Mimicking FogDirector Application Management

Stefano Forti, Ahmad Ibrahim and Antonio Brogi

Service-oriented, Cloud and Fog Computing Research Group

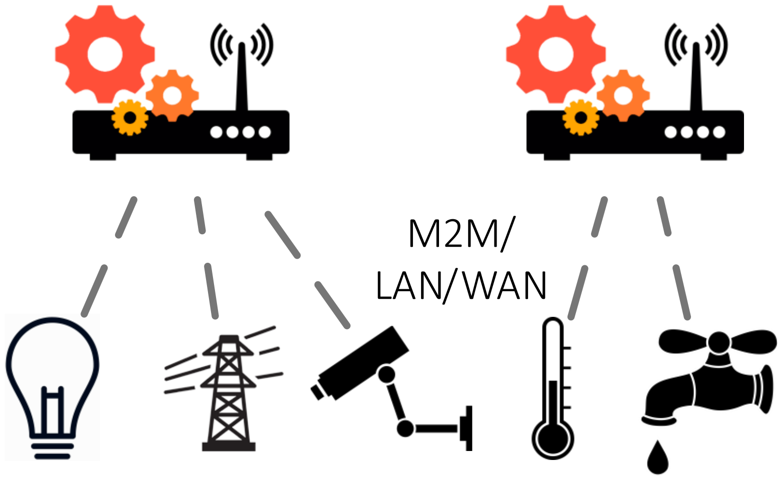
Department of Computer Science

University of Pisa, Italy

12th Symposium and Summer School On Service-Oriented Computing, June 25 – June 29, 2018 in Crete, Greece

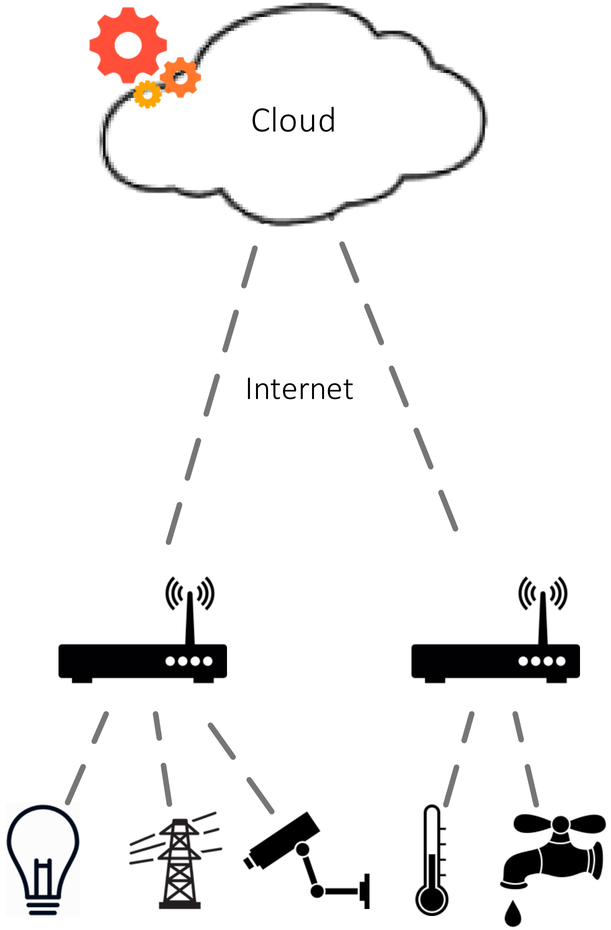
IoT Deployment Models

- Not sufficient *per se* to support the **IoT momentum** alone.
- There is a need for **filtering** and **processing** *before* the Cloud.
- Processing should occur wherever it is ***best-placed*** for any given IoT application



