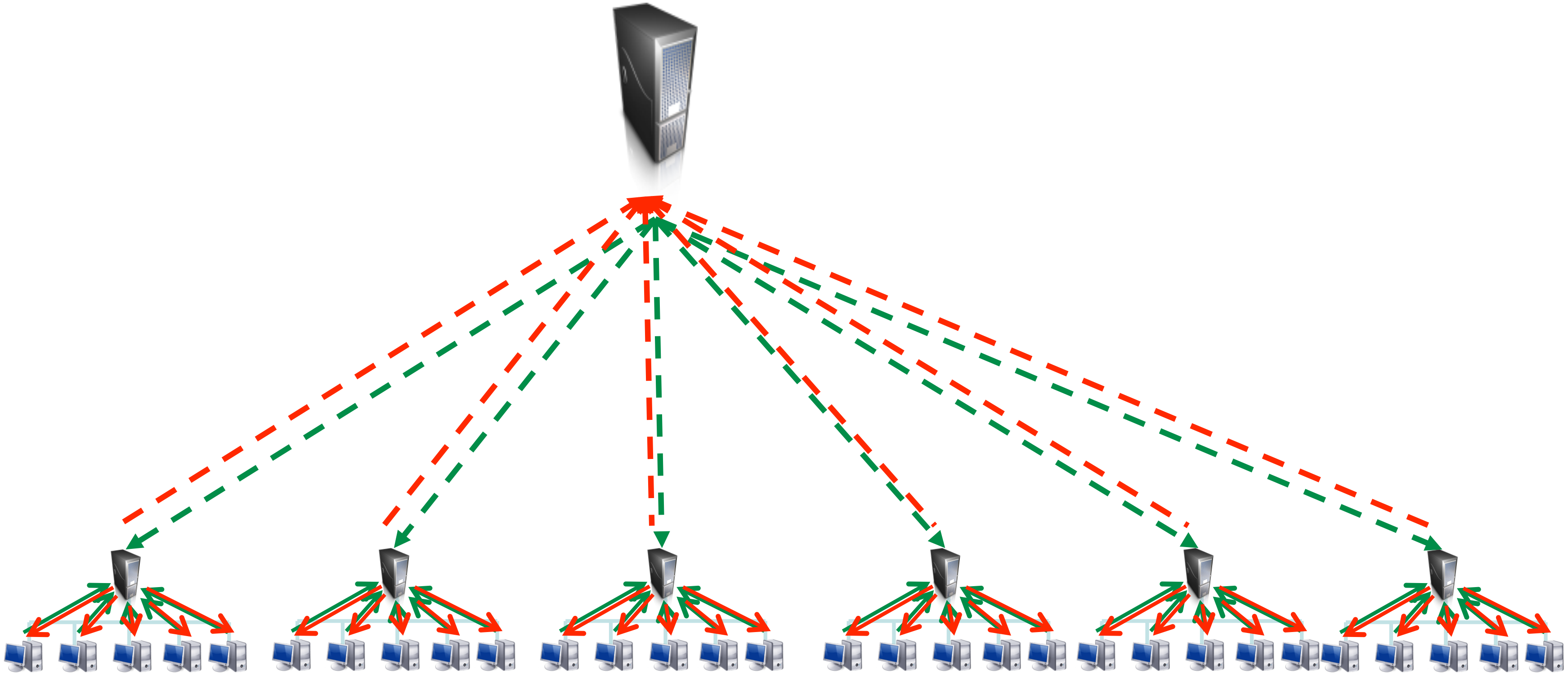
Machine organization

- Hierarchical
  (UNICORE, GridWay, BOINC,...)



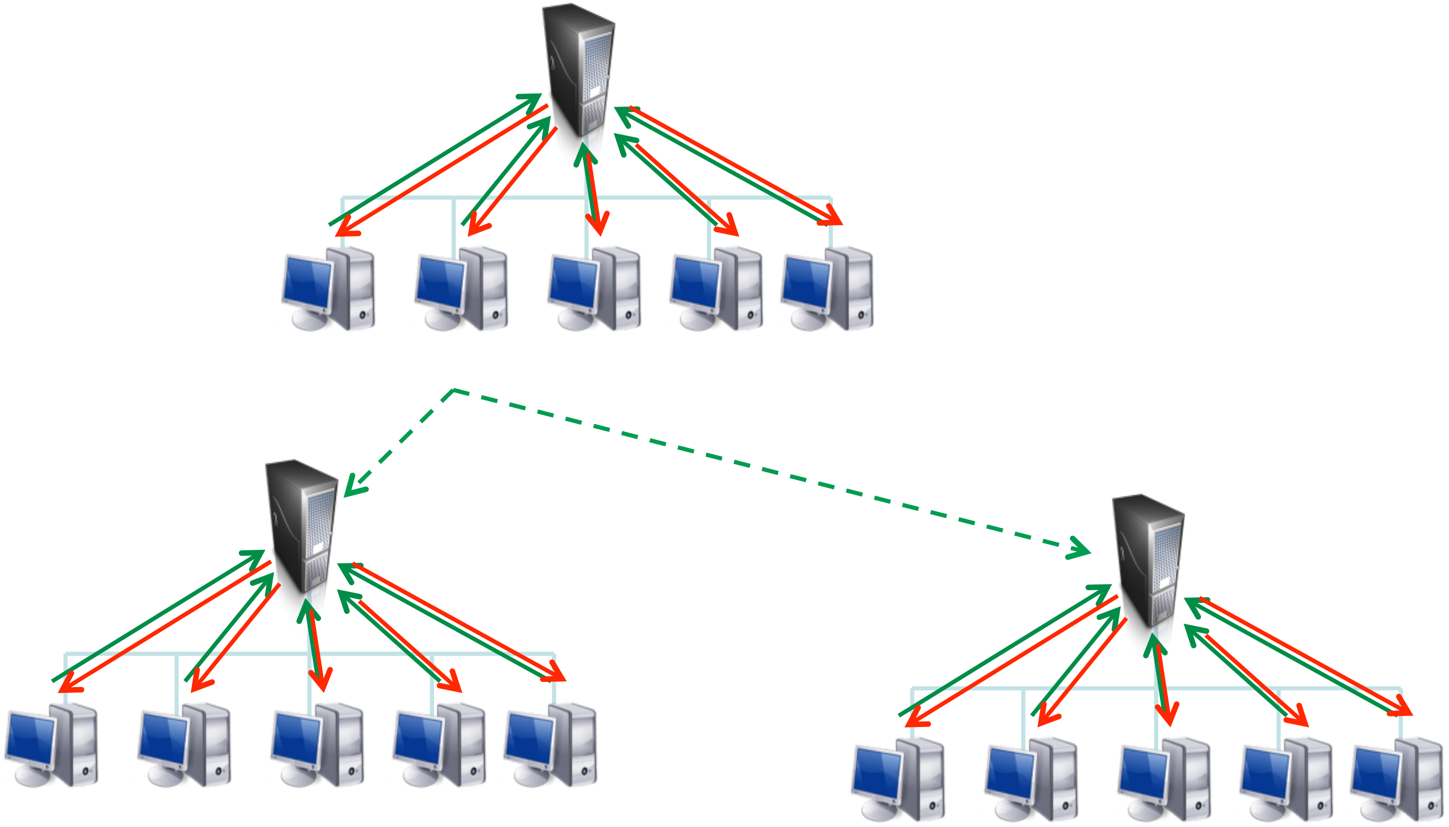
Machine organization

- Hierarchical (UNICORE, GridWay, BOINC,...)



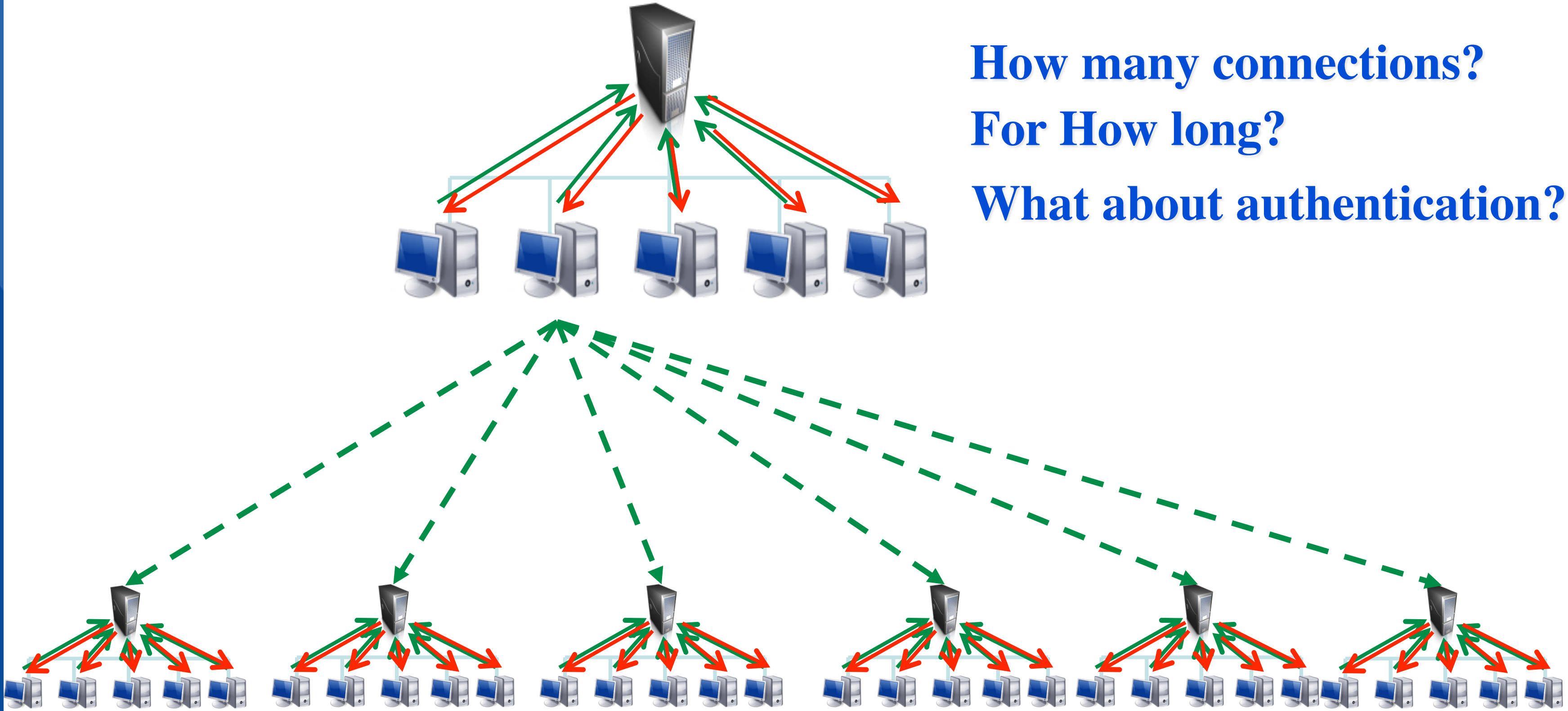
Machine organization

- Interconnected (Condor (flocking), DEISA, EGEE, NorduGrid)



Machine organization

- Interconnected
  (Condor (flocking), DEISA, EGEE, NorduGrid)



How many connections?

