Governing Adaptive and Unplanned Systems of Systems

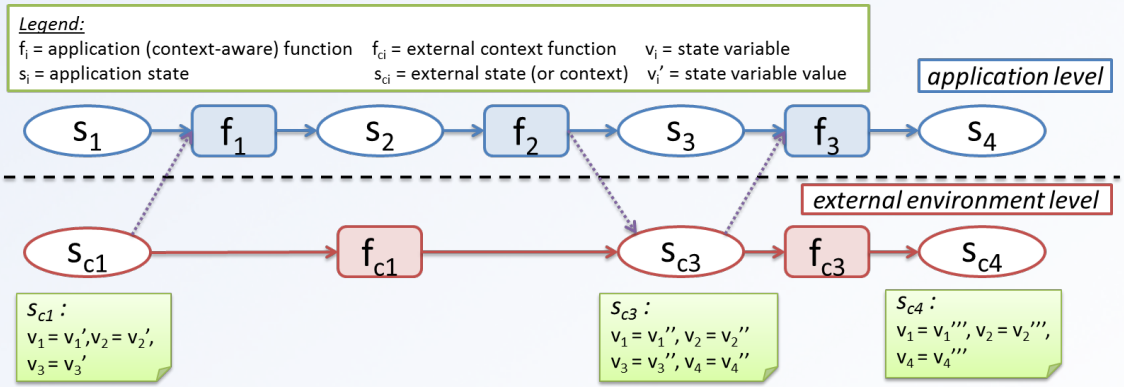
GAUSS project

Kick-off meeting
February 22-23, 2017
Università Bicocca (Milano)

Gerardo Canfora, **Eugenio Zimeo**
Università del Sannio

Context definition

- **Application state**
  - set of variables and corresponding values the application is able to access or modify
  - **Internal state**
    - set of variables only visible to the application
      - input, output
      - referred by pre/post conditions
  - **External state**
    - set of variables accessible by:
      - users
      - devices
      - other applications
    - **the context**
      - every attribute that characterizes a user and/or the (smart) environment a distributed application interacts with



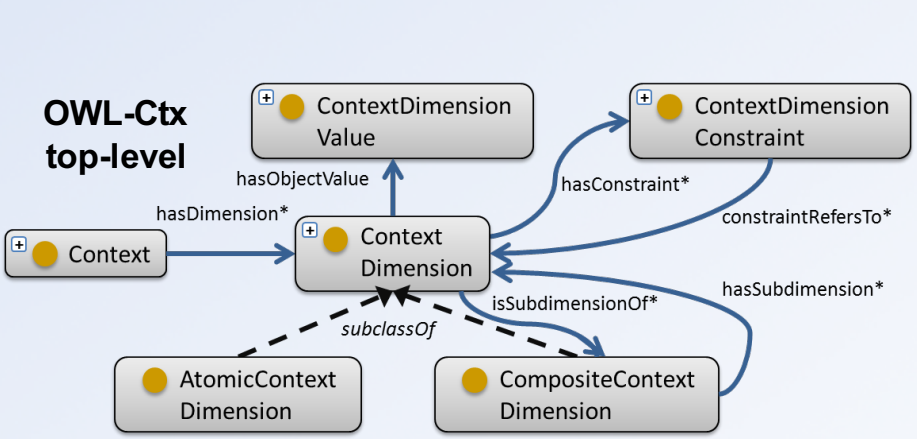
- **Context-aware applications**
  - Apps exhibiting **dependencies from the context**
  - They may change context variables

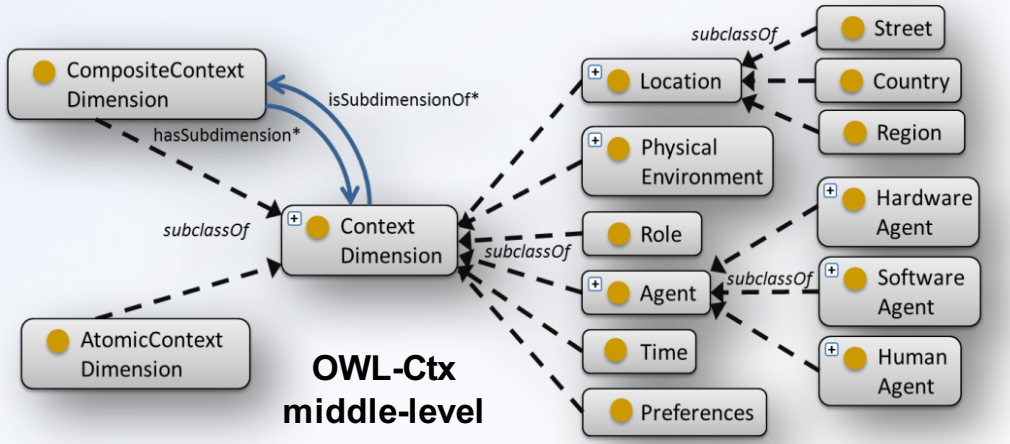
Context dimensions

- Main eSoS context dimensions
  - Users profiles
  - Users or applications usage profiles
  - Systems execution environments
    - Client devices
    - Server class hardware (e.g. Cloud infrastructures)
  - Space/time
  - Physical environment

OWL-Ctx: ontology for context

- **Extensible OWL ontology** for **modelling contexts** and **context-dependent conditions**
- Top-level ontology
  - context as a set of context dimensions and values
- Partial context **middle-level ontology**:
  - **domain-independent context dimensions** (time, location, etc.)
- **Designers** may **specify** domain-specific context dimensions and their relationships



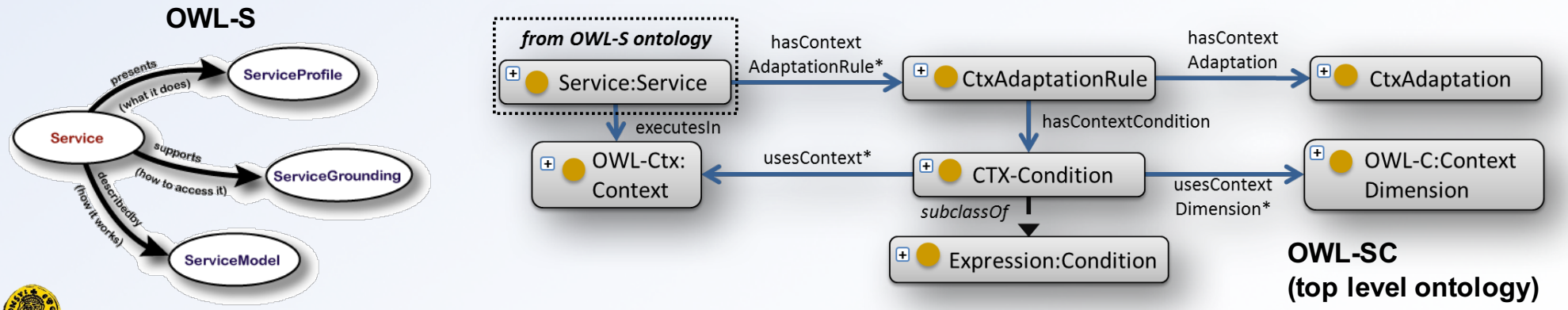



```
01 <owl:Class rdf:about="&Media;MediaContext">
02   <rdfs:subClassOf rdf:resource="&OWL-Ctx;Context"/> ...
03   <owl:equivalentClass>
04     <owl:Class><owl:intersectionOf rdf:parseType="Collection">
05       <rdf:Description rdf:about="&OWL-Ctx;Context"/>
06       <owl:Restriction><owl:onProperty rdf:resource="&OWL-Ctx;hasDimension"/>
07       <owl:someValuesFrom><owl:Class><owl:unionOf rdf:parseType="Collection">
08         <rdf:Description rdf:about="&OWL-Ctx;Agent"/>
09         <rdf:Description rdf:about="&Media;MediaContent"/>
10         <rdf:Description rdf:about="&OWL-Ctx;PhysicalEnvironment"/> ...
```