IoT+Edge

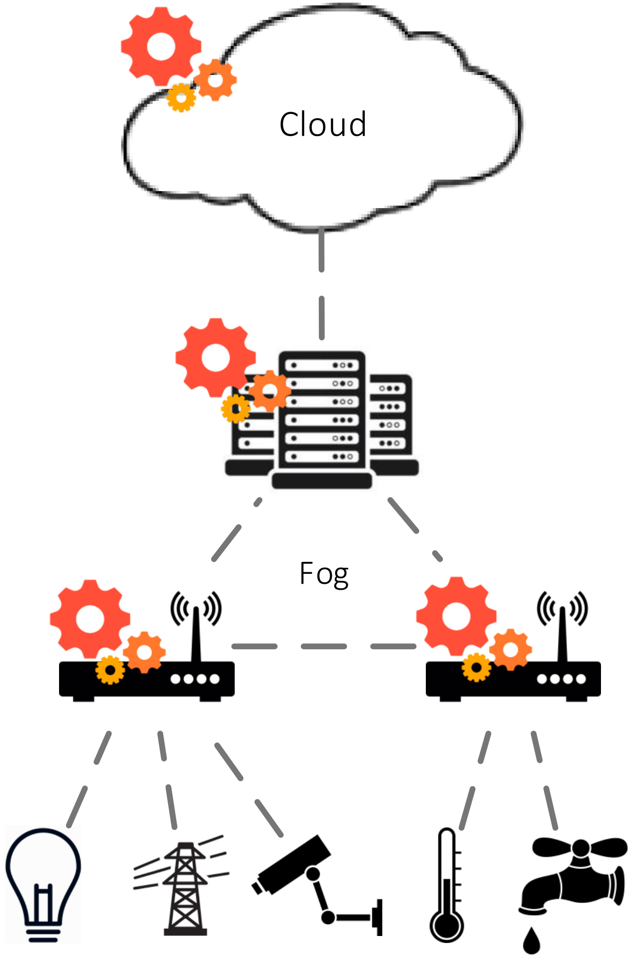
- Low latencies, but
- Limited capabilities,
- Difficulties in sharing data



IoT+Cloud

- Huge computing power, but
- Mandatory connectivity,
- High latencies,
- Bandwidth bottleneck.

Fog Computing



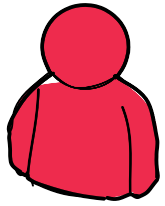
Fog computing is a system-level horizontal architecture that **distributes** resources and services of **computing, storage, control and networking** anywhere along the continuum **from Cloud to Things**, thereby **accelerating** the velocity of **decision making**.

Fog-centric architecture **serves** a specific subset of **business problems that cannot be** successfully **implemented using** only traditional **cloud based architectures or solely intelligent edge devices**.

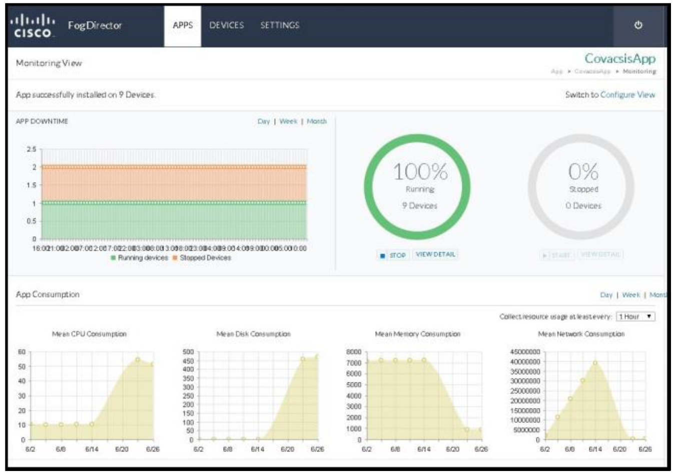
[*OpenFog Reference Architecture*, 2016.]

Fog Director

A single pane of glass to **manage application lifecycle** on Fog devices.



app admin



Monitoring View

CovacsisApp

App successfully installed on 9 Devices.