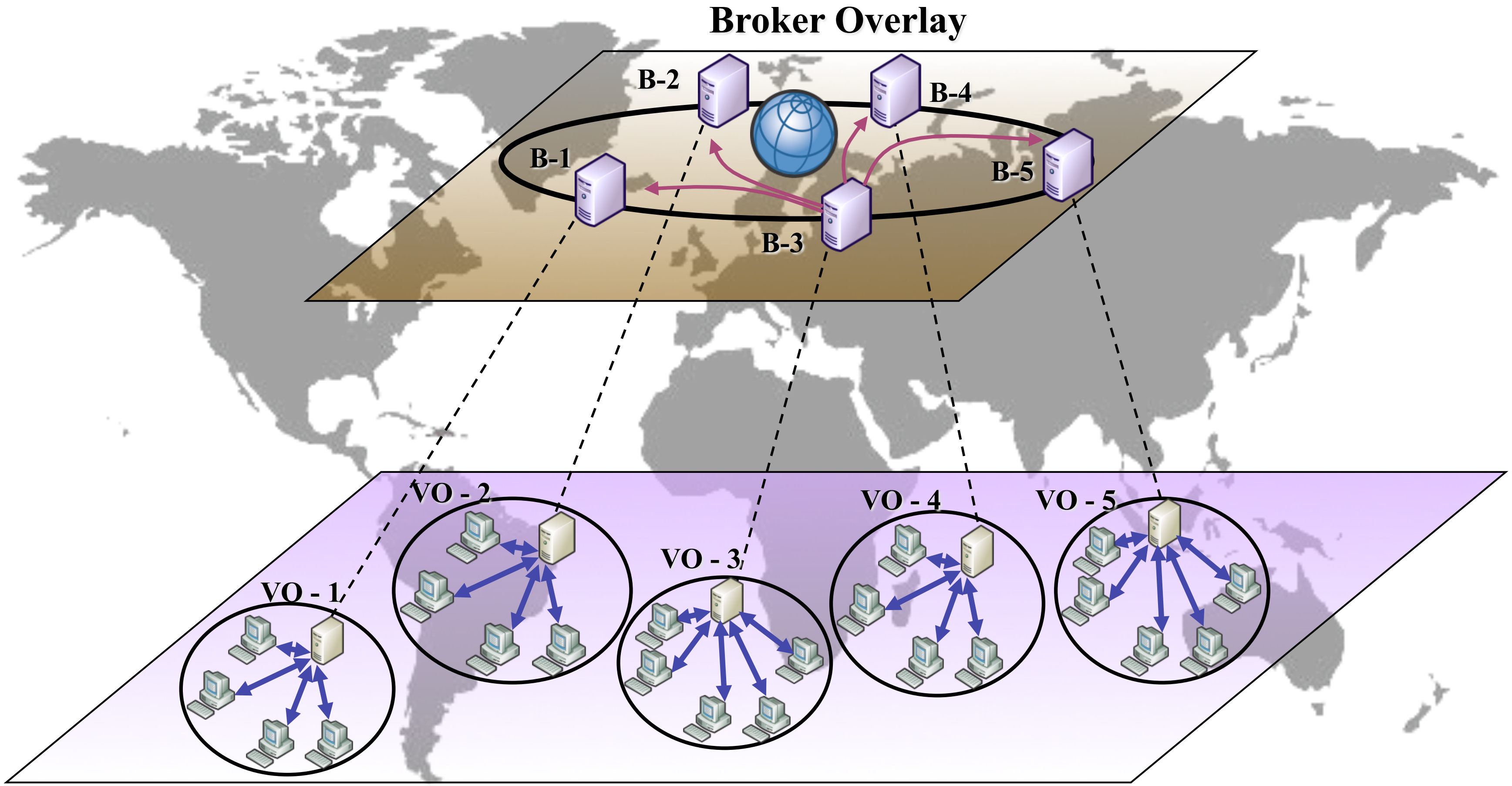
For How long?

What about authentication?