Media-Ctx specification

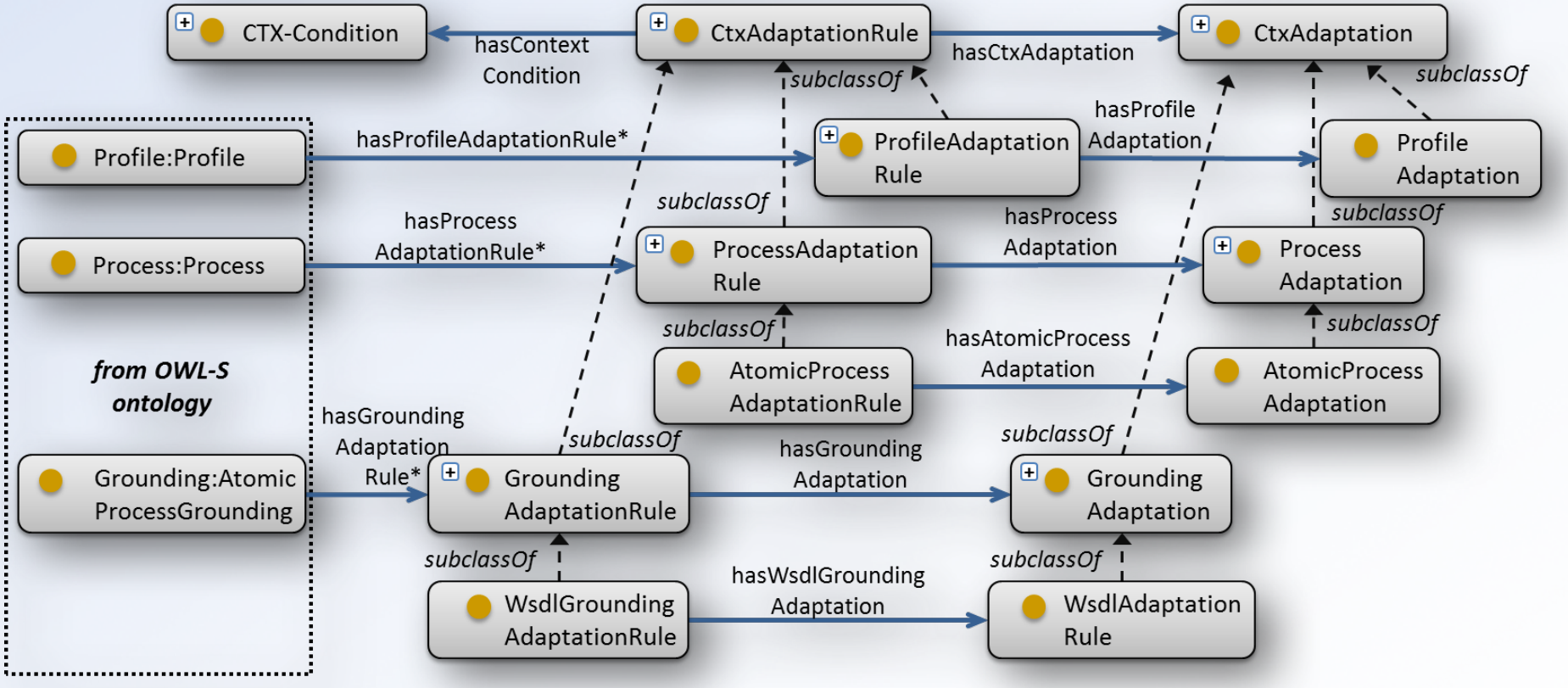
Experiences from WS: OWL-SC (1)

- *Designing context-aware services means to extend their descriptions with new attributes and rules that are able to slightly change the structure and behavior of the services according to the needs emerging in the specific context where they will be used*
- Designers may **describe slightly different versions** of the **same service**
  - **by defining context adaptation rules** over a basic OWL-S description
    - **profile, process or grounding properties** are adapted when conditions are satisfied
      - OWL-S **Profile**, **Process** and **Grounding** middle-level ontology
  - *Contexts* or *Context dimension* values may be used in ***CTX-Conditions***
    - a **current context reference** is used to refer the current situation
      - may be automatically updated by a monitoring component



Experiences from WS: OWL-SC (2)

- OWL-S **Profile**, **Process** and **Grounding** may have **context-dependent adaptations**
  - *CtxAdaptationRule*s specifications and *hasContextAdaptationRule* object property.



OWL-SC (3): Context Conditions and Adaptations

- **Context-dependent conditions:**
  - Expressed using SWRL

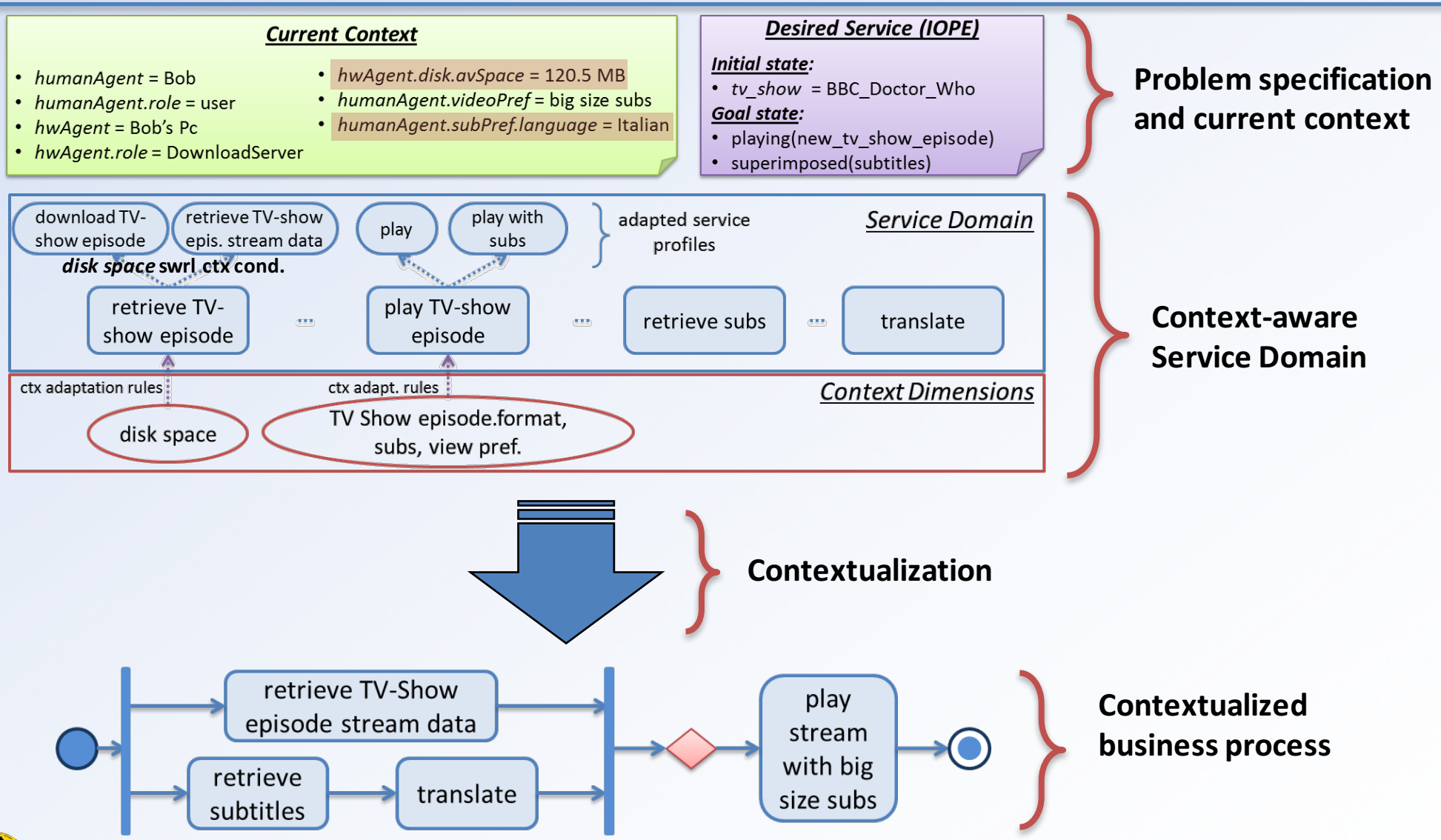
```
OWL-Ctx:hasDimension(current_ctx, ?hwAgent) ∧ Media:
HWAgent(?hwAgent) ∧ OWL-Ctx:hasSubdimension(?hwAgent,
?role) ∧ OWL-Ctx:Role(?role) ∧ sameAs(?role,
Media:DownloadServer) ∧ OWL-Ctx:hasSubdimension(?hwAgent,
?disk) ∧ Media:Disk(?disk) ∧ Media:diskMBSpace(?disk,
?avSpace) ∧ swrlb:lowerThanOrEqual(?avSpace, 500)
```

SWRL context condition (*diskSpace*)

- **Context adaptations**:
  - *Grounding adaptation: retrieve TV-show -> "retrieve TV-show streaming data"*

  - **Defaulting** an input/output **parameter**
  - **Nulling a parameter**, not applicable for a specific context condition
  - **Changing the owls** *<process:parameterType>* of an input/output parameter to a different ontology concept
  - **Replacing pre-conditions or effects** of the basic OWL-S service description
  - **Changing the WsdlAtomicProcessGrounding** input/output section of an atomic Process **with a new Wsdl MessageMap**
  - **Changing the WsdlAtomicProcessGrounding section of an atomic Process with a new WSDL operation and/or WSDL portType/interface**

