JCatascopia: Monitoring Elastically Adaptive Applications in the Cloud

Demetris Trihinas, George Pallis, Marios D. Dikaiakos

{trihinas, gpallis, mdd}@cs.ucy.ac.cy

14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2014)

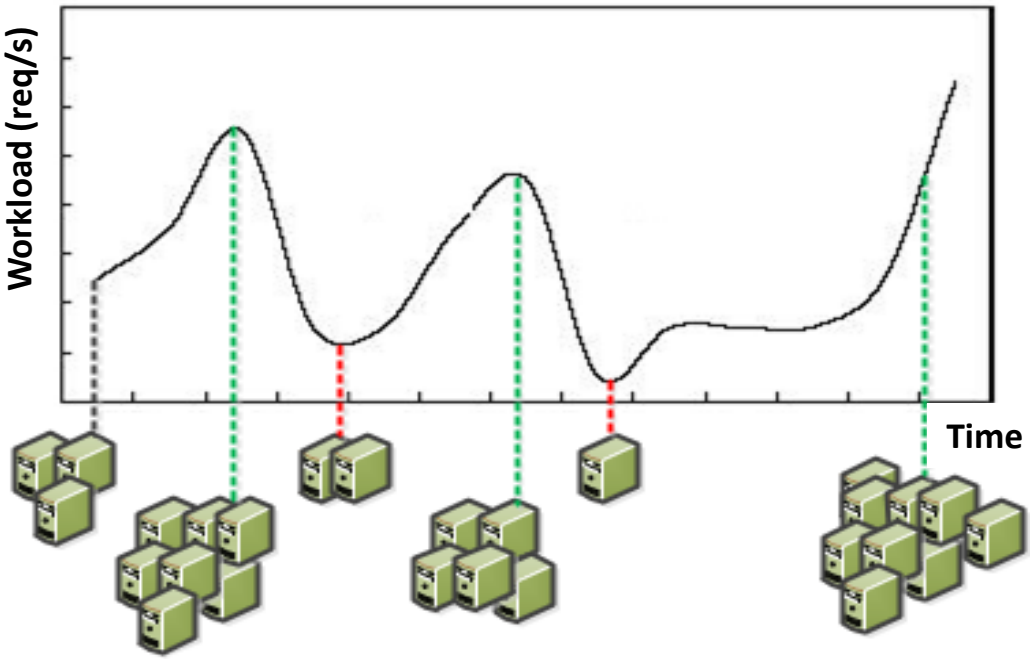
May 26-29, Chicago, IL, USA

Presentation Outline

- Elasticity in Cloud Computing
- Cloud Service Monitoring Challenges
- Existing Monitoring Tools and their Limitations
- JCatascopia Monitoring System
  - Architecture
  - Features
  - Evaluation
- Conclusions and Future Work

Elasticity in Cloud Computing

- Ability of a system to ***expand*** or ***contract*** its dedicated resources to meet the current demand



Cloud Monitoring Challenges

- Monitor heterogeneous types of information and resources
- Extract metrics from multiple levels of the Cloud
  - Low-level metrics (i.e. CPU usage, network traffic)
  - High-level metrics (i.e. application throughput, latency, availability)
- Metrics collected at different time granularities

Cloud Monitoring Challenges

- Operate on any Cloud platform
- Monitor Cloud services deployed across multiple Cloud platforms
- Detect configuration changes in a cloud service
  - Application topology changes (e.g. new VM added)
  - Allocated resource changes (e.g. new disk attached to VM)

Elasticity Support

Existing Monitoring Tools

Cloud Specific Monitoring Tools

Benefits

- Provide MaaS capabilities
- Fully documented
- Easy to use
- Well integrated with underlying platform

Limitations

- Commercial and proprietary which limits them to operating on specific Cloud IaaS providers

General Purpose Monitoring Tools

Benefits

- Open-source
- Robust and light-weight
- System level monitoring
- Suitable for monitoring Grids and Computing Clusters

Limitations

- Not suitable for dynamic (elastic) application topologies
- Limited application-level monitoring

Monitoring Tools with Elasticity Support

- [de Carvalho, INM 2011]
  - Nagios + Controller on each physical host to notify *Nagios Server* with a list of instances currently running on the system
- Lattice Monitoring Framework [Clayman, NOMS 2011]
  - Controller periodically requests from hypervisor list of current running VMs

Limitations

- Special entities required at physical level
- Depend on current hypervisor

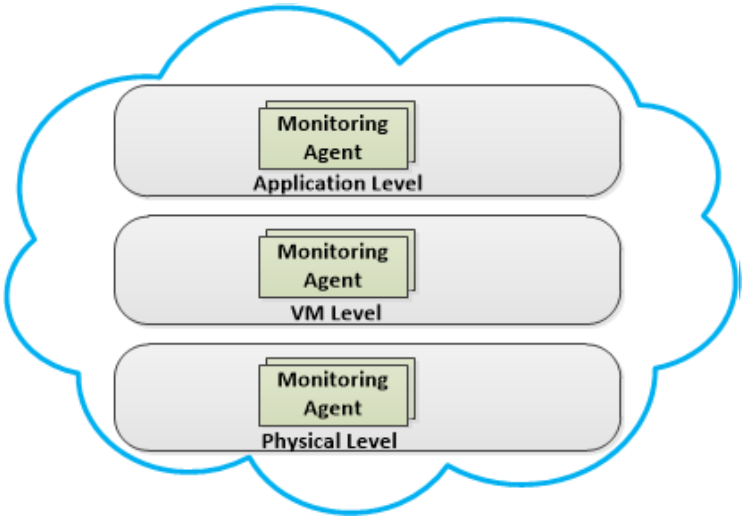
JCatascopia Monitoring System

JCatascopia Monitoring System

- ✓ Open-source
- ✓ Multi-Layer Cloud Monitoring
- ✓ Platform Independent
- ✓ Capable of Supporting Elastic Applications
- ✓ Interoperable
- ✓ Scalable