University of Stavanger

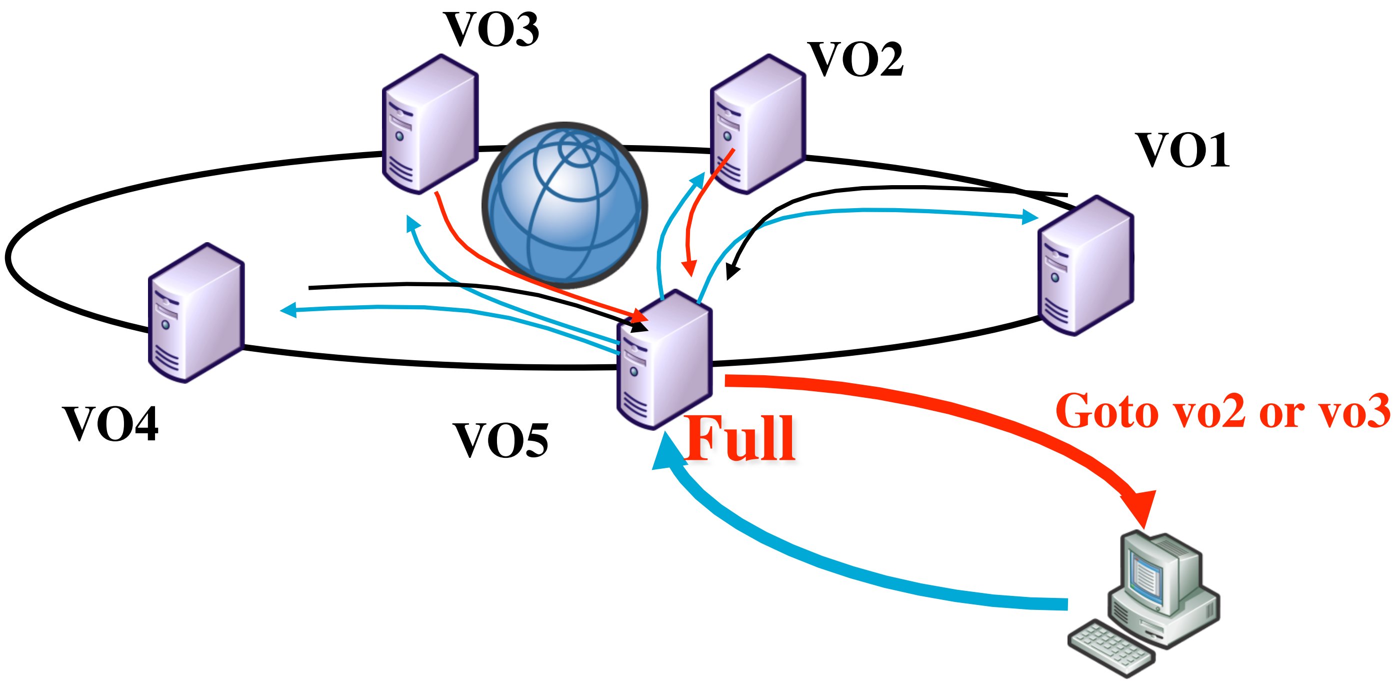
Proposal

Machine Organization: Cell

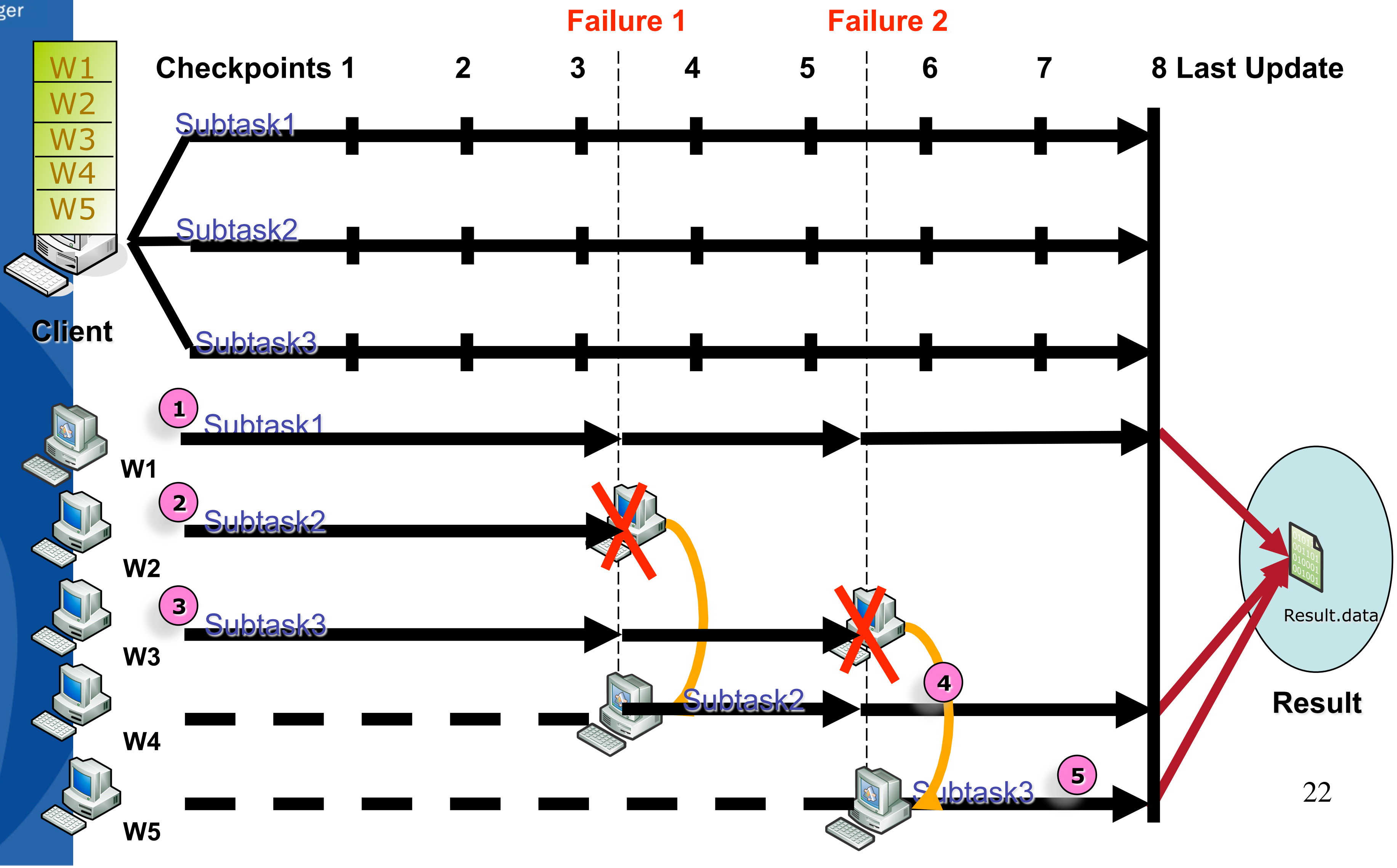


Scheduling: Cooperative

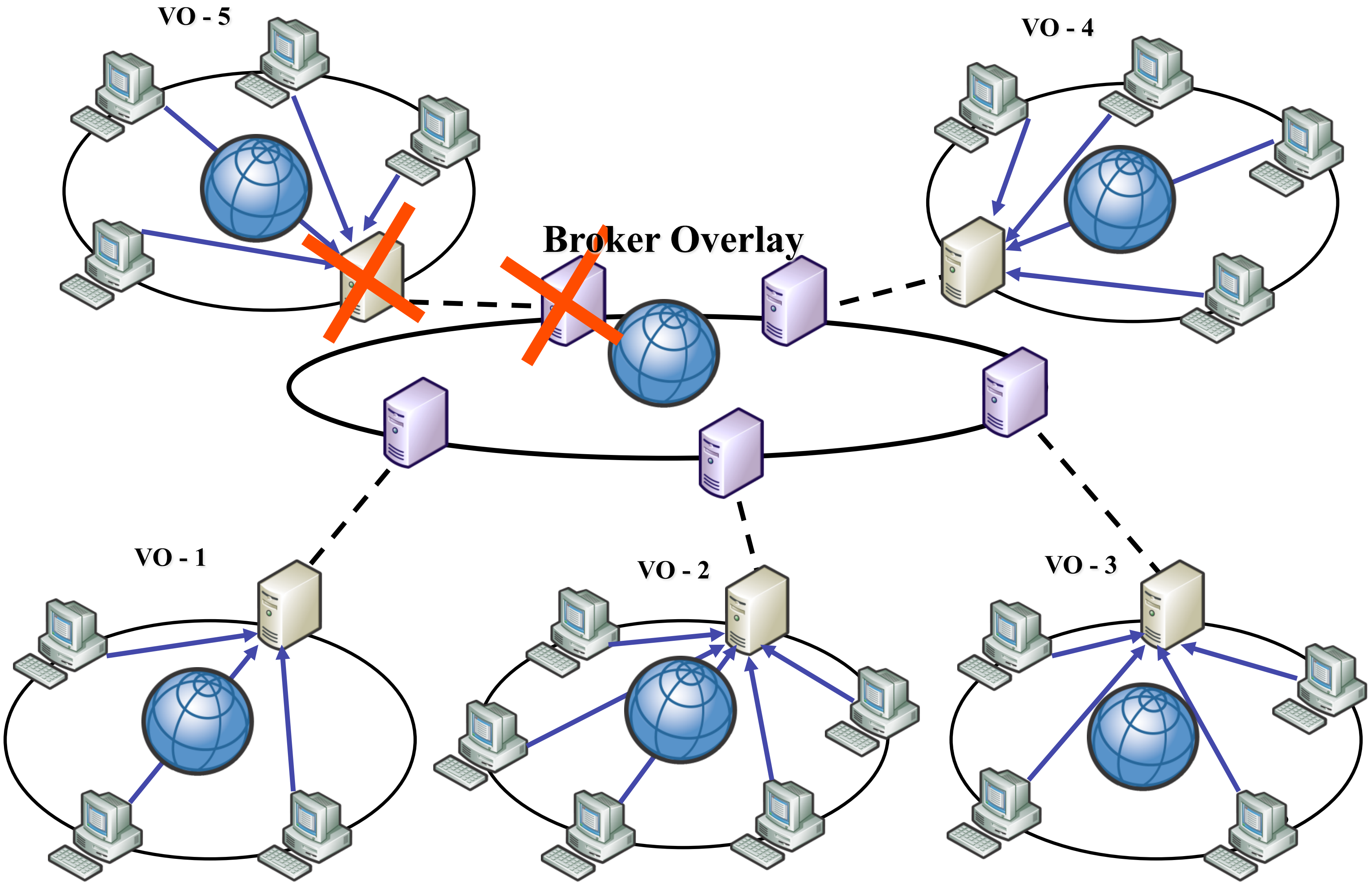
- Minimize scheduling overhead using Fuzzy logic