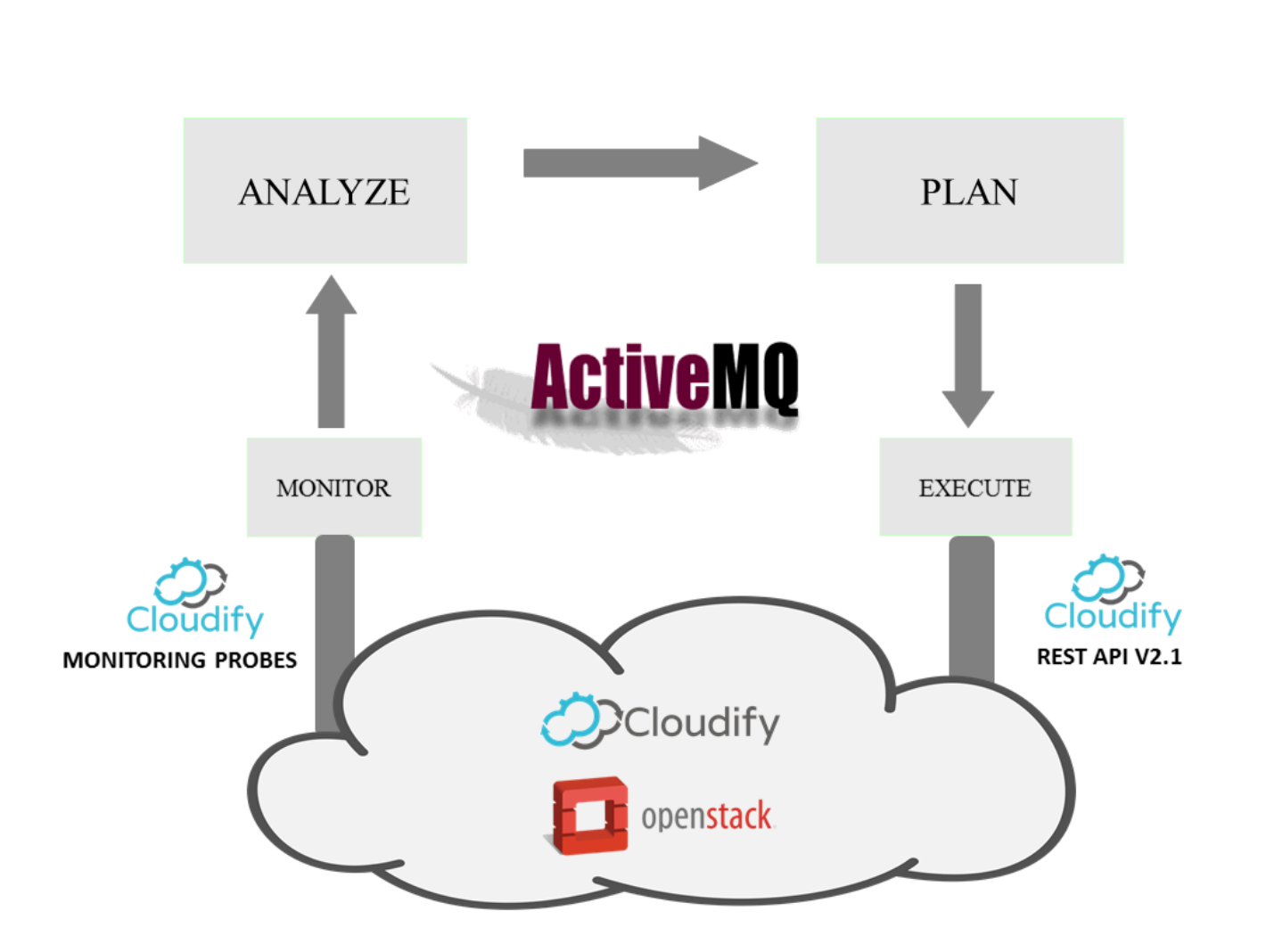
Example: Context-aware Composition



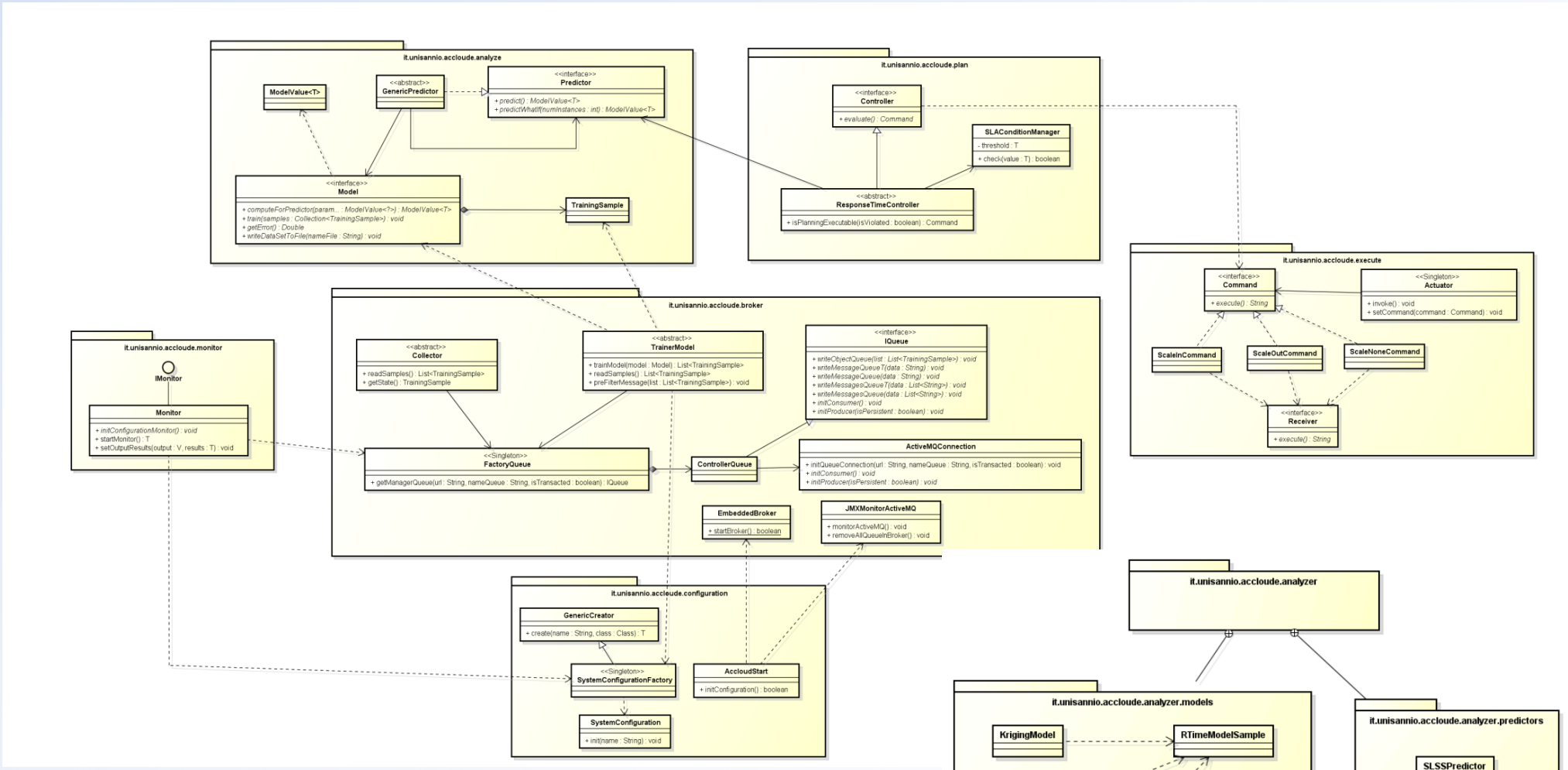
Cloud contexts

- System behavior
  - rt = SYSTEM_MODEL(resources, pending_queue, reqtype_rate)
- Users behavior
  - reqtype_rate = USAGE_MODEL(APP_BEHAVIOR, reqtype_proc, USERS(time, events))
- **Contexts**
  - SYSTEM_MODEL -> BlackBox/WhiteBox/Hybrid models
  - USERS (time) ->BlackBox (e.g. SVR) -- Cyclic/Seasonal behaviors
  - APP_BEHAVIOR -> BlackBox (e.g. HMM) – admitted/possible transitions