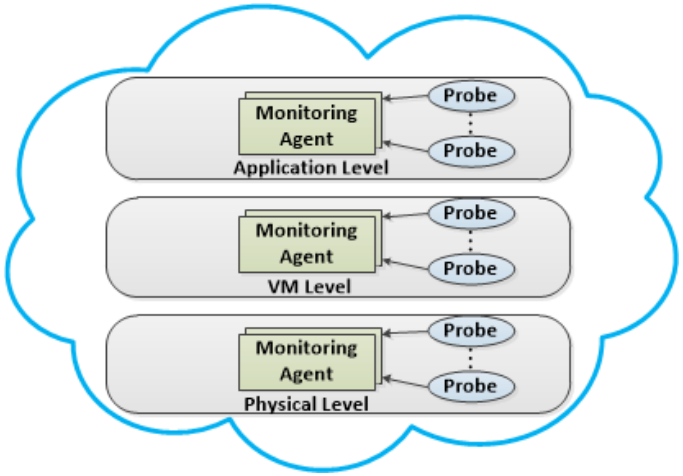
JCatascopia Architecture

Monitoring Agents



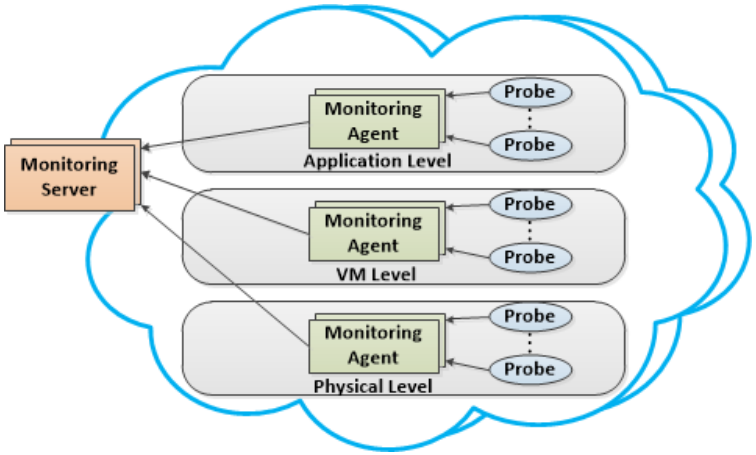
- Light-weight monitoring instances
- Deployable on physical nodes or virtual instances
- Responsible for the metric collection process
- Aggregate and distribute collected metrics (pub/sub)

Monitoring Probes



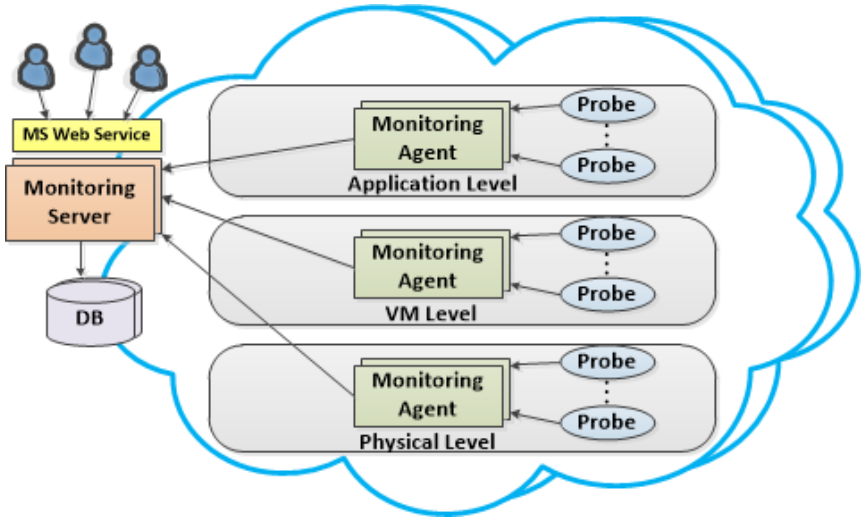
- The actual metric collectors managed by Monitoring Agents
- Collect system-level and application performance metrics
- JCatascopia Probe API
- Dynamically deployable to Monitoring Agents
- Filtering mechanism at Probe level

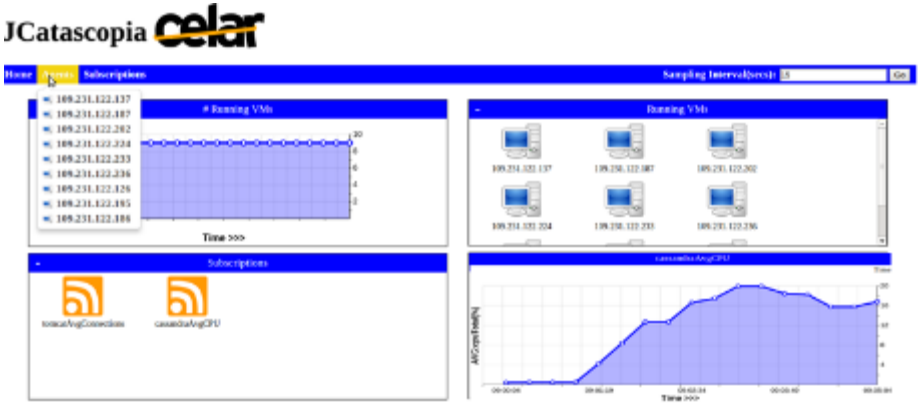
Monitoring Servers



- Receive metrics from Monitoring Agents
- process and store metrics in Monitoring Database
- Handle user metric and configuration requests
- Hierarchy of Monitoring Servers for greater scalability