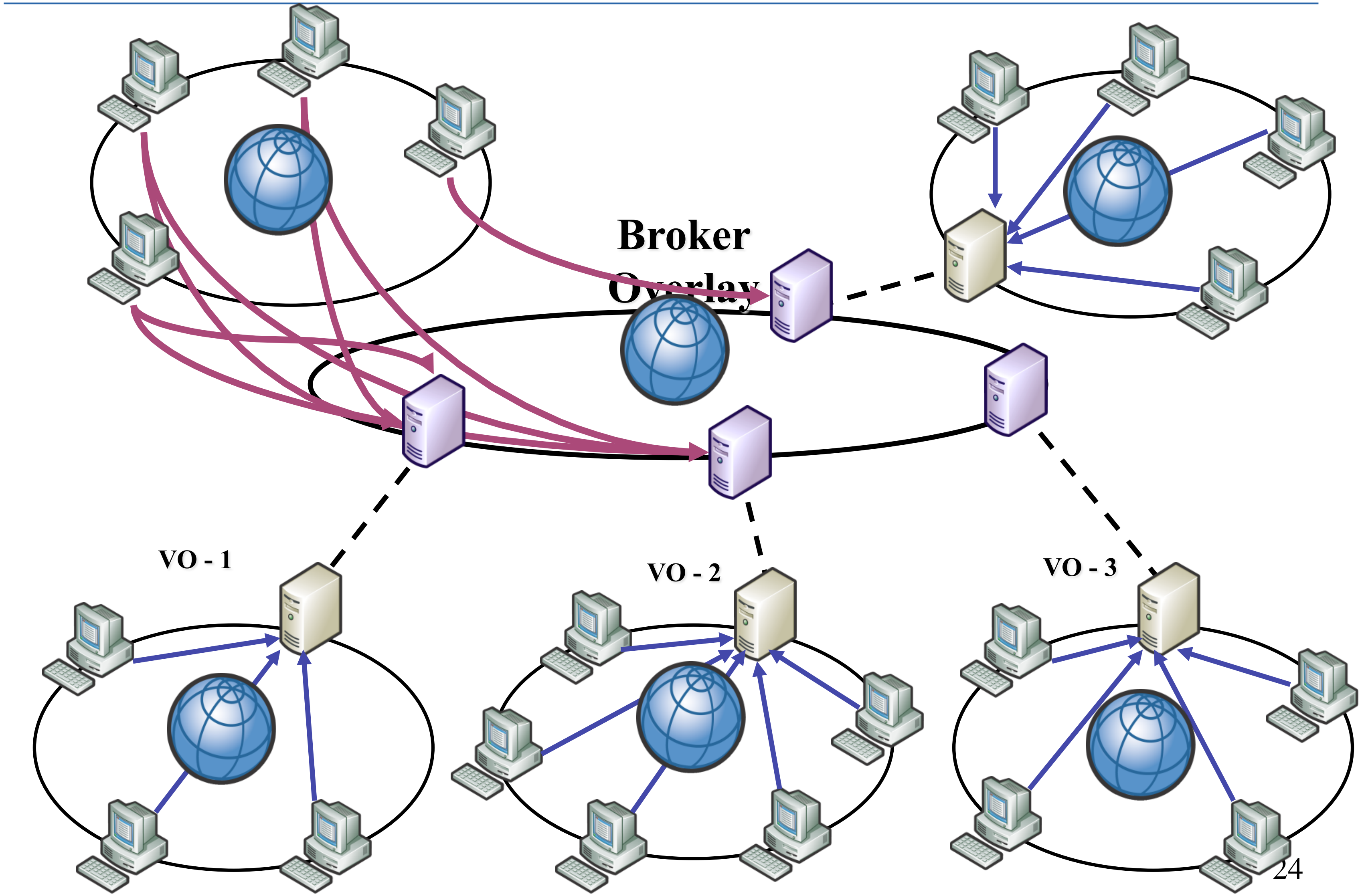
Worker Failures



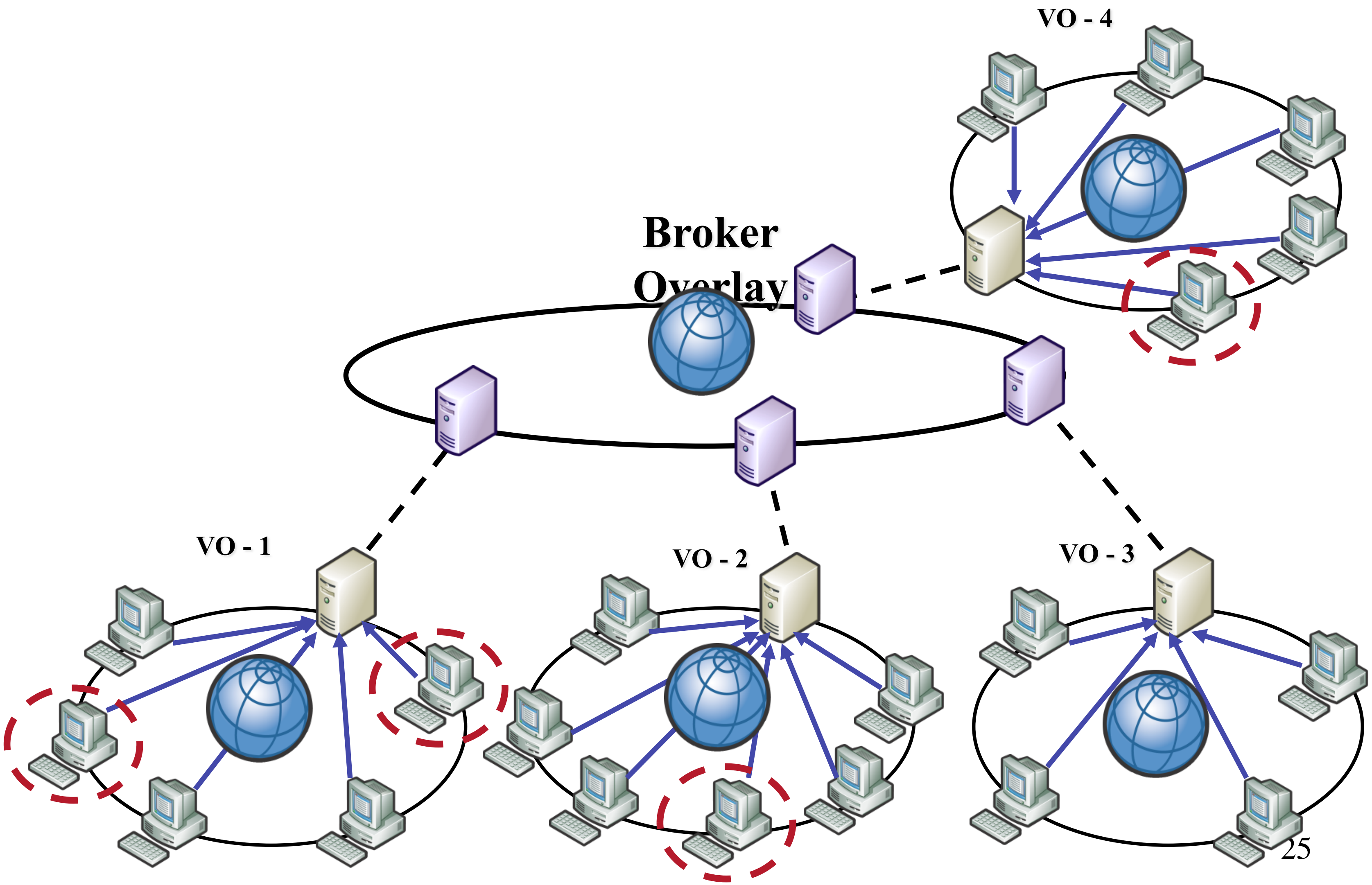
Broker Failures



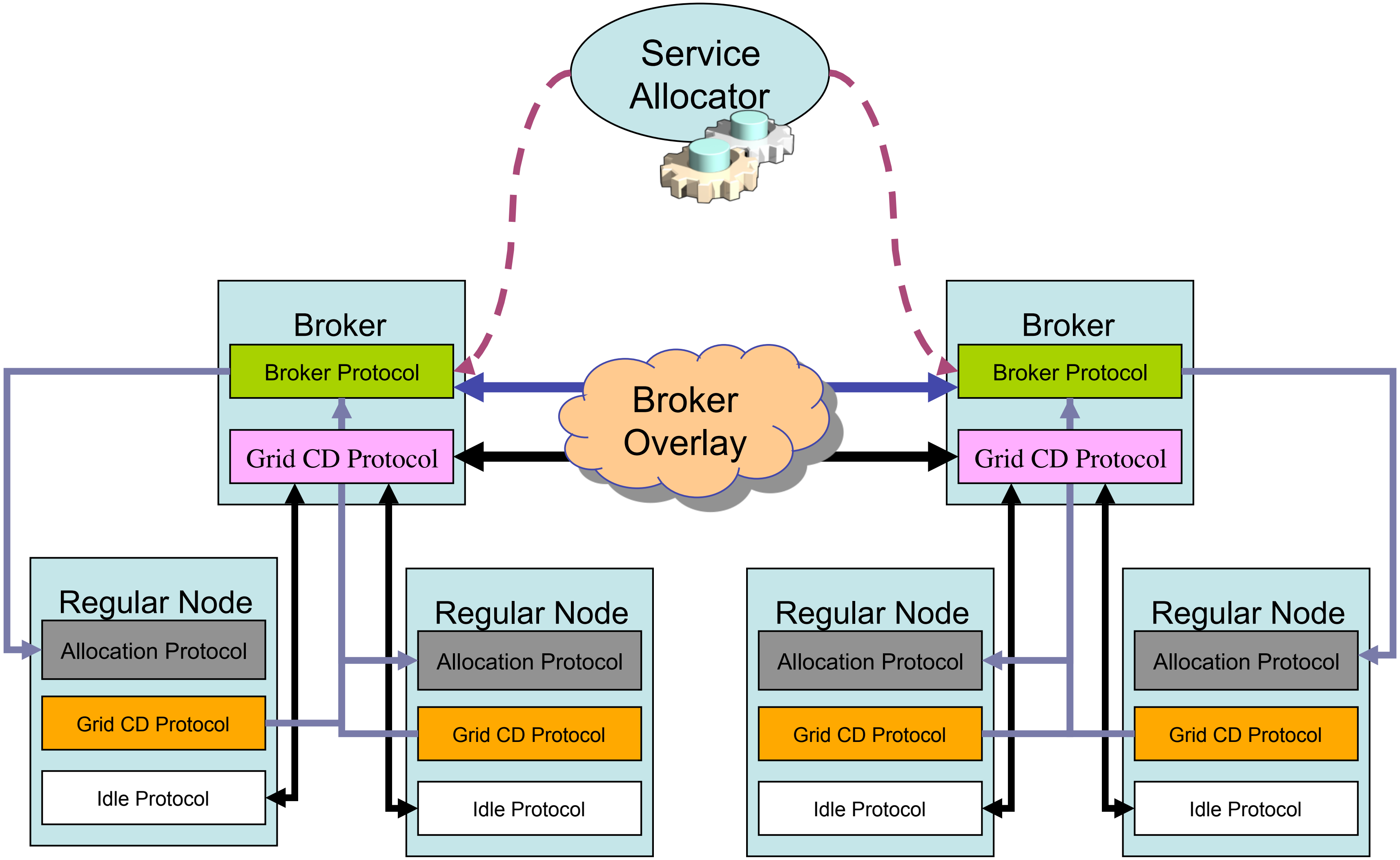
Broker Failures



Broker Failures



Simulation Model: PeerSim



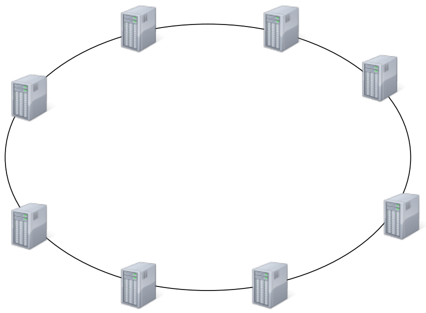
Performance Evaluation

- Validity of the stored resource information.
- Efficiency of service allocation.
- Impact of broker failure on resource information updating.

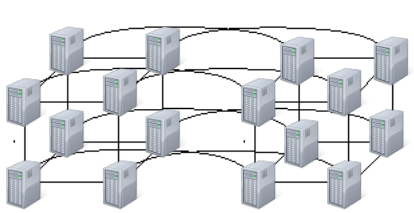
N → Total Grid size, M → Number of VOs

Performance Evaluation

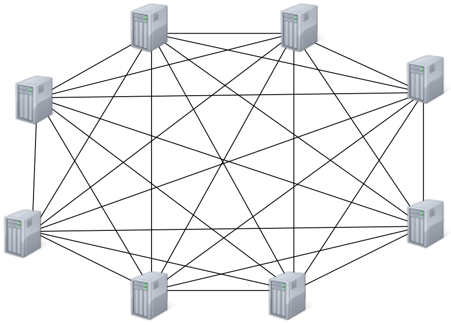
- Broker Overlay Topologies



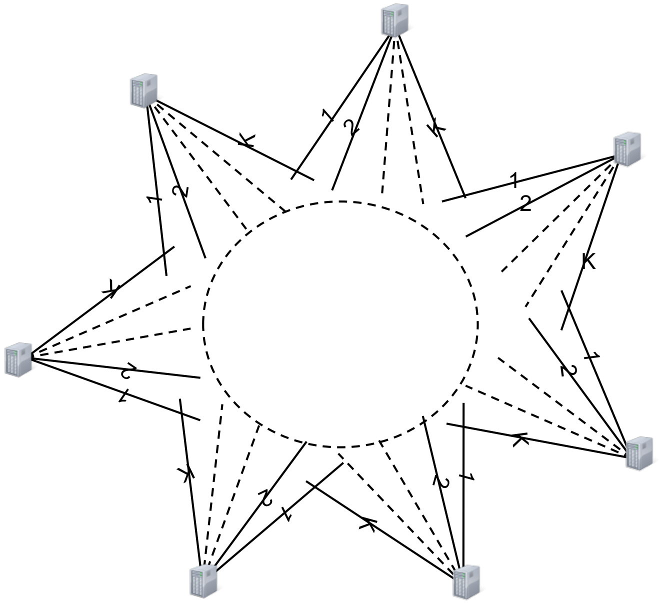
Ring



Hyper-Cube



Fully connected



Wire-*k*-out

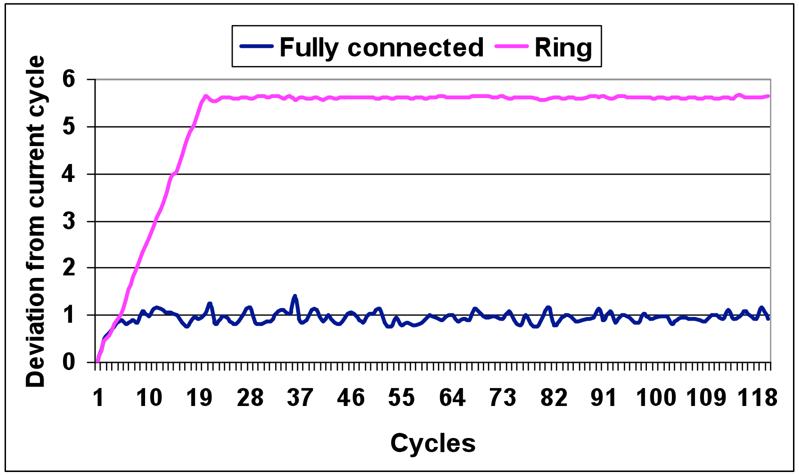
Validity of the stored resource information

- The deviation of the reading time values of RIDBs stored in the resource information data set, from the current cycle in a broker, with the simulation cycles.