ACCloudE Framework: big picture

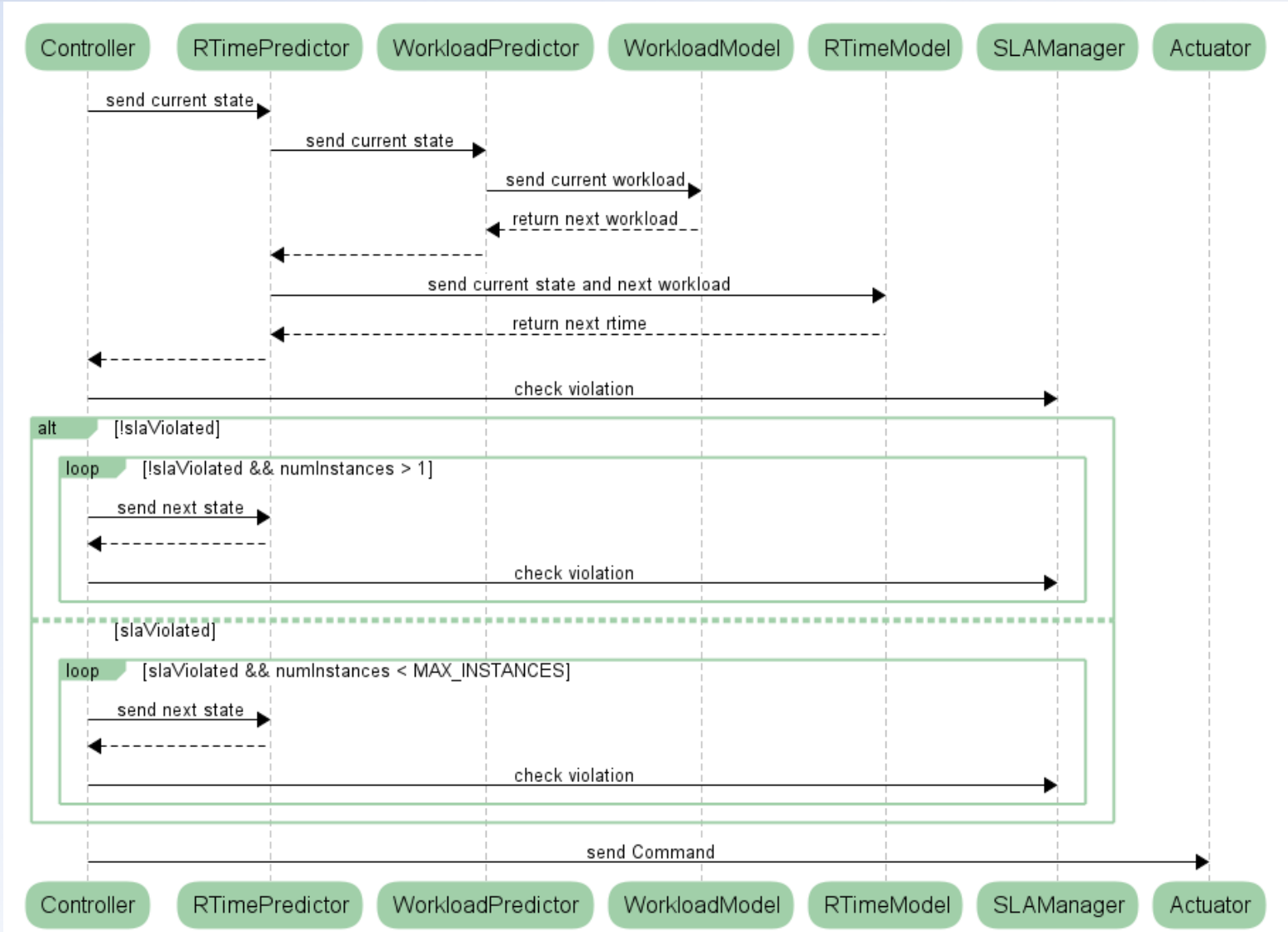


ACCloudE framework

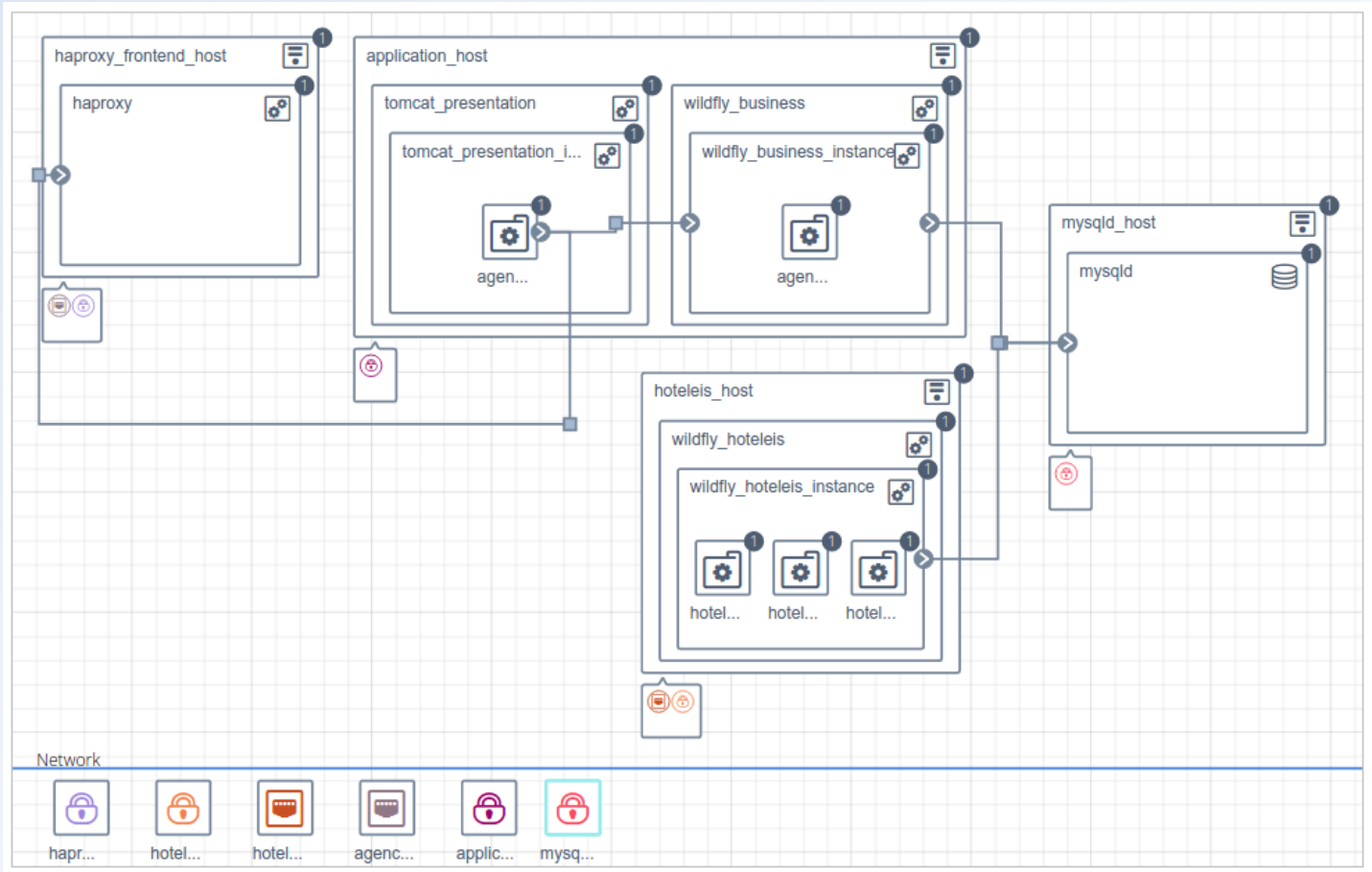


Implemented models

ACCloudE: Controller



Test application: Cloudify topology



System Behavior models

Kriging

- Geo-statistics method of interpolation

$$Z^*(x_0) = \sum_{i=1}^{n} \lambda_i Z(x_i)$$

$$\gamma^*(h) = \frac{1}{2n} \sum_{i=1}^{n} [Z(x_i + h) - Z(x_i)]^2$$

- Ponderators λ
  - variation *vs* distance

Libreria JSAT

Support Vector Regression

- SVM applied to regression

$$\{\mathbf{x}_n, y_n\}\, n = 1, ..., N \quad \mathbf{x}_n \in \mathbb{R}^D \quad y_n \in \mathbb{R}$$

$$f(\mathbf{x}) = \mathbf{x} \cdot \mathbf{w} + b \quad w \in \mathbb{R}^D \quad b \in \mathbb{R}$$

Distance: $\frac{1}{\|\mathbf{w}\|}$

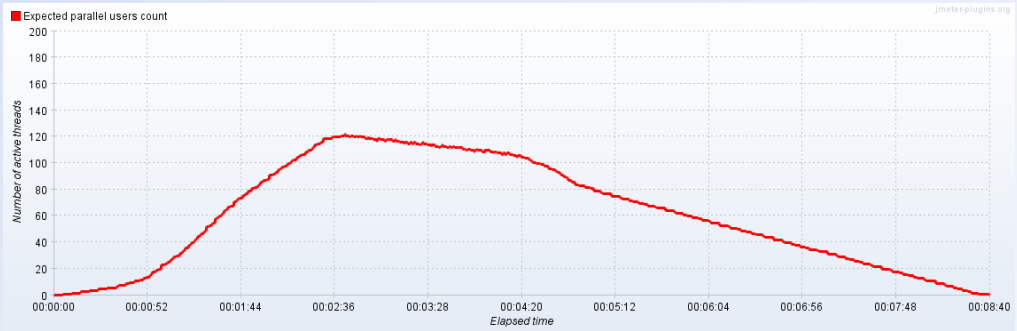
$$|y_n - \mathbf{x}_n \cdot \mathbf{w} - b| \leq \varepsilon$$

- Parameters
  - Kernel function
  - C
  - γ

Libreria WEKA

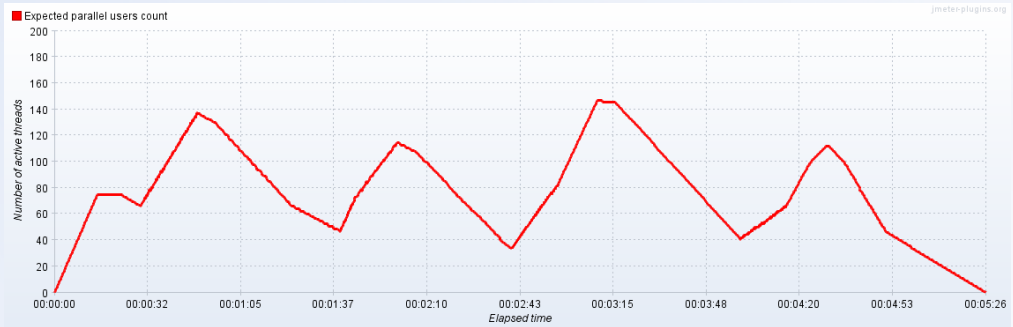
Experimentation (1)

- Workload 1

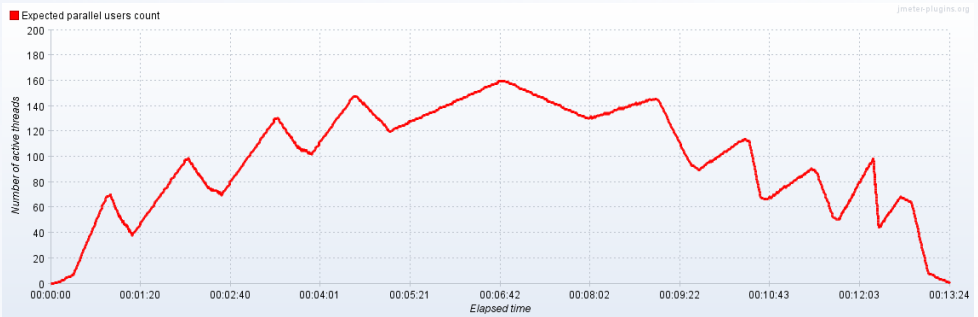


- Workload 2

- Workload 3



- Workload 4



Experimentation (2)