JCatascopia Architecture

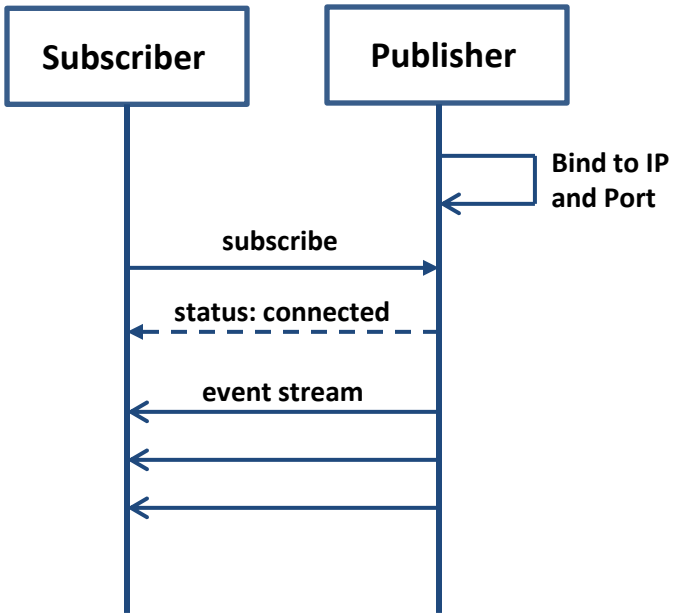




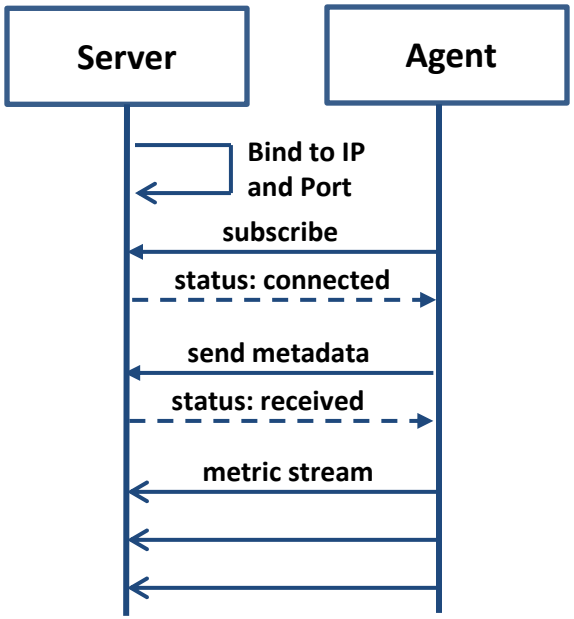
- JCatascopia REST API
- JCatascopia-Web User Interface

- JCatascopia Database Interface
  - Allows users to utilize their own Database solution with JCatascopia
  - Currently available: MySQL, Cassandra

Dynamic Agent Discovery



(a) Classic pub/sub



(b) JCatascopia

Benefits

- Monitoring Servers are agnostic of Agent network location
- Agents appear dynamically

Eliminated the need to

- Restart or reconfigure Monitoring System
- Depend on underlying hypervisor
- Require directory service with Agent locations

Dynamic Agent Removal

- Heartbeat monitoring to detect when Agents:
  - Removed due to scaling down elasticity actions
  - Temporary unavailable (network connectivity issues)

Metric Subscription Rule Language

- Aggregate single instance metrics

```
SUM(errorCount)
```

- Generate high-level metrics at runtime

```
DBthroughput =
            AVG(readps+writeps)
```

```
<SubscriptionRule> ::= <Filter>, <Members>, <Action>

<Filter> ::= <MetricName> = <Expression> | <GroupFunction>(<Expression>)
<Expression> ::= <Operand> | <Operand> <Op> <Expression>
<Operand> ::= <Number> | <MetricName> | (<Expression>)
<Op> ::= +|-|*|/
<MetricName> ::= <String>
<GroupFunction> ::= AVG|SUM|MIN|MAX

<Members> ::= MEMBERS = ({<AgentID>,} <AgentID>)
<AgentID> ::= <String>

<Action> ::= ACTION = NOTIFY(<Act>) | PERIOD(<Number>)
<Act> ::= ALL | {<Relation> <Number>,} <Relation> <Number>
<Relation> ::= <|>|=|!=|>=|<=
```

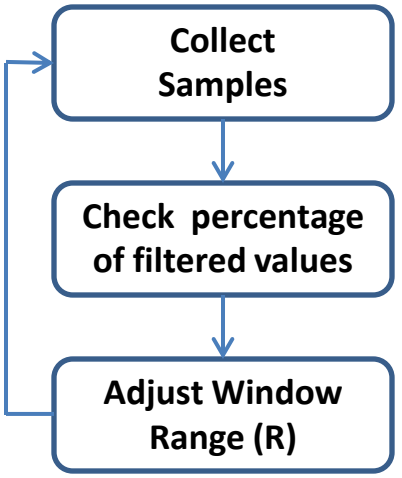
Subscription Rule Example

Average `DBthroughput` from the low-level metrics `readps` and `writeps` of a database cluster comprised of *N* nodes:

```
DBthroughput = AVG(readps + writeps)
      MEMBERS = [id1, ... ,idN]
      ACTION = NOTIFY(<25,>75%)
```

Adaptive Filtering

- Simple fixed uniform range filter windows are not effective:
  - i.e. filter `currentValue` if in window `previousValue`±`R`
  - No guarantee that any values will be filtered at all
- *Adaptive filter window range*
  - window range (R) is not static but depends on percentage of values previously filtered



JCatascopia Evaluation

Evaluation

- Validate JCatascopia functionality and performance
- Compare JCatascopia to other Monitoring Tools
  - Ganglia
  - Lattice Monitoring Framework
- Testbed
  - Different domains of Cloud applications
  - Various VM flavors
  - 3 public Cloud providers and 1 private Cloud

Testbed

Cloud Provider	VM no.	VM Flavor	Applications
GRNET Okeanos public Cloud	15	1GB RAM, 10GB Disk, Ubuntu Server 12.04 LTS	12 VMs Cassandra 3 VMs YCSB Clients
Flexiant FlexiScale platform	10	2 VCPU, 2GB RAM, 10GB Disk, Debian 6.07 (Squeeze)	HASCOP an attributed, multi-graph clustering algorithm
Amazon EC2	10	m1.small with CentOS 6.4 (1VCPU, 1.7GB RAM, 160GB Disk)	
OpenStack Private Cloud	60	2 VCPU, 2GB RAM, 10GB Disk, Ubuntu Server 12.04 LTS	

We have deployed on all VMs JCastascopia Monitoring Agents, Ganglia gmonds and Lattice DataSources