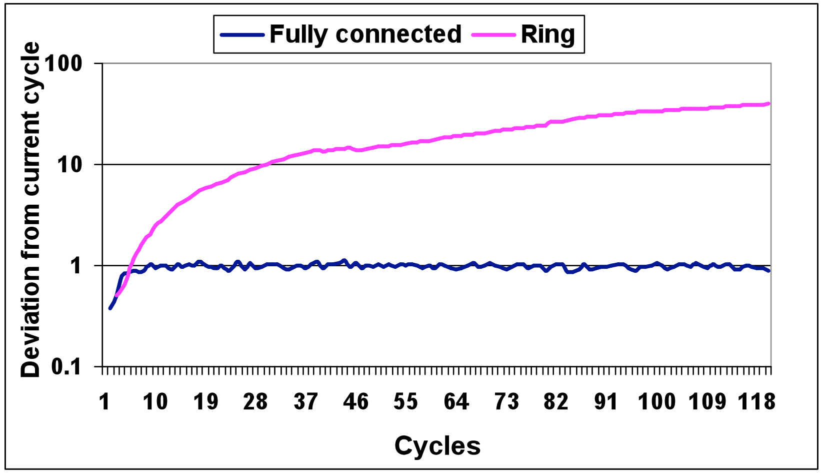
- The deviation value for cycle (c):

$$D(c) = \sqrt{\sum_{i=1}^{N} \frac{(Time(RIDB(i)) - c)^2}{N}}$$

Validity of the stored resource information



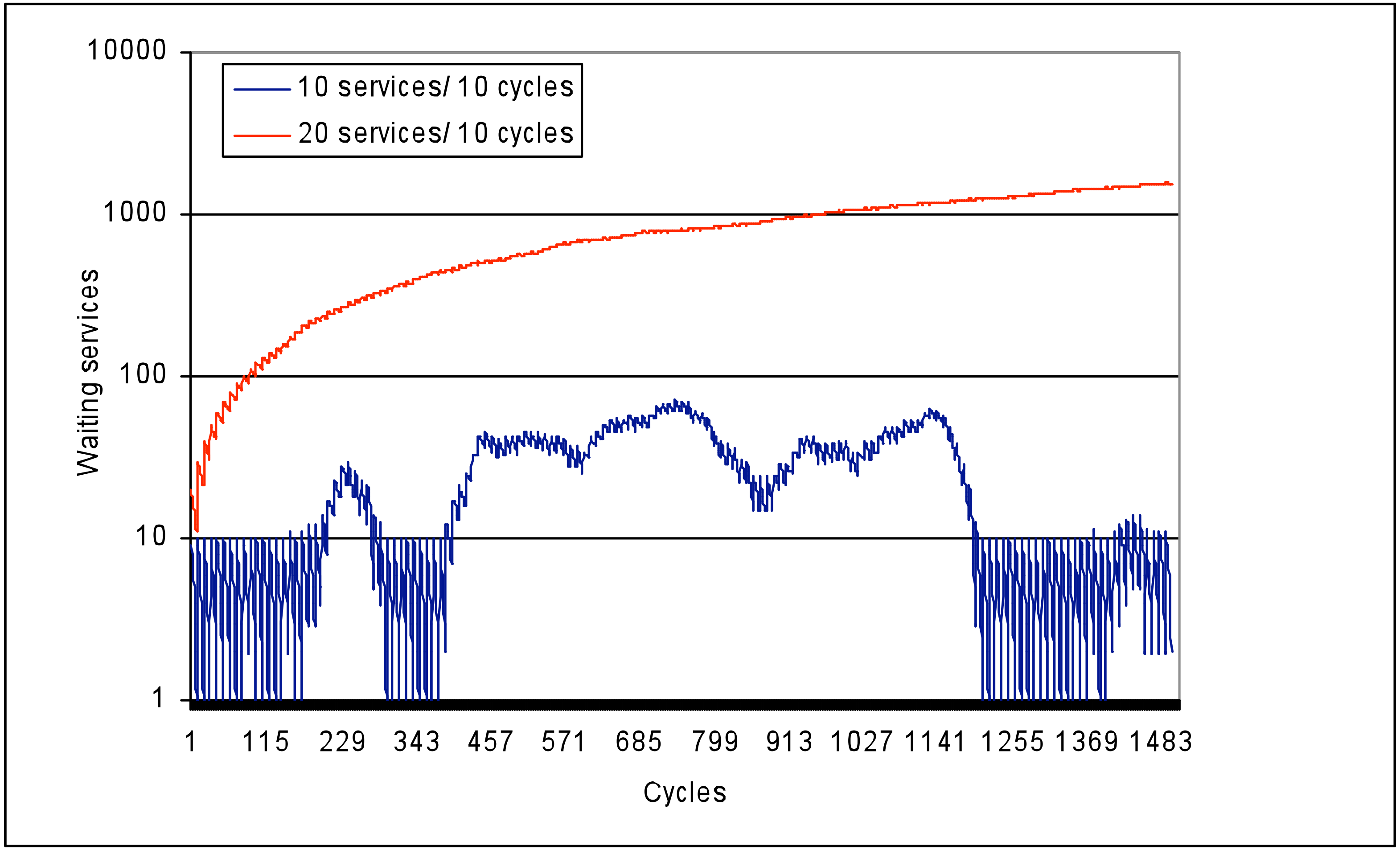
N = 100, M = 20



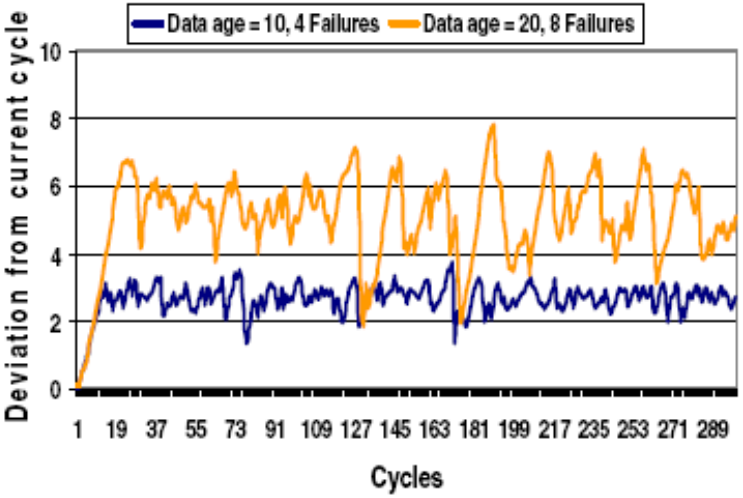
N = 500, M = 100 (log scale)

Efficiency of Job Allocation

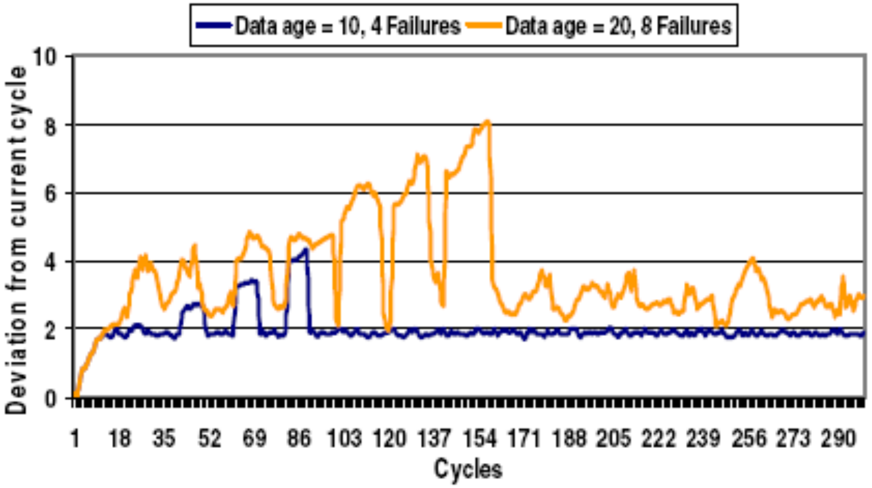
- One broker periodical allocation.