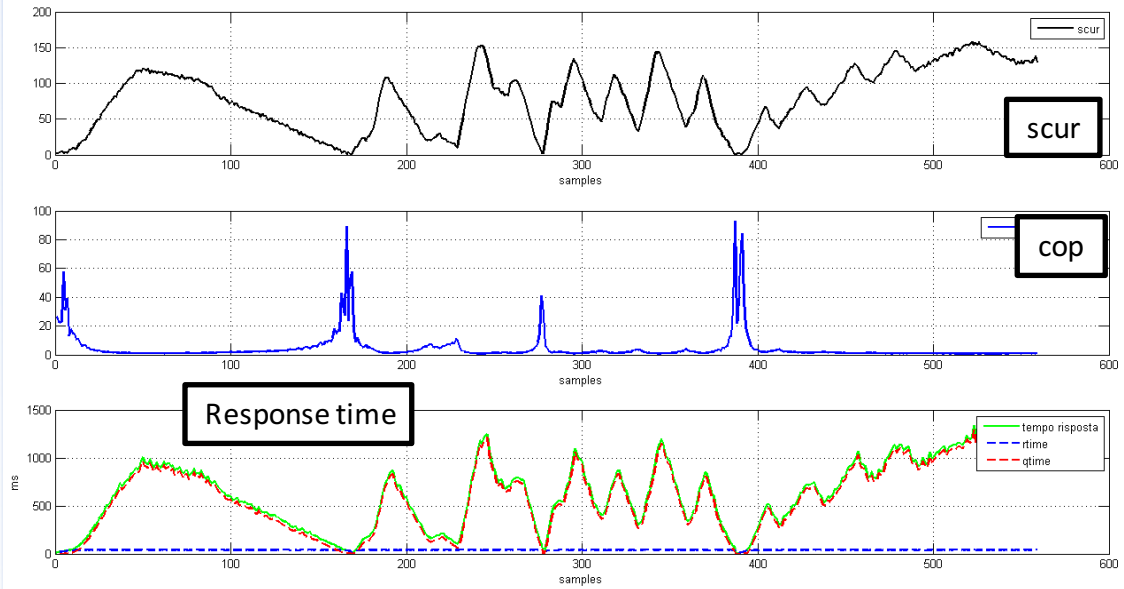
- Data analysis and pre-processing
  - data cleaning: $scur = 0$

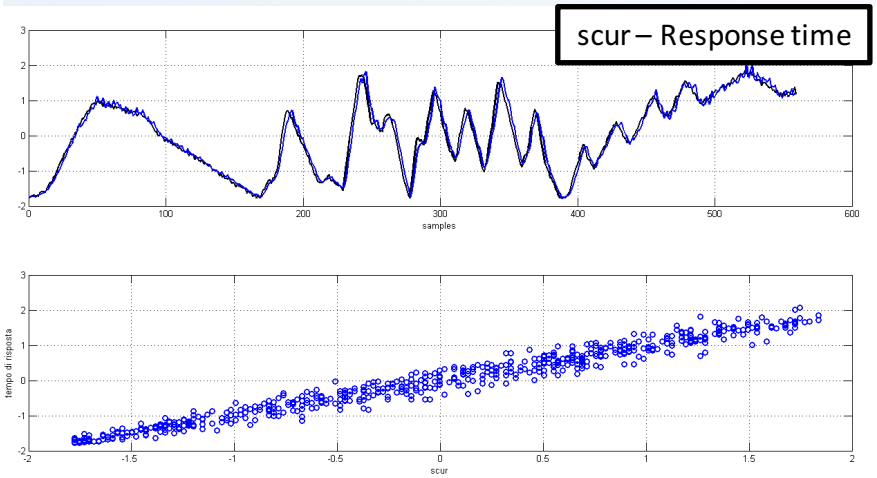
Metric	Correlation
qcur	0.1115
rate	0.1462
req_rate	0.1462
scur	0.9838
bin	0.2986
bout	0.2993
cop	−0.4952

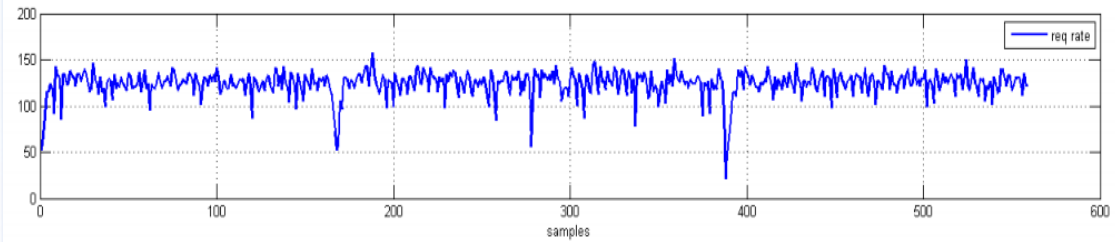
- Feature Selection
  - $\rho = \frac{Cov_{x,y}}{\sigma_x \sigma_y}$
  - Response time = *rtime* + *qtime*

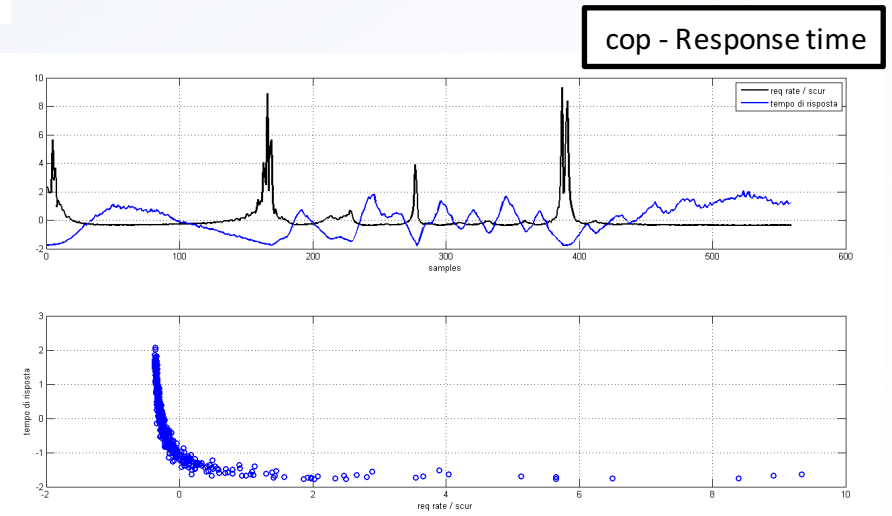
- Response time model function $\hat{y}(k) = func(act(k), scur(k), reqRate(k)/scur(k), reqType(k))$
- Workload Model function $scur(k) = wma(scur(k)) + \frac{wma(scur(k)) - wma(scur(k-1))}{T}$

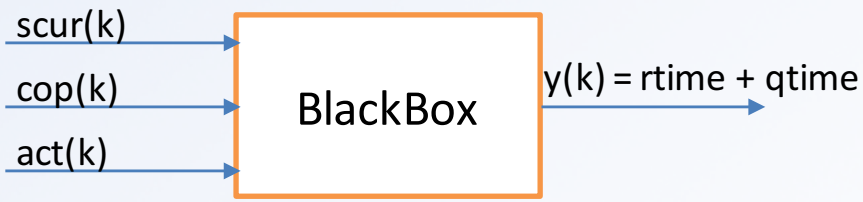
Experimentation (3)