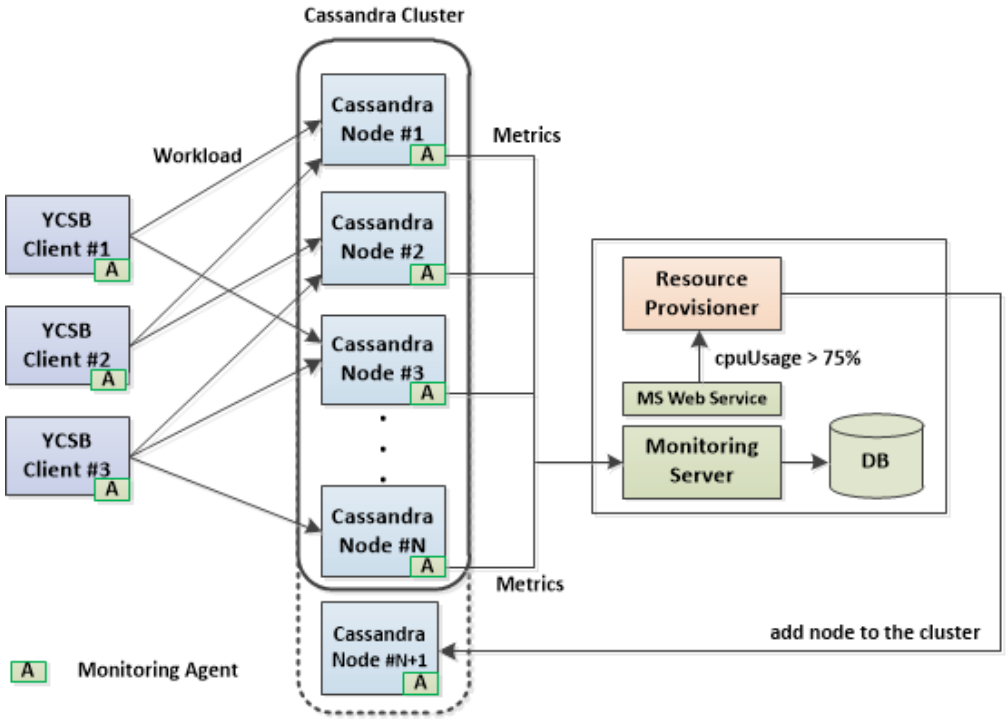
Testbed - Available Probes

Probe	Metrics	Period (sec)
CPU	cpuUserUsage, cpuNiceUsage, cpuSystemUsage, cpuIdle, cpuIOWait	10
Memory	memTotal, memUsed, memFree, memCache, memSwapTotal, memSwapFree	15
Network	netPacketsIN, netPacketsOUT, netBytesIN, netBytesOUT	20
Disk Usage	diskTotal, diskFree, diskUsed	60
Disk IO	readkbps, writekbps, iotime	40
Cassandra	readLatency, writeLatency	20
YCSB	clientThroughput, clientLatency	10
HASCOP	clustersPerIter, iterElapTime, centroidUpdTime, pTableUpdTime, graphUpdTime	20

Experiment 1. Elastically Adapting Cassandra Cluster

- Scale out Cassandra cluster to cope with increasing workload
- Experiment uses 15 VMs in Okeanos cluster
- Subscription Rule to notify Provisioner to add VM when scaling condition violated:

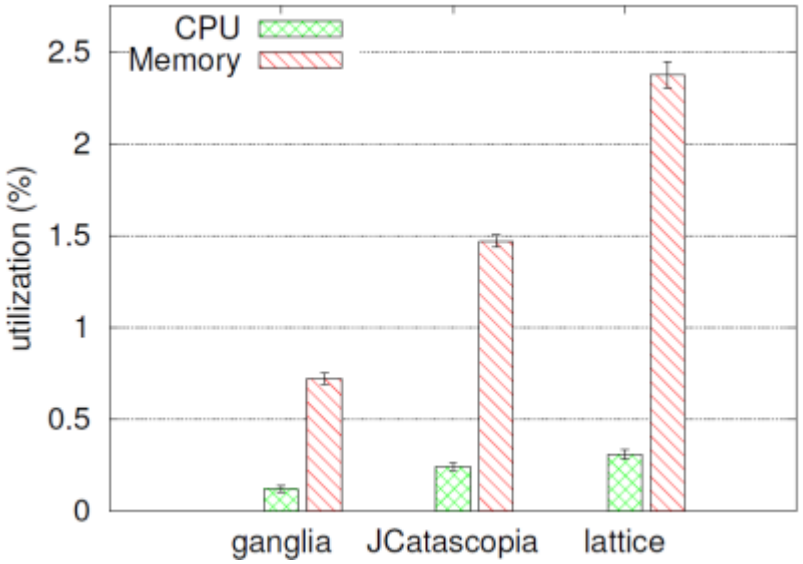
```
cpuTotalUsage = AVG(1 - cpuIdle)
    MEMBERS = [id1, ... ,idN]
      ACTION = NOTIFY(>=75%)
```



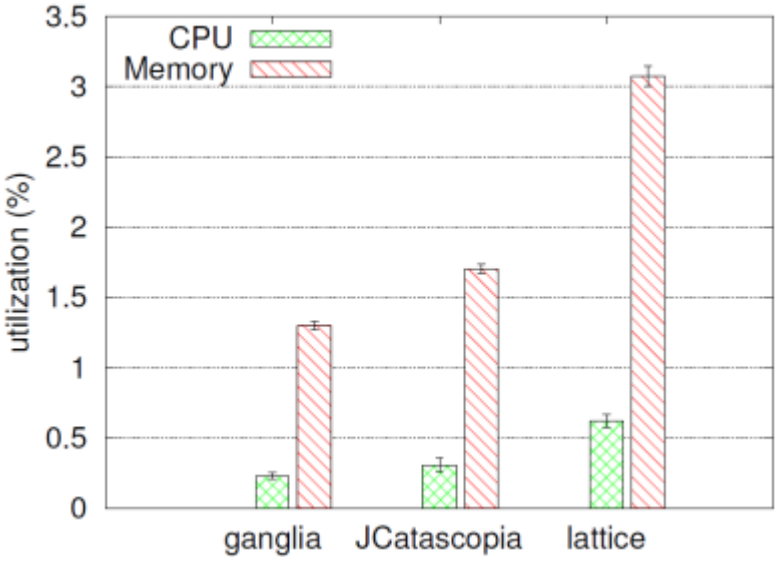
VMs	Probes
YCSB Clients	YCSB
Cassandra	CPU, Memory, Network, DiskIO, Cassandra