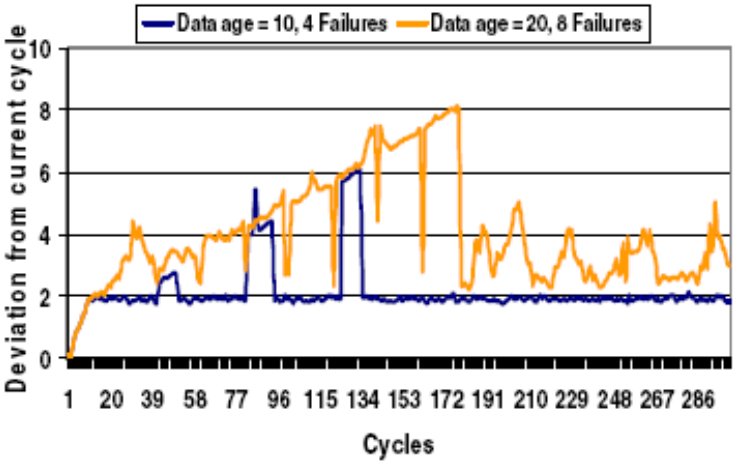
Impact of Broker Failures on Resource Information Updating **(N = 500, M = 100)**



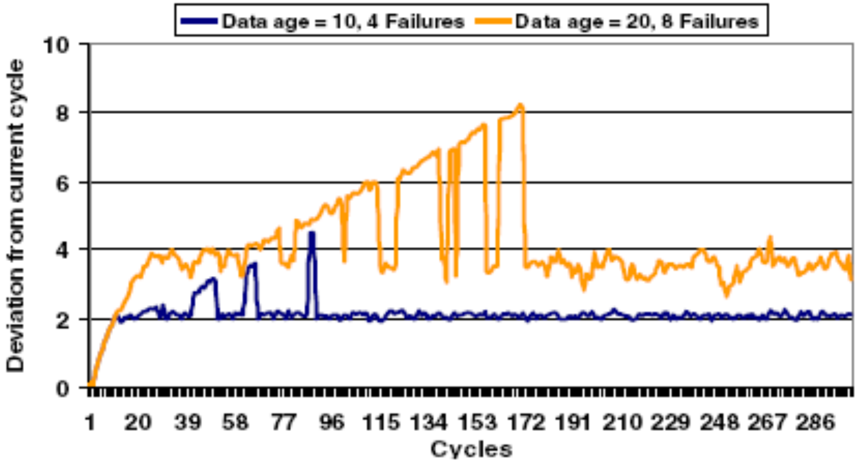
a) Ring broker overlay topology



b) Fully Connected broker overlay topology

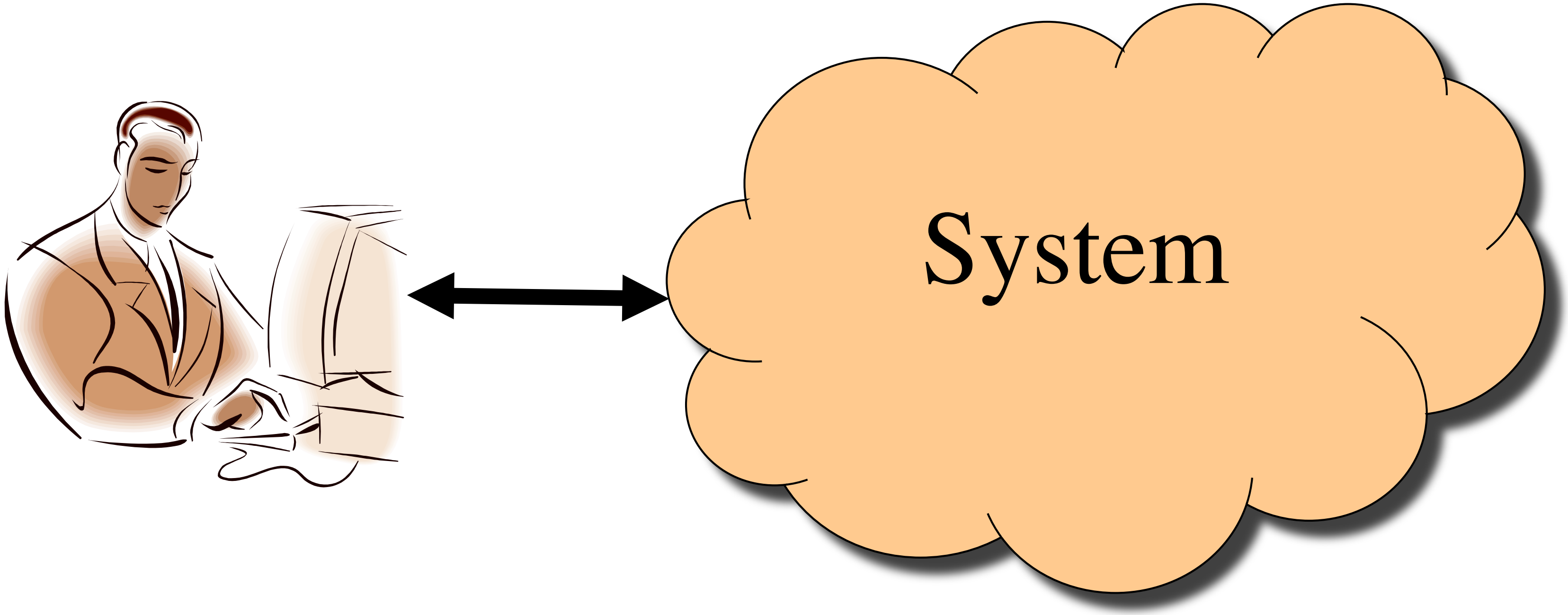


c) Wire-*k*-Out broker overlay topology, $k = 60$



d) Hyper-cube broker overlay topology

System Transparency



Challenge

- To submit jobs to a Grid system you need to learn how to:

1. Prepare your input files
2. Write a detailed submission script.
3. Submit your jobs through the front end.
4. Monitor the execution.
5. Collect the results.

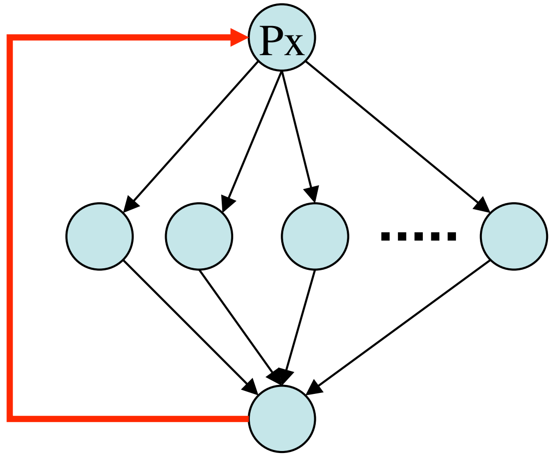
Example for 2: condor_submit

Do scientists have time for this ?

Current solutions

- **Grid portals** (Web-based gateways)
  WebSphere, WebLogic, GridSphere, GridPortlets,..

➢ Useful for manual submission. In many cases, it is required to perform job submission automatically from a user code.



Current solutions

- **Web services**

  Birdbath (condor), GRAM (Globus), GridSAM, ..

Keep it simple, stupid!

- **APIs**

  DRMAA, SAGA, HiLA, CondorAPI, GridR, ..

➢ The programming language has to support the technology and the user must have the proper experience. This is not the case for many low level special purpose languages and most of the scientists

Our Solution: GAFSI

- **Grid Access File System Interface**

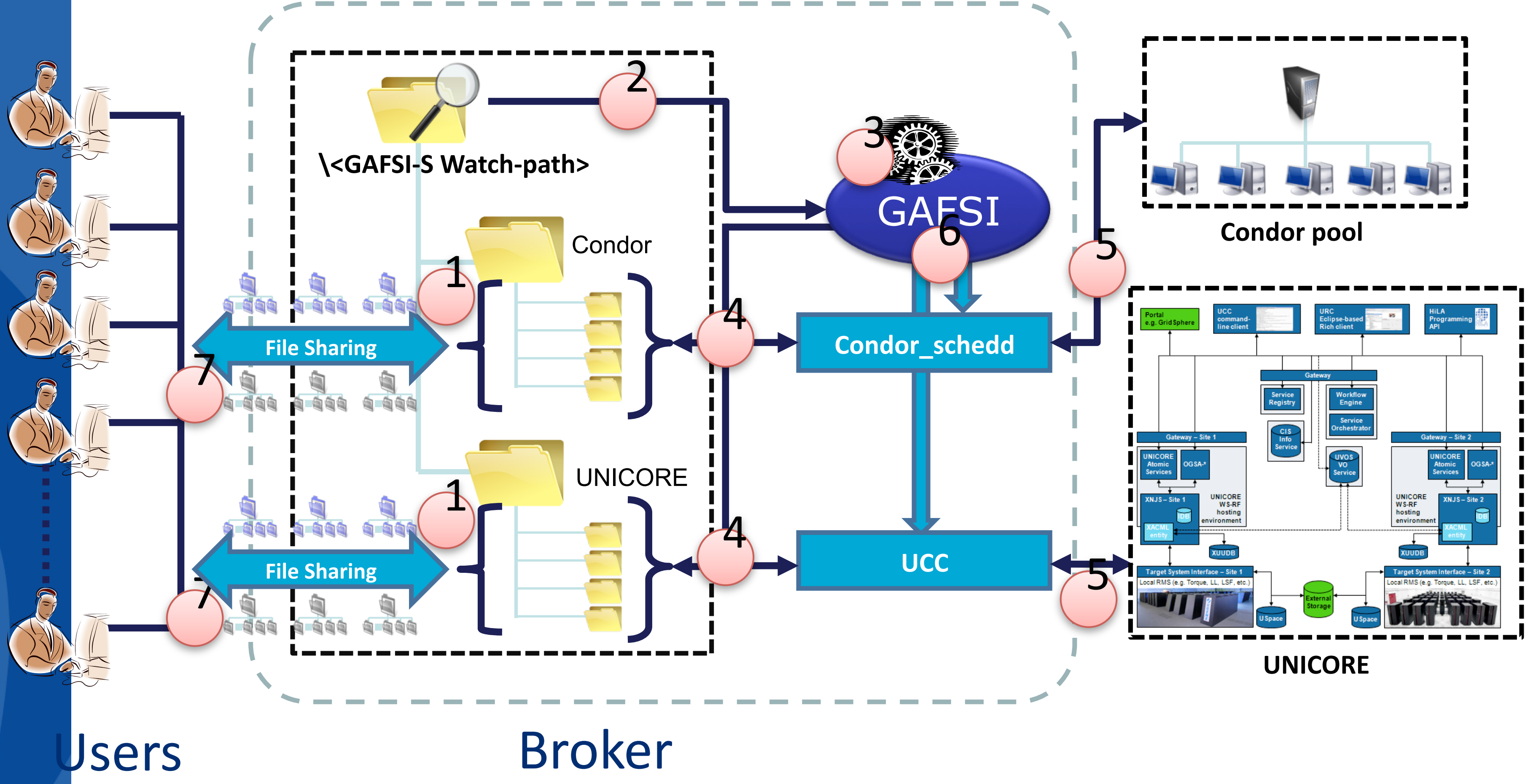
  submission and management of grid jobs is carried out by executing **simple** read() and write() file system commands.

- This technique allows all categories of users to submit and manage grid jobs both manually and from their codes which may be written in any language.