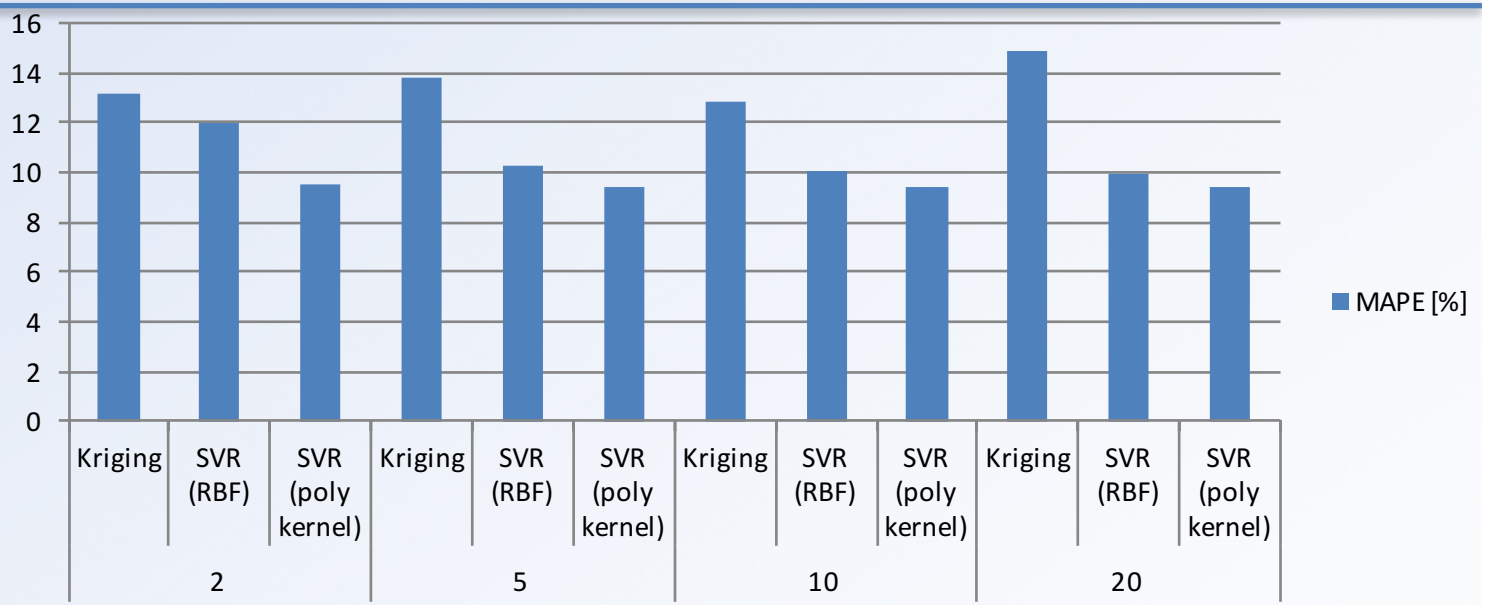




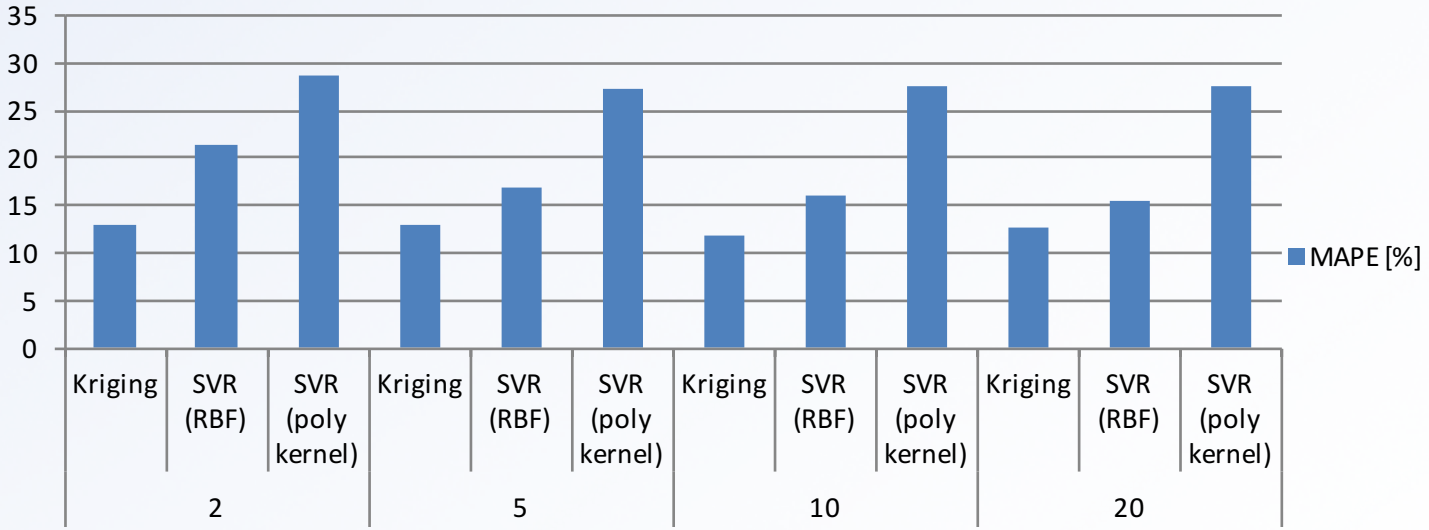
Black-box models comparison

K-fold cross validation

- 1 Instance



- 2 Instances

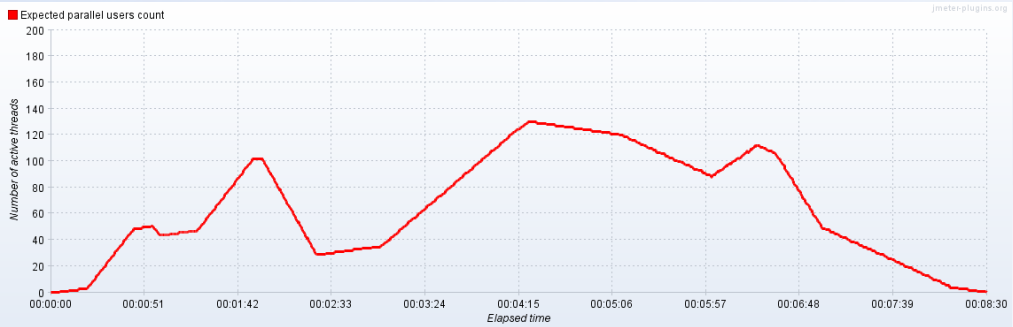


Preliminary results

Evaluation (1)

Test Set

- Workload Test 1



- Workload Test 2

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

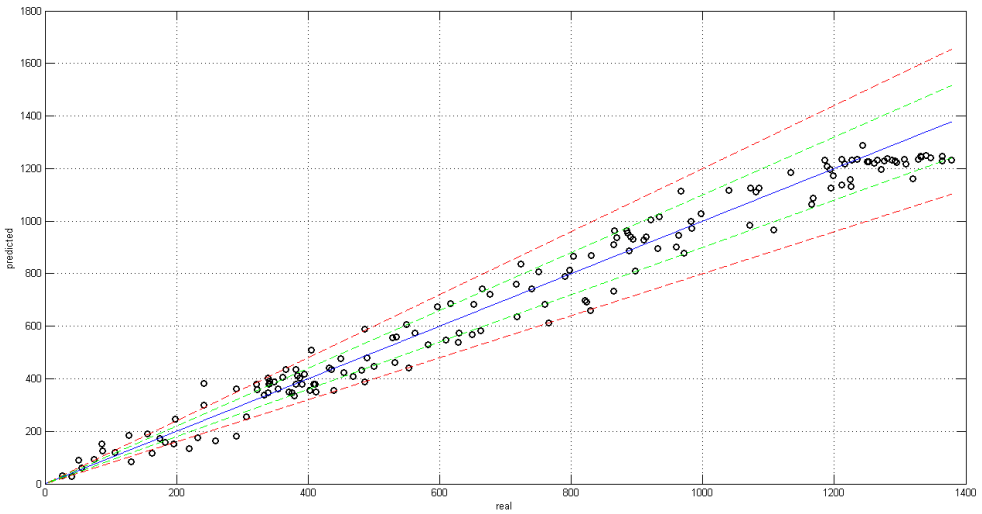
$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

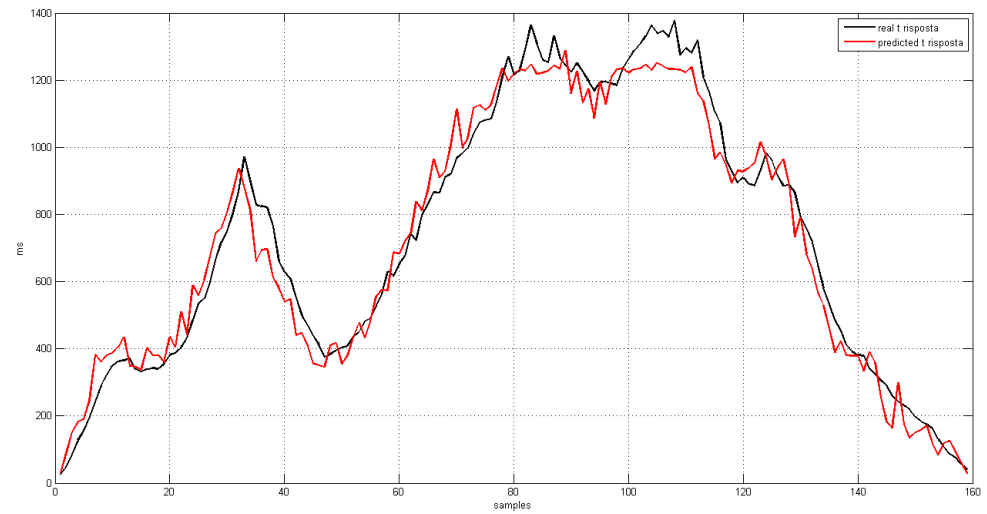
$$\text{MAPE} = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Error	**TestSet 1**	**TestSet 2**
$MAPE[\%]$	$11,30$	$13,23$
$MAE[ms]$	$55,86$	$72,16$
$RMSE[ms]$	$67,419$	$91,8062$

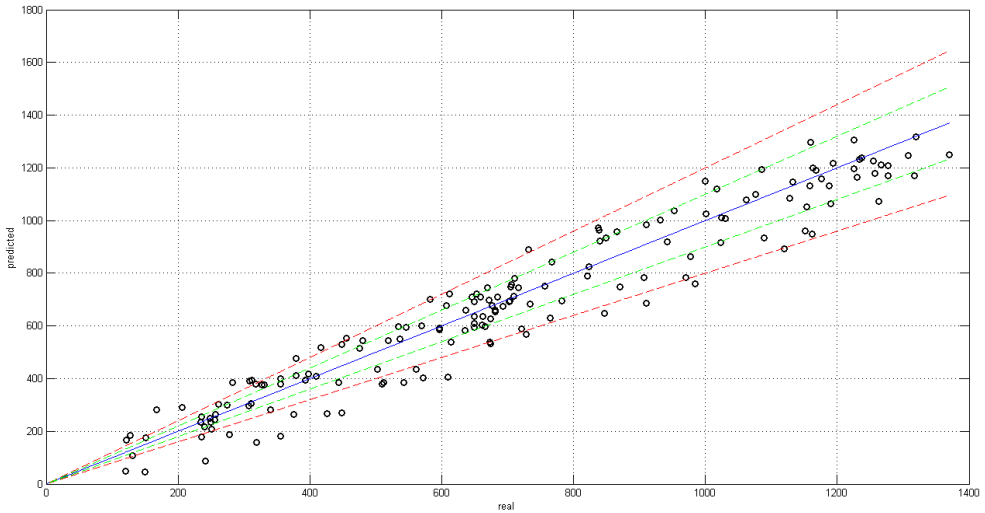
Evaluation (2)