Experiment 1. Elastically Adapting Cassandra Cluster

Monitoring Agent Runtime Impact



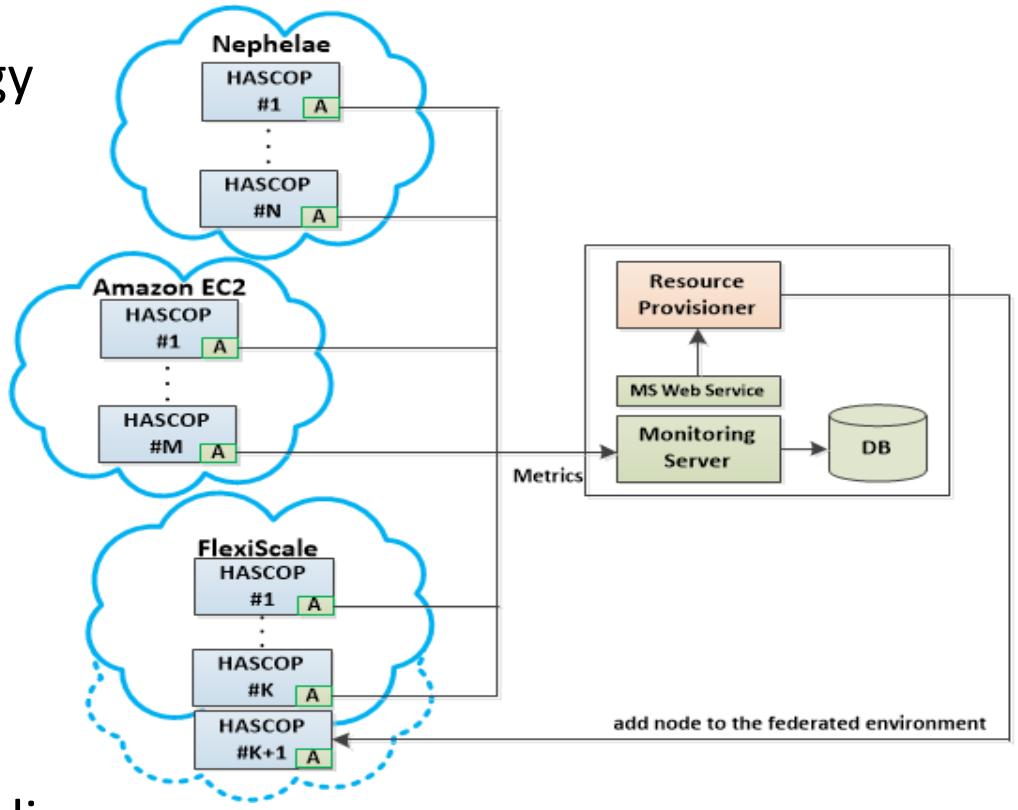
YCSB Agent Utilization



Cassandra Agent Utilization

Experiment 2. Monitoring a Cloud Federation Environment

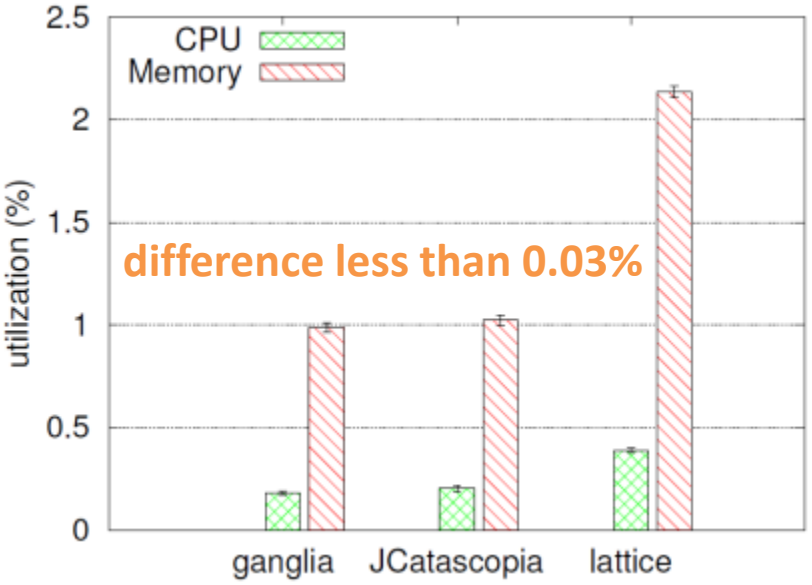
- Monitor an application topology spread across multiple Clouds:
  - OpenStack (10 VMs)
  - Amazon EC2 (10 VMs)
  - Flexiant (10 VMs)
- Compare JCatascopia, Ganglia and Lattice runtime footprint
- Compare JCatascopia and Ganglia network utilization



VMs	Probes
HASCOP	CPU, Memory, DiskUsage, HASCOP

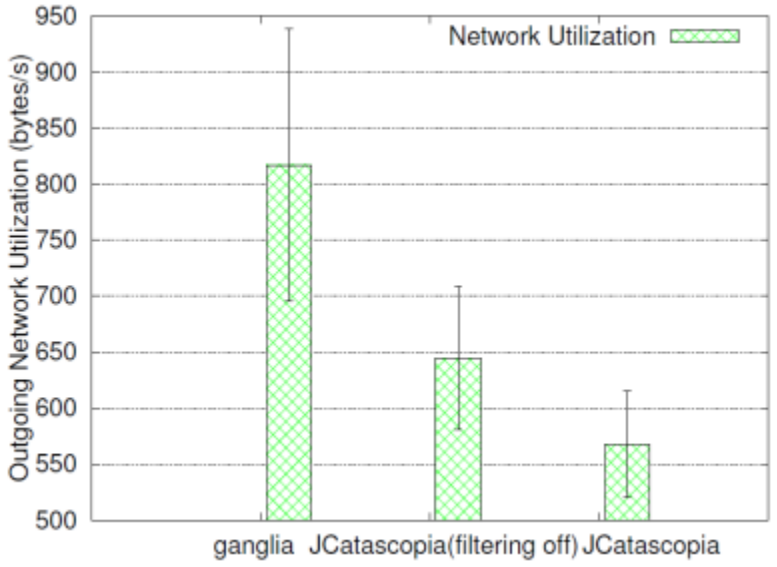
Experiment 2. Monitoring a Cloud Federation Environment