Demo

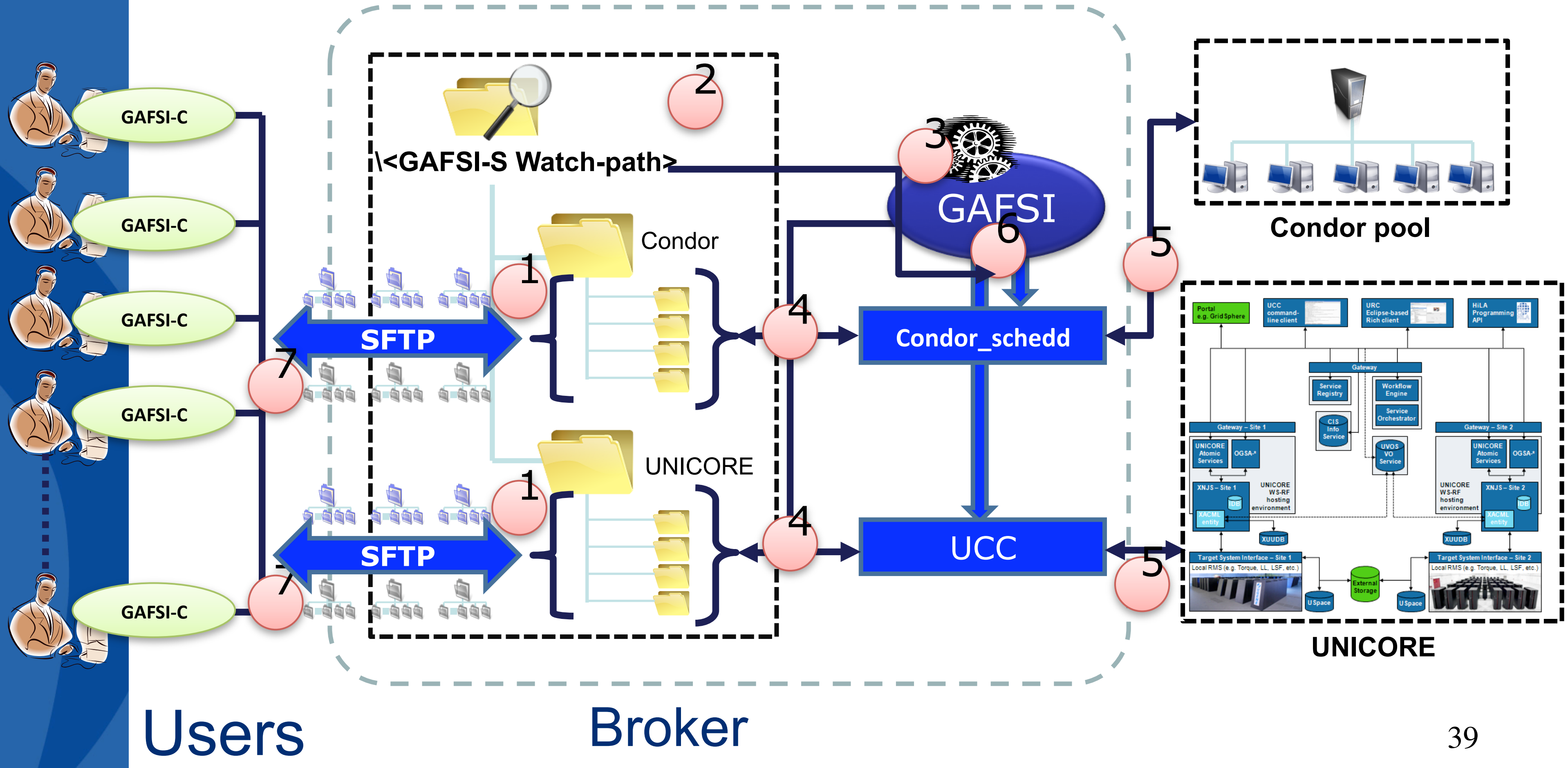
GAFSI-File sharing

File name: Job$Cluster$R$memory1024$Condor$start



GAFSI-SSH

File name: Job$Cluster$R$memory1024$Condor$start



Simple Example: R code

1. Create the input files:

```
for (j in 1:Grid.workers){
...
save(param,dataList,iterationList,file=paste(j,".RData",
sep="")) }
```

2. Copy them to the GAFSI watch path:

```
for (j in 1:Grid.workers){

    file.copy(paste(j,".RData",
    sep=""),paste(Grid.workers.addresses[j],
    "\\input.RData", sep=""))


}
```

Simple Example: R code

3.Copy the code file to the same path:

```
file.copy("worker.apl.kf.R", paste(Grid.mainpath,"\
\","code.R", sep=""))
```

4.Create the start file to trigger the submission:

```
file.create(paste(Grid.mainpath,"\\ mytask$cluster$R
$memory300$start", sep=""))
```

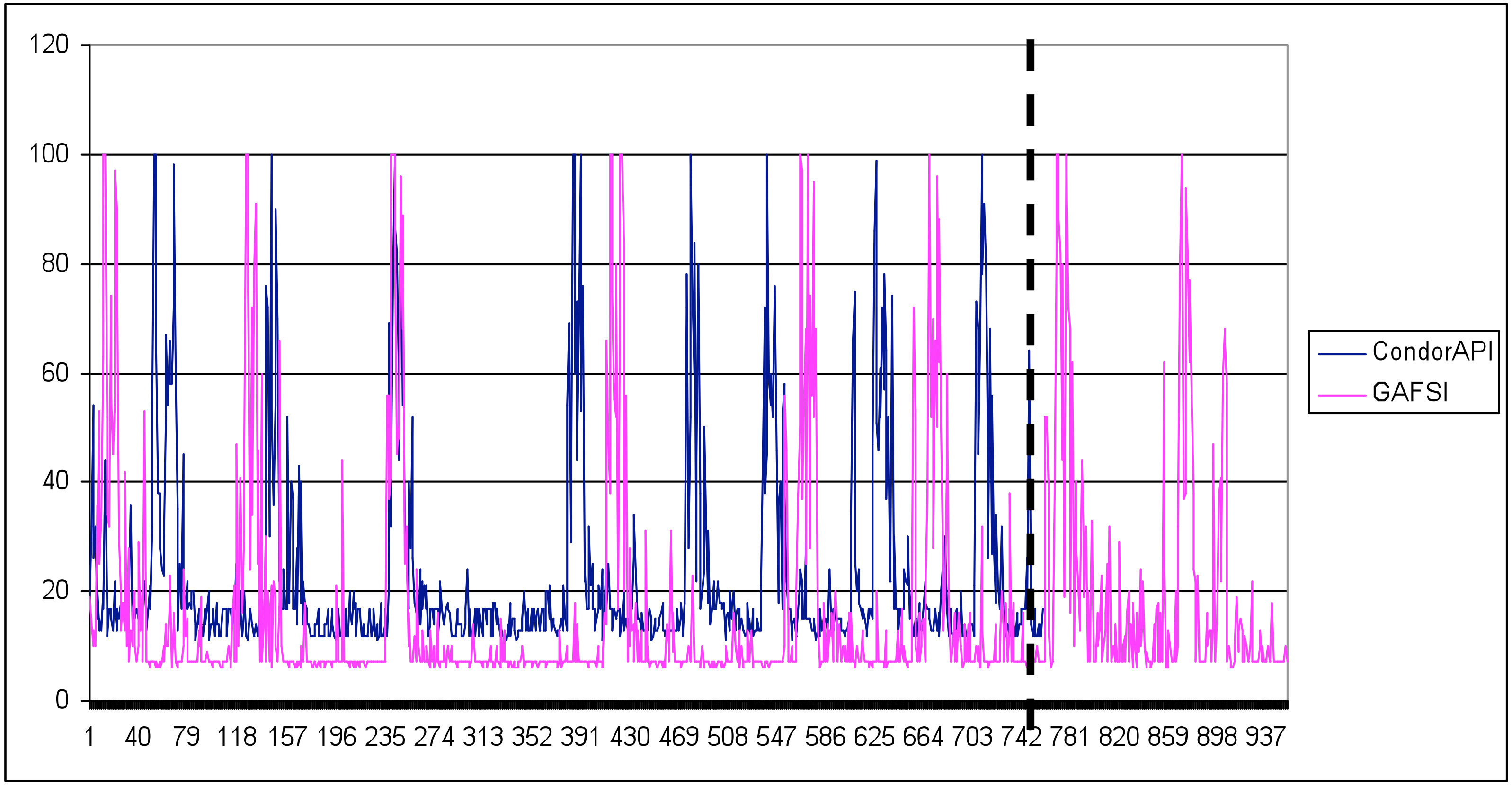
Simple Example: R code

5. Wait for the completion, then collect the result files:

```
while(TRUE){
    Sys.sleep(1)
    if(file.exists(Grid.mainpath+
    "mytask$cluster$R$exports=result.RData$memory300$start"))
    next
}
//Result collection
for(j in 1:results){
load(Grid.mainpath+"\\result"+j+".RData")
}
```

Initial Performance Evaluation

- CPU utilization of R process during the execution of a parallel version PSM.estimate() statistical modeling function on Condor



University of Stavanger

Conclusions and Future work

- Maintaining stability with scalability together with achieving system transparancy is a considerable challenge.
- We've proposed a broker overlay based model as an infrastructure to maintain stability with scalability.
- A grid access file system interface is proposed to solve the concurrency problem. It is currently being implemented on Condor and UNICORE frameworks.

- The proposed architecture is to be implemented on existing Grid frameworks.
- GAFSI is to be implemented on Linux based on FUSE.