Switch to Configure View

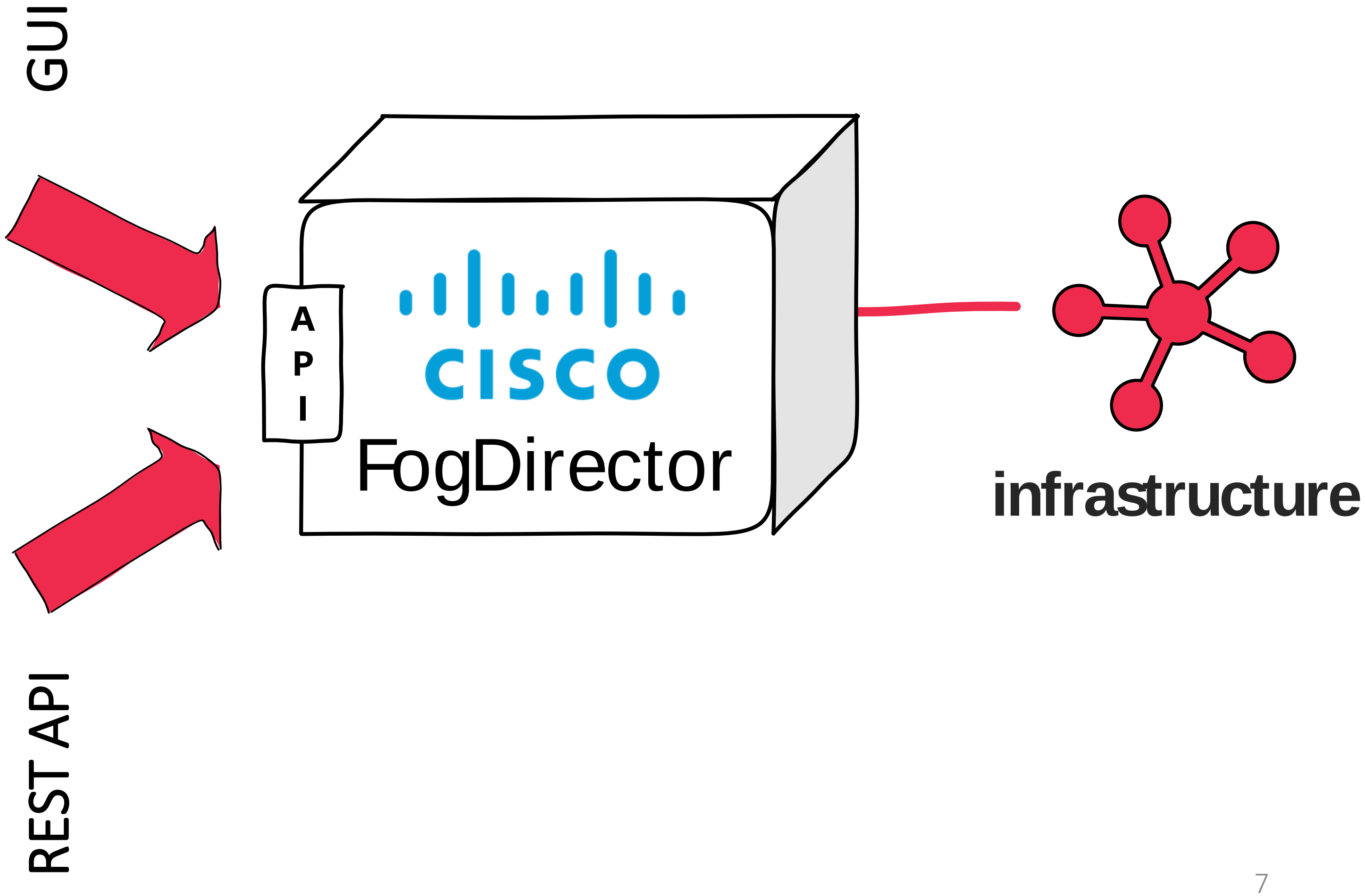
100% Running 9 Devices

0% Stopped 0 Devices

App Consumption

Collect resource usage at least every: 1 Hour

```
# management.py
…
publish(id,A)
n = choose(get_info_N())
d = deploy(id,n)
start(d)
…
on alert do
   stop(d)
   undeploy(d);
   m = choose(get_info_N()))
   deploy(id, m)
…
```