Monitoring Agent Runtime Impact



HASCOP Agent Utilization

Monitoring Agent Network Utilization



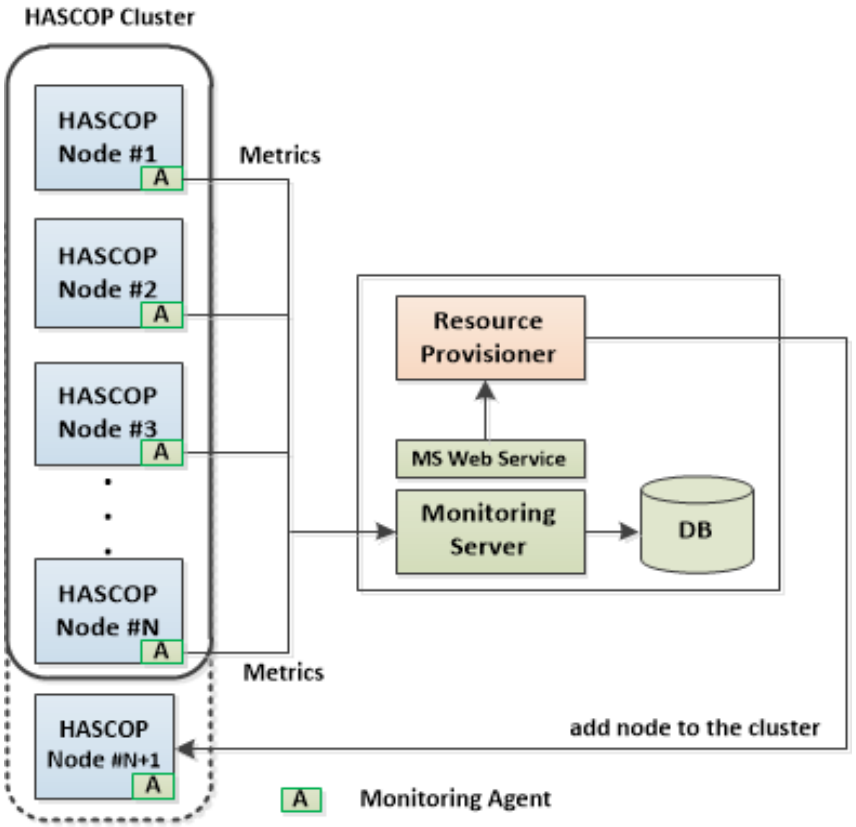
Agent Network Utilization

When in need of application-level monitoring, for a small runtime overhead, JCatascopia can reduce monitoring network traffic and consequently monitoring cost

Experiment 3. JCatascopia Scalability Evaluation

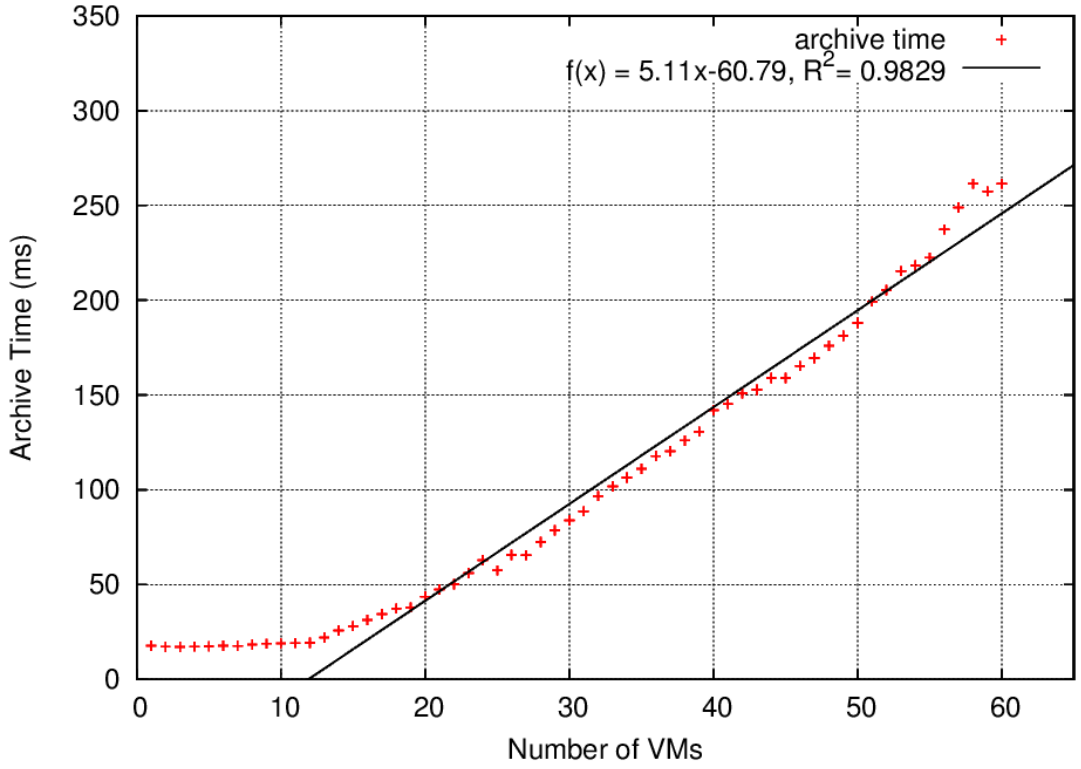
- Experiment uses the 60 VMs on OpenStack private Cloud to scale a HASCOP cluster
- 1 Monitoring Server for 60 Agents
- Subscription Rule:

```
hascopIterElapsedTime = AVG(iterElapTime)
        MEMBERS = [id1, ... ,idN]
          ACTION = NOTIFY(ALL)
```