University of Stavanger

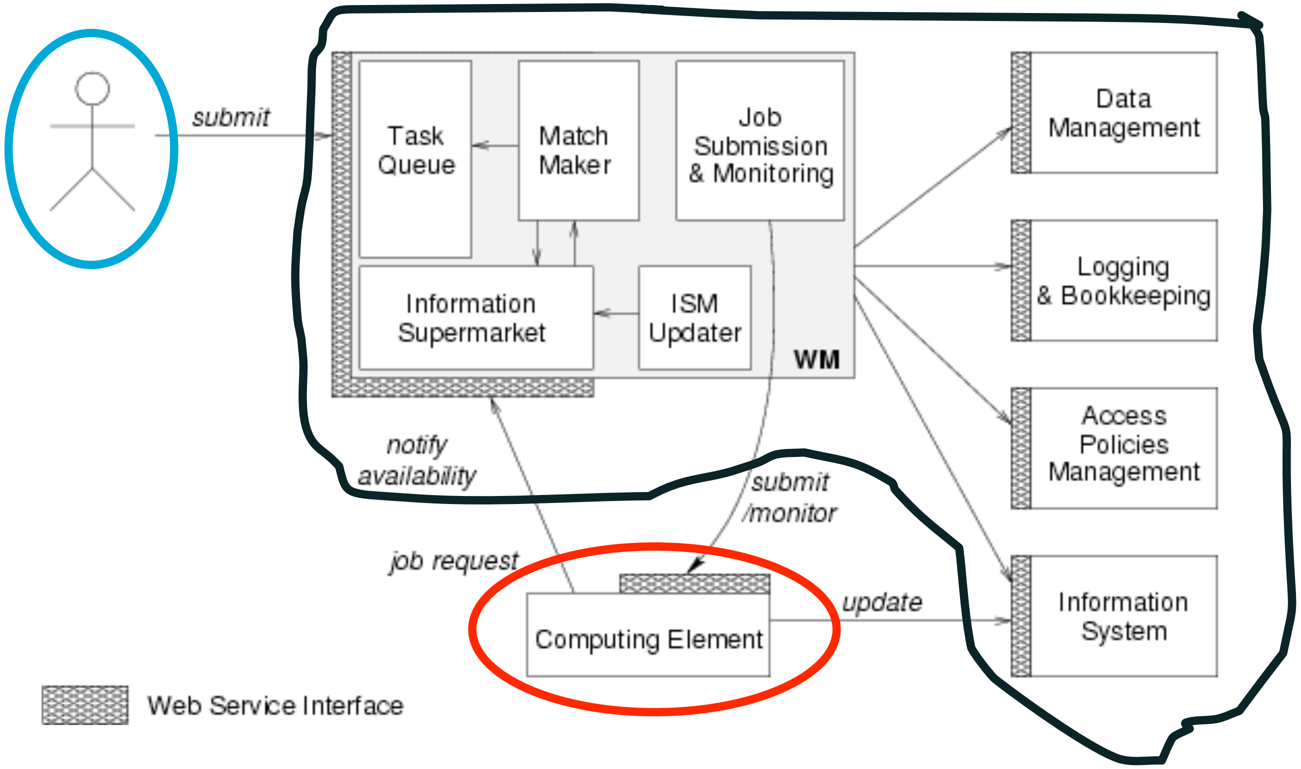
Thank You

Universitetet i Stavanger

Additional Slides

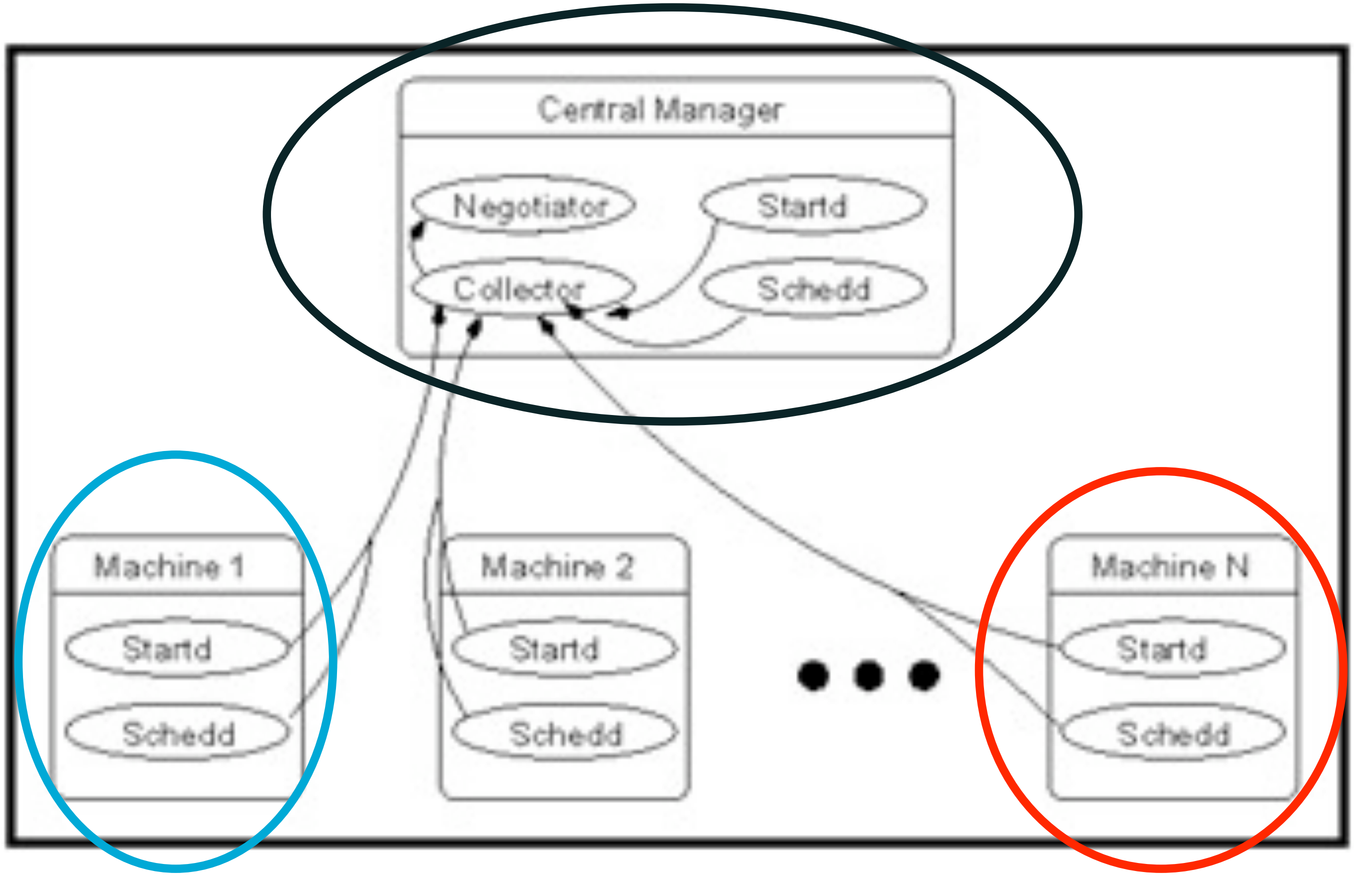
Machine organization: Flat

- **gLite** Workload Management System (**WMS**)



Machine organization: Flat

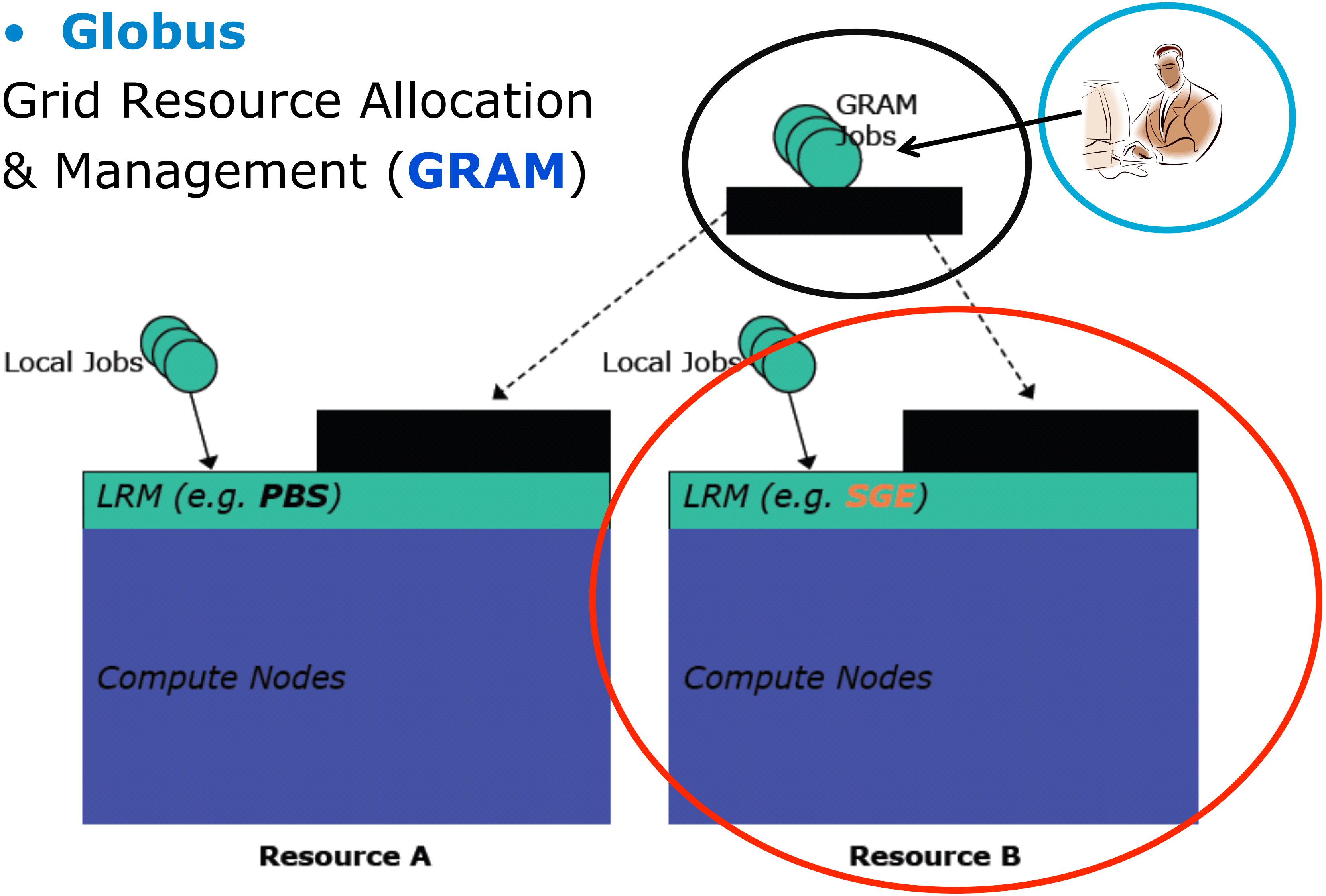
- **Condor** Central Manager (**CM**)



Machine organization: Flat

- **Globus**

Grid Resource Allocation & Management (**GRAM**)


GRAM Jobs
Local Jobs
Local Jobs
LRM (e.g. PBS)
LRM (e.g. SGE)
Compute Nodes
Compute Nodes
Resource A
Resource B