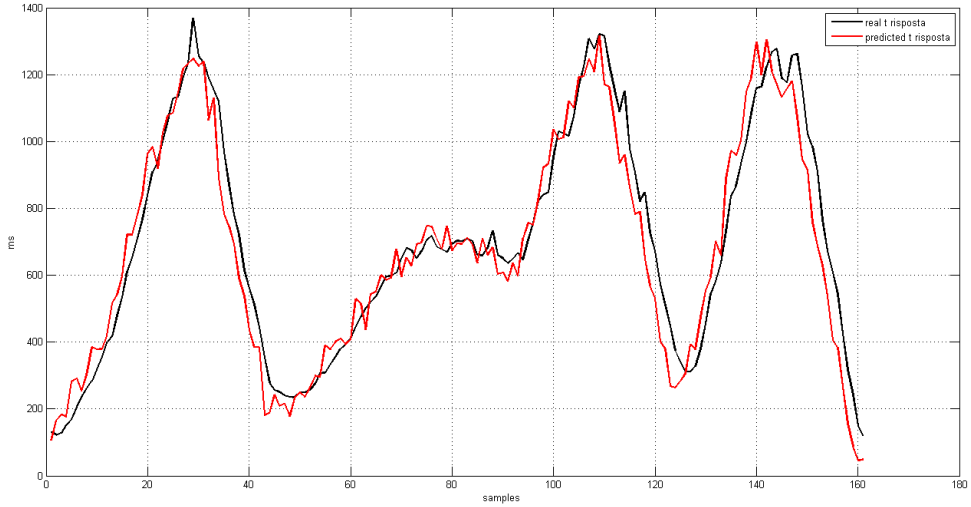
Test set 1





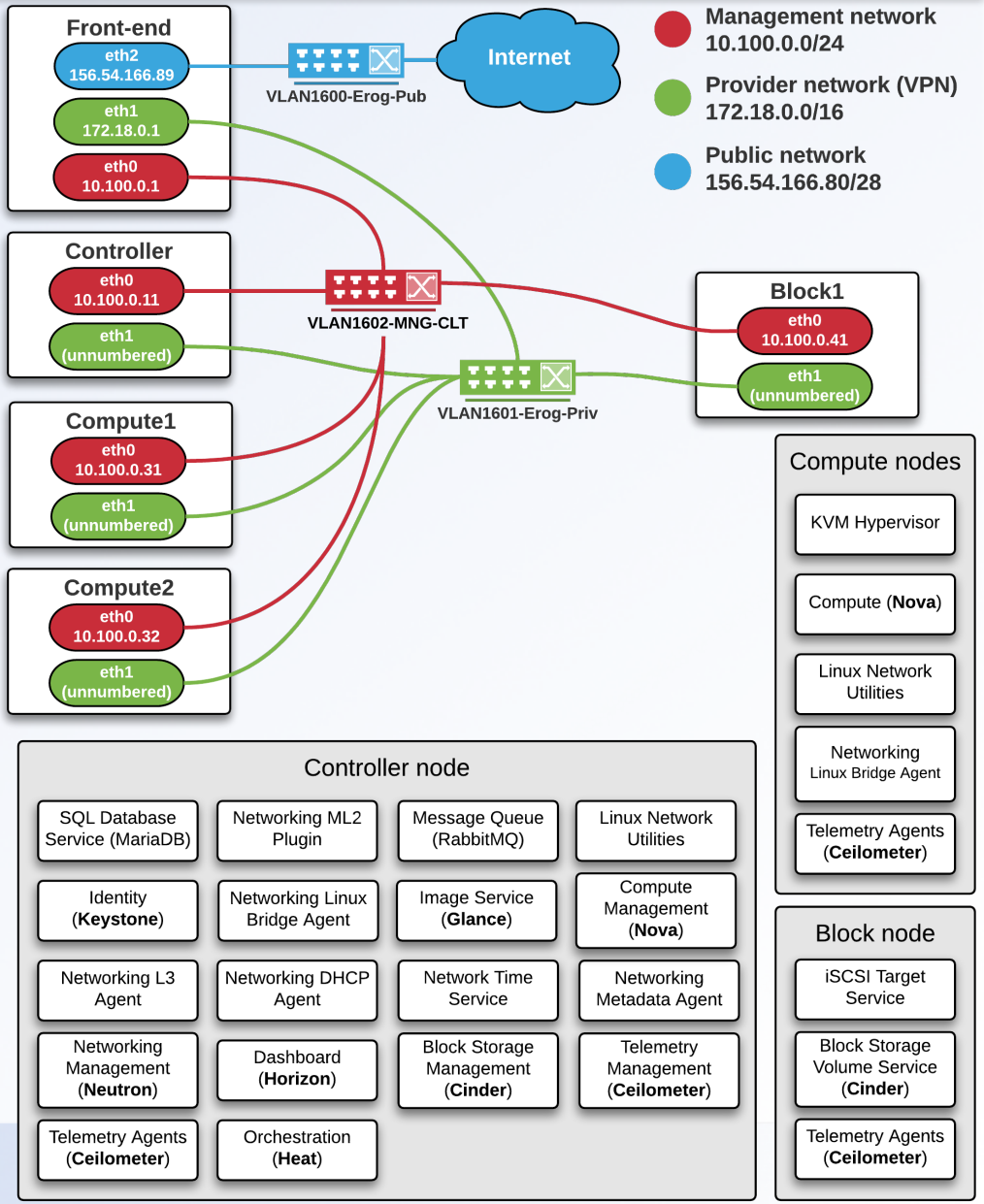
Test set 2





New environment configuration

- Installation and configuration of **OpenStack Mitaka** on a **5-node cluster computer**, each with at least 2 NICs and CentOS 7
  - **Front-end**
    - Provides Internet access for management and provider networks
  - **Controller**
    - Hosts DBs, NTP server, MQ server, dashboard (**Horizon**), networking agents (**Neutron**), Compute management (**Nova**), Identity service (**Keystone**), and Image service (**Glance**)
  - **Compute1** and **Compute2**
    - Host guest VMs through KVM hypervisor
  - **Block1**
    - Provides general-purpose block storage for VMs
- Installation and configuration of the **Cloudify Manager** (on a running VM with CentOS 7)

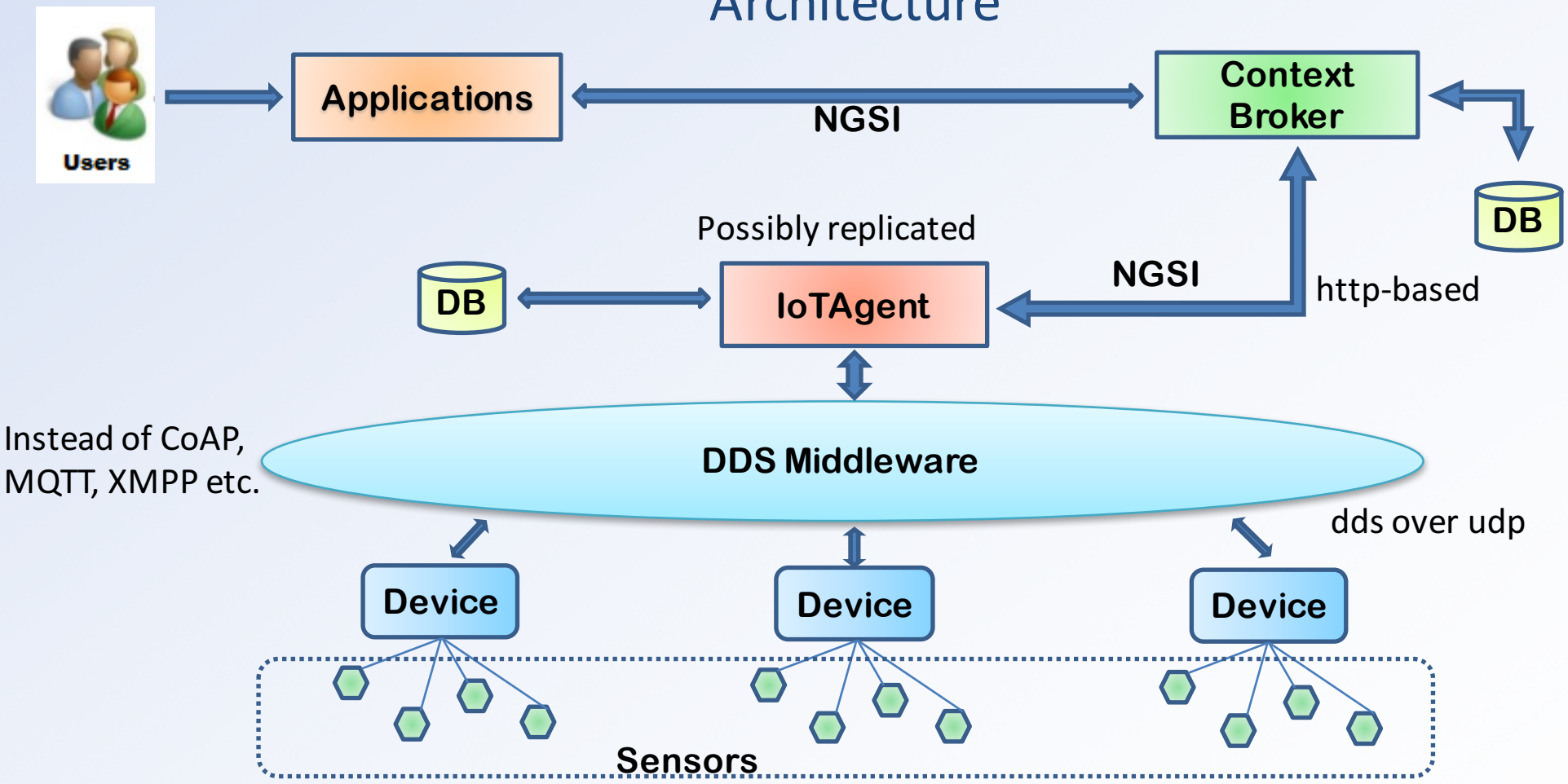


Dependable systems: reconfiguration

- Applications
  - Adding, removing, moving activities in business logic to achieve a prefixed goal (e.g. workflow reconfiguration/autonomicity)
- Execution infrastructure: Cloud
  - Adding, removing VMs or containers to satisfy SLAs constraints (e.g. response time, availability)
- Communication middleware
  - Avoiding single points of failure and managing replicas
    - Passive vs Active replication
    - Consistency properties
  - Context-broker and DDS experience