VMs	Probes
HASCOP	CPU, Memory, DiskUsage, HASCOP

Scalability Evaluation

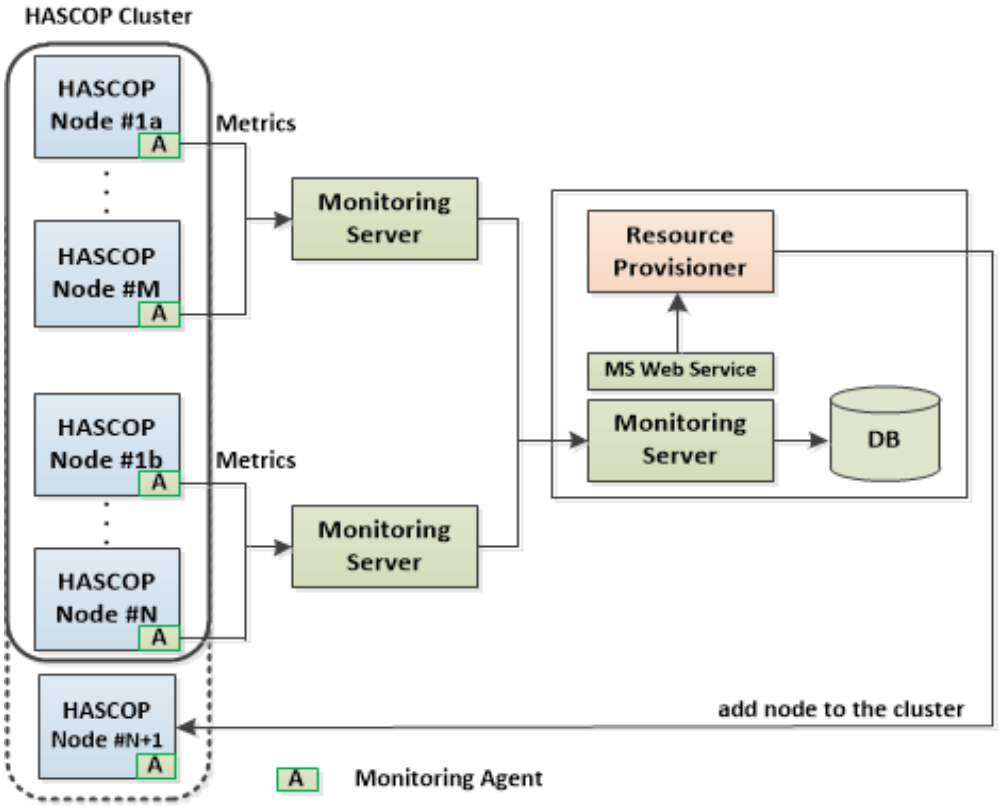


Archiving time grows linearly

Experiment 3. JCatascopia Scalability Evaluation

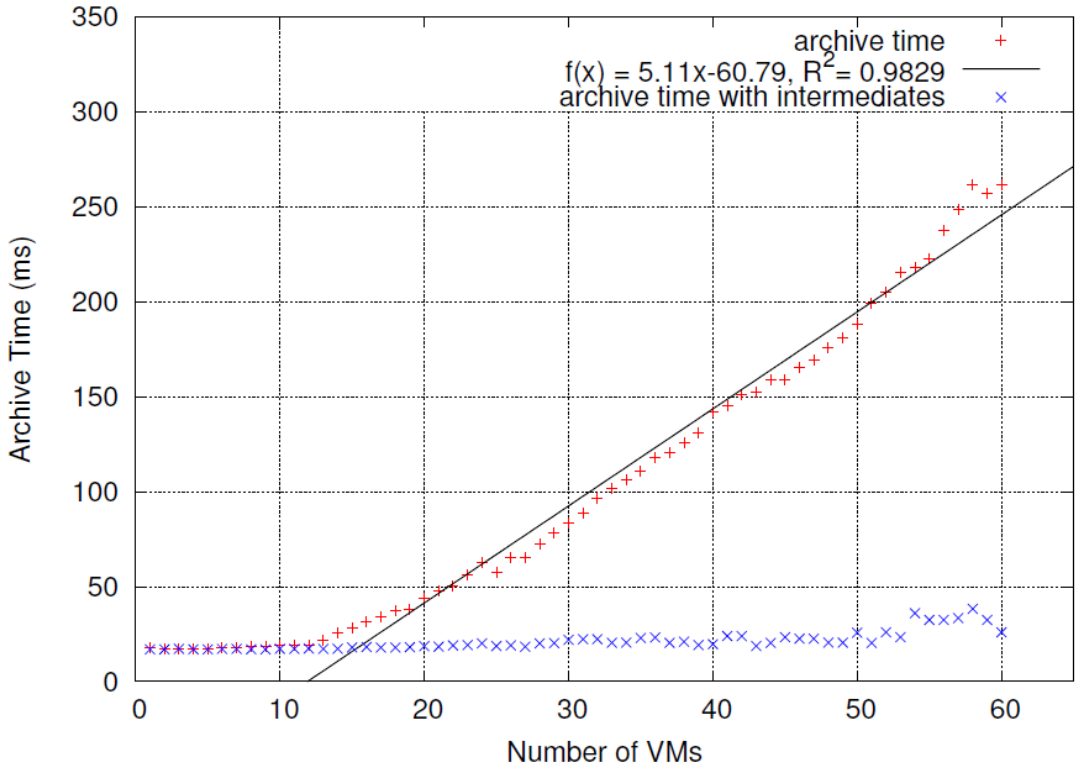
New Setup

- 2 Intermediate Monitoring Servers which aggregate metrics from underlying Agents
- 1 root Monitoring Server



VMs	Probes
HASCOP	CPU, Memory, DiskUsage, HASCOP

Scalability Evaluation



When archiving time is high, we can redirect monitoring metric traffic through Intermediate Monitoring Servers, allowing the monitoring system to scale

Conclusions

- Experiments on public and private Cloud platforms show that JCatascopia is:
  - capable of **supporting automated elasticity controllers**
  - able to adapt in a **fully automatic** manner when **elasticity** actions are enforced
  - **open-source**, **interoperable**, **scalable** and has a **low runtime footprint**

Future Work

- Further pursue **adaptive filtering**
- Enhance Probes with **adaptive sampling**
  - Adjust sampling rate when stable phases are detected
- Integrate JCatascopia with cloud cost-evaluation system

Acknowledgements

www.celarcloud.eu

co-funded by the
European Commission

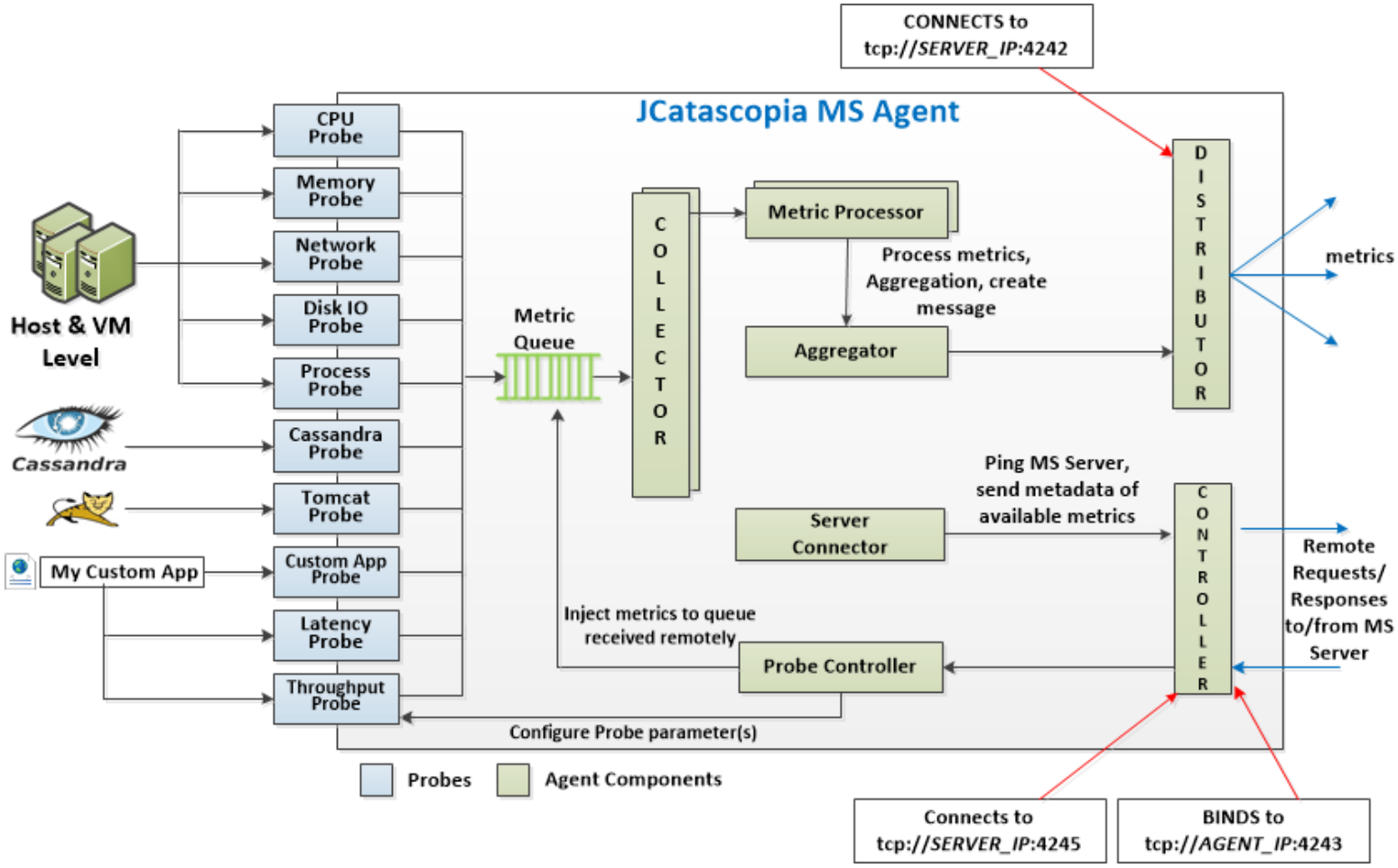
JCatascopia

https://github.com/CELAR/cloud-ms

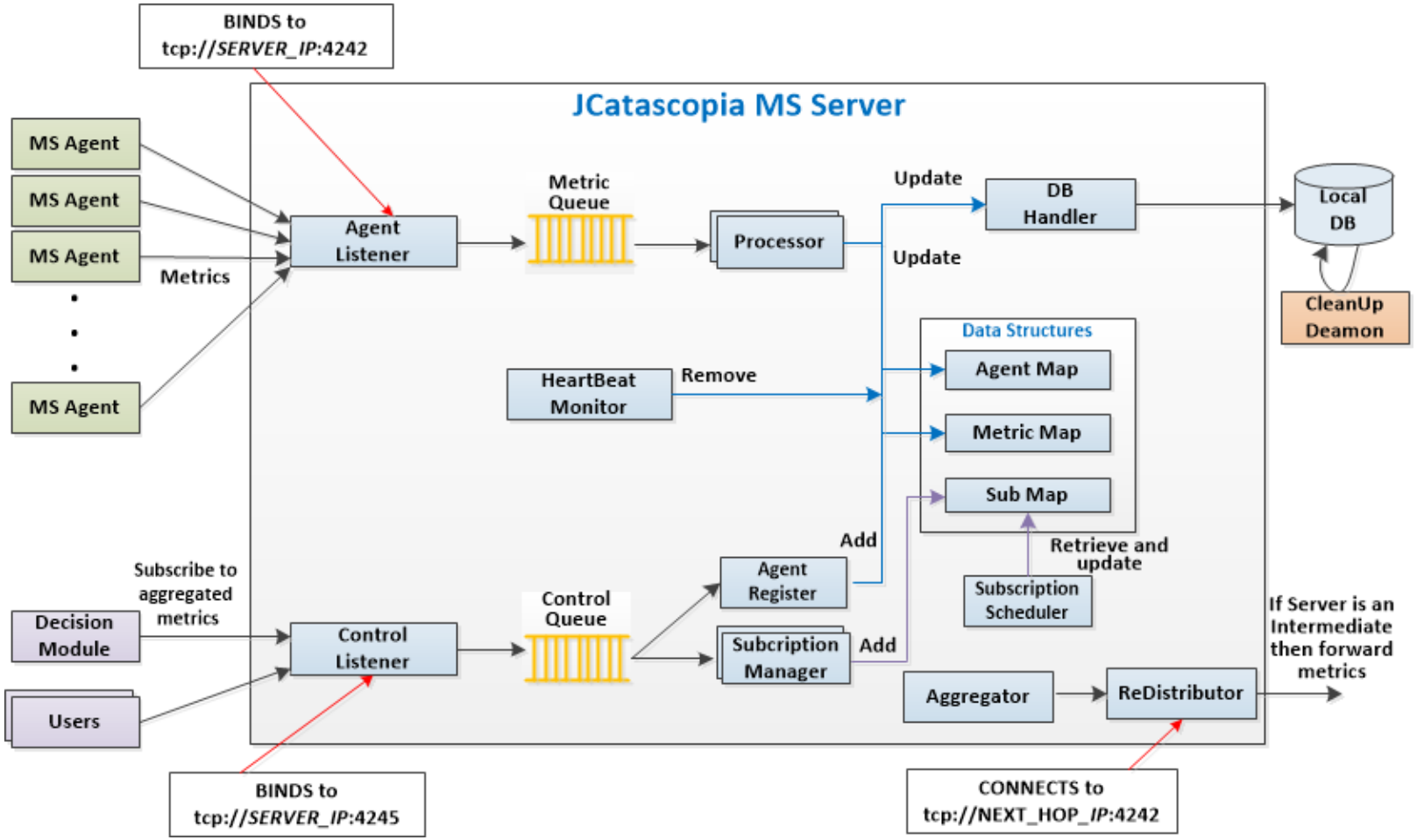
Laboratory for Internet Computing
Department of Computer Science
University of Cyprus
http://linc.ucy.ac.cy

BACKUP SLIDES

Monitoring Agents



Monitoring Servers



Dynamic Agent Removal

- Heartbeat monitoring to detect when Agents:
  - Removed due to scaling down elasticity actions
  - Temporary unavailable (network connectivity issues)