Reliable context brokering

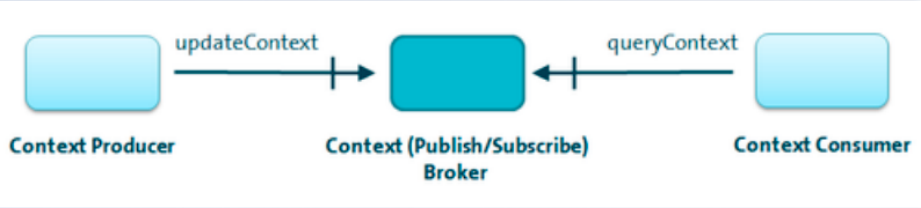
Architecture

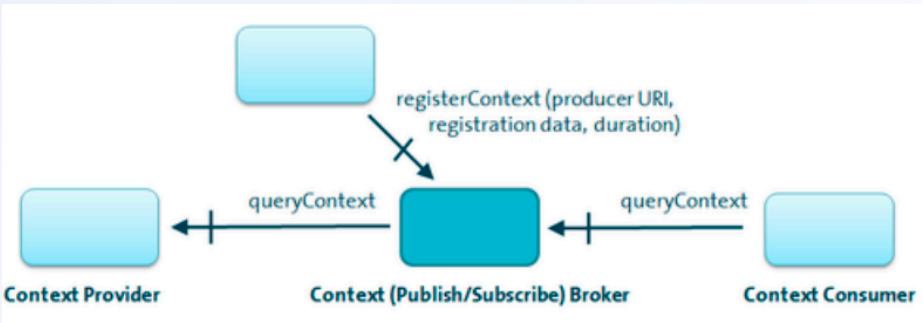


IoT scenario

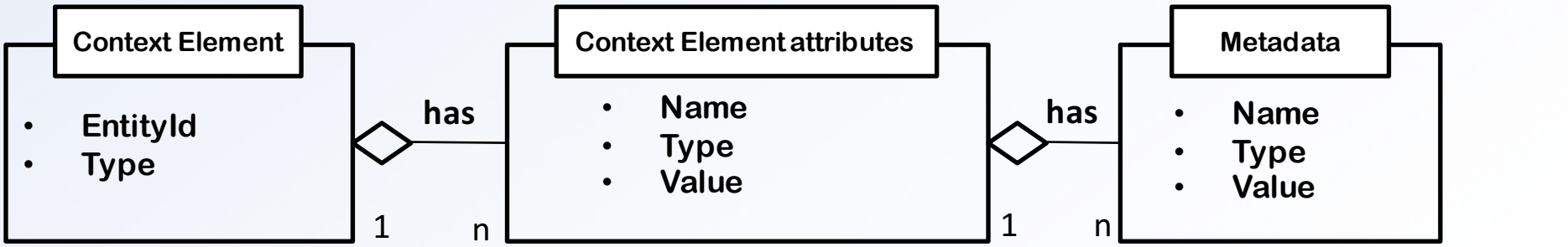
IoT context management

- Main features:
  - Context management (NGSI-10)
  - Context availability management (NGSI-9)
- HTTP e REST-based
  - JSON





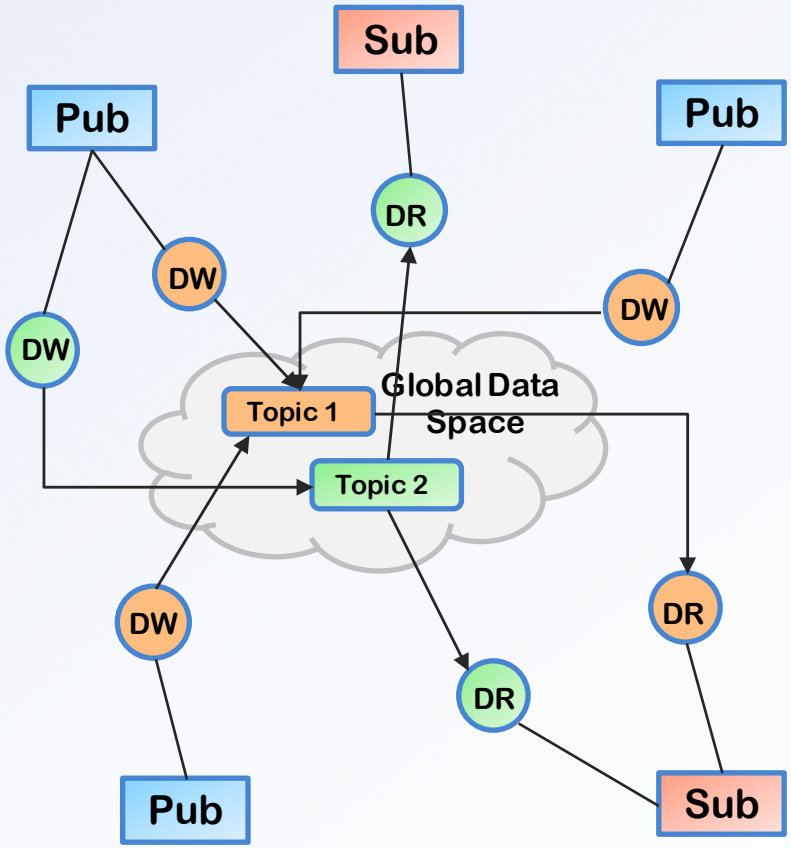
- Data model:



DDS middleware

- System components:
  - Global Data Space
  - Publisher
  - Subscriber
  - Topic
  - Data Writer
  - Data Reader

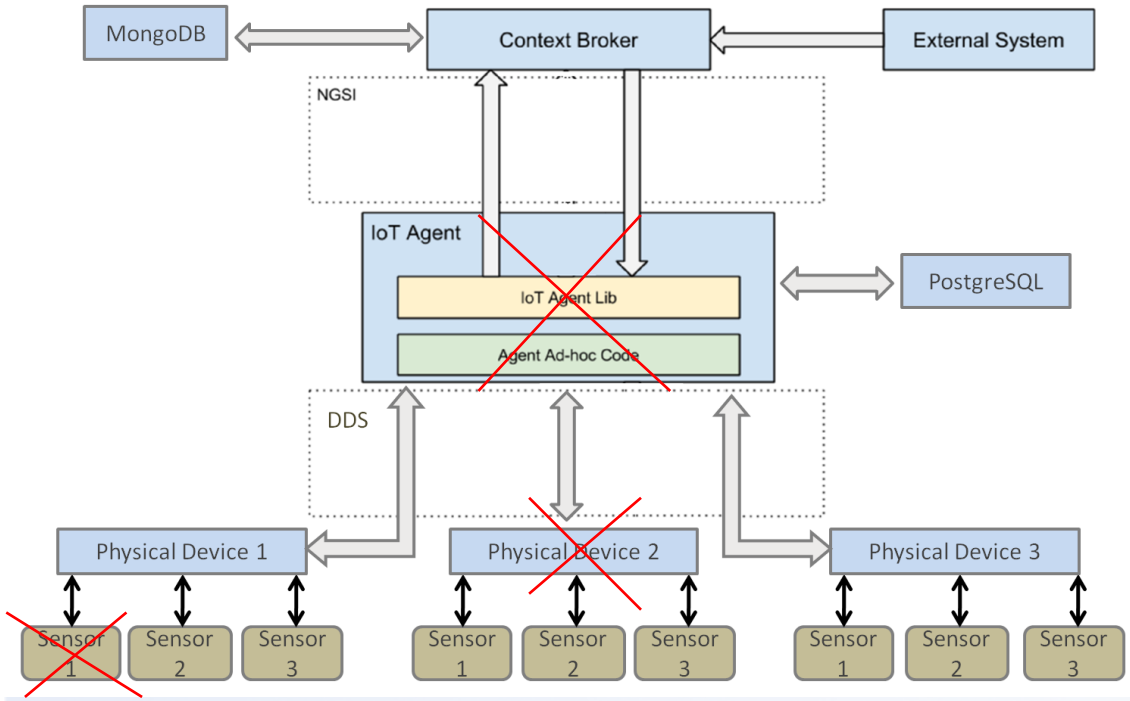
Fully decentralized



Resilience by replication

- Passive replication
  - Only one replica is active at a given time
  - Heartbeat and failover techniques used to elect a new active replica
- Active replication
  - Many replicas are active at a given time
  - All replicas are updated contemporarily (for example through multicast) or capture data from the env.
  - One active replica should be elected as *master* by other nodes to avoid duplicates

Reconfiguration for failure recovery



How to avoid losses and reduce messages?

Tuning between heartbeats (for liveliness) and application sample frequency

How to overcome Byzantine failures ?

Timestamp	Entity	Event
10:17:05.116	DataReaderImpl	handle timeout: reader {id reader} has 2 live writers; from reactor=1
10:17:05.220	TcpConnection	handle timeout, we tried and failed to re-establish connection on transport: OPENDDS 0500 TCP
10:17:05.220	DataReaderImpl	remove associations: bit 0 local
10:17:05.783	DataReaderImpl	writer became dead: reader {id reader} from writer {id writer} previous state ALIVE.
10:17:05.783	Ownership Manager	broadcast new owner: owner writer {id writer}, instance handle 6 strength 5 num of candidates 0
10:17:05.784	ConditionImpl	signal all
10:17:05.784	DataReaderImpl	notify liveliness change: NOTIFY: subscription {id subscriber}, listener at: {id listener} NOTIFY: writer[id writer] == DEAD NOTIFY: writer[id writer] == ALIVE
10:17:05.784	DataReaderImpl	handle timeout: reader {id reader} has 1 live writers; from reactor=1

Example of reconfiguration time using OpenDDS