The App

Steve, App Admin

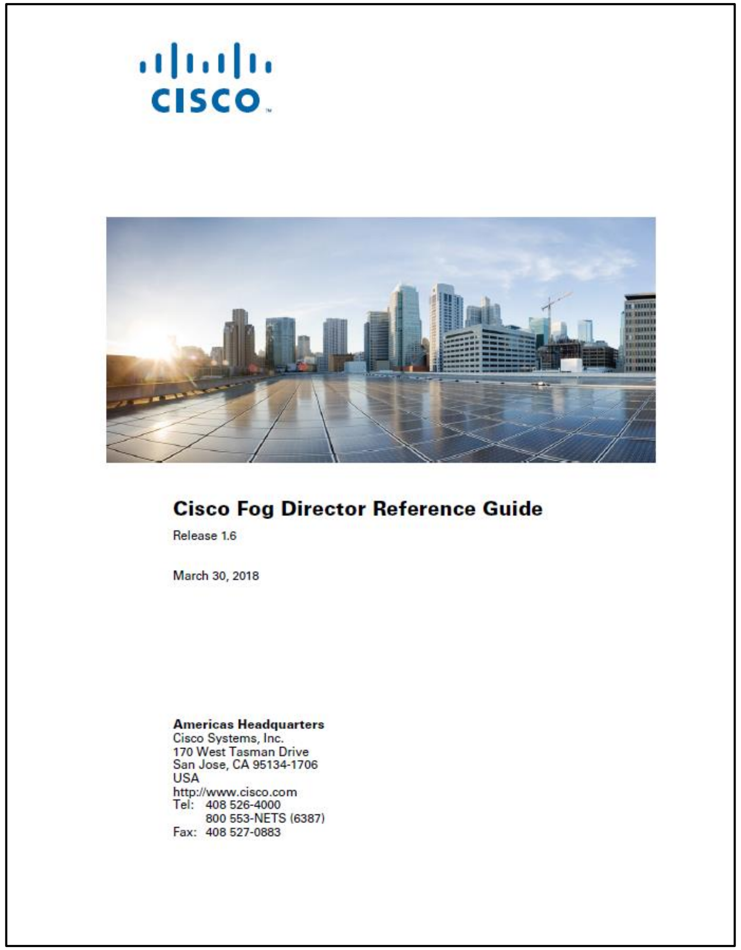
SmartBuilding

Problems

#1 (quickly) **understand FogDirector** functioning

#2 write **correct** and **effective** management

Problem #1



Cisco Fog Director Reference Guide

Release 1.6

March 30, 2018

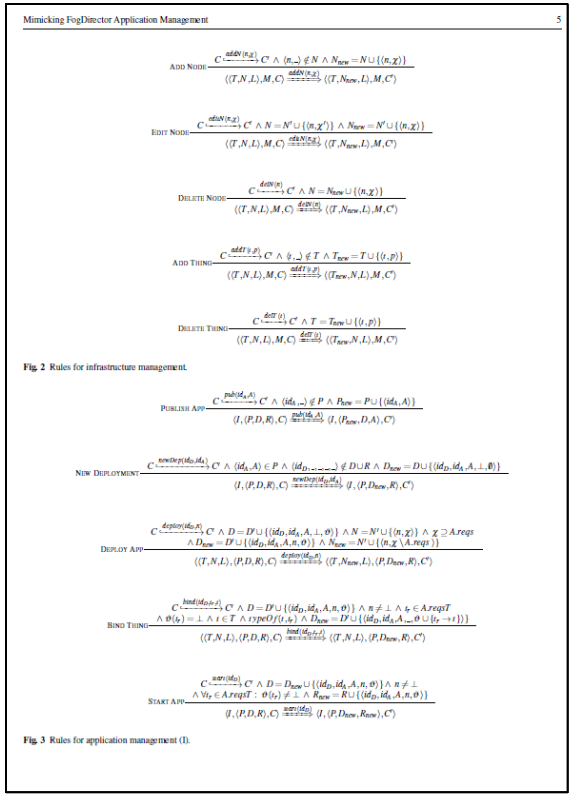
Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

!!!
160 PAGES
!!!

https://www.cisco.com/c/en/us/td/docs/routers/access/800/software/guides/iox/fog-director/reference-guide/1-6/fog_director_ref_guide.html

Solution #1

- **Operational semantics** of all **basic functionalities** of FogDirector.
- **Compact** and **unambiguous** reference.



Mimicking FogDirector Application Management 5

$$\textsc{Add Node}\;\dfrac{C \xhookrightarrow{addN(n,\chi)} C' \wedge \langle n,\_\rangle \notin N \wedge N_{new} = N \cup \{\langle n,\chi\rangle\}}{\langle\langle T,N,L\rangle,M,C\rangle \xRightarrow{addN(n,\chi)} \langle\langle T,N_{new},L\rangle,M,C'\rangle}$$

$$\textsc{Edit Node}\;\dfrac{C \xhookrightarrow{editN(n,\chi)} C' \wedge N = N' \cup \{\langle n,\chi'\rangle\} \wedge N_{new} = N' \cup \{\langle n,\chi\rangle\}}{\langle\langle T,N,L\rangle,M,C\rangle \xRightarrow{editN(n,\chi)} \langle\langle T,N_{new},L\rangle,M,C'\rangle}$$

$$\textsc{Delete Node}\;\dfrac{C \xhookrightarrow{delN(n)} C' \wedge N = N_{new} \cup \{\langle n,\chi\rangle\}}{\langle\langle T,N,L\rangle,M,C\rangle \xRightarrow{delN(n)} \langle\langle T,N_{new},L\rangle,M,C'\rangle}$$

$$\textsc{Add Thing}\;\dfrac{C \xhookrightarrow{addT(t,p)} C' \wedge \langle t,\_\rangle \notin T \wedge T_{new} = T \cup \{\langle t,p\rangle\}}{\langle\langle T,N,L\rangle,M,C\rangle \xRightarrow{addT(t,p)} \langle\langle T_{new},N,L\rangle,M,C'\rangle}$$

$$\textsc{Delete Thing}\;\dfrac{C \xhookrightarrow{delT(t)} C' \wedge T = T_{new} \cup \{\langle t,p\rangle\}}{\langle\langle T,N,L\rangle,M,C\rangle \xRightarrow{delT(t)} \langle\langle T_{new},N,L\rangle,M,C'\rangle}$$

Fig. 2 Rules for infrastructure management.

$$\textsc{Publish App}\;\dfrac{C \xhookrightarrow{pub(id_A,A)} C' \wedge \langle id_A,\_\rangle \notin P \wedge P_{new} = P \cup \{\langle id_A,A\rangle\}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{pub(id_A,A)} \langle I,\langle P_{new},D,A\rangle,C'\rangle}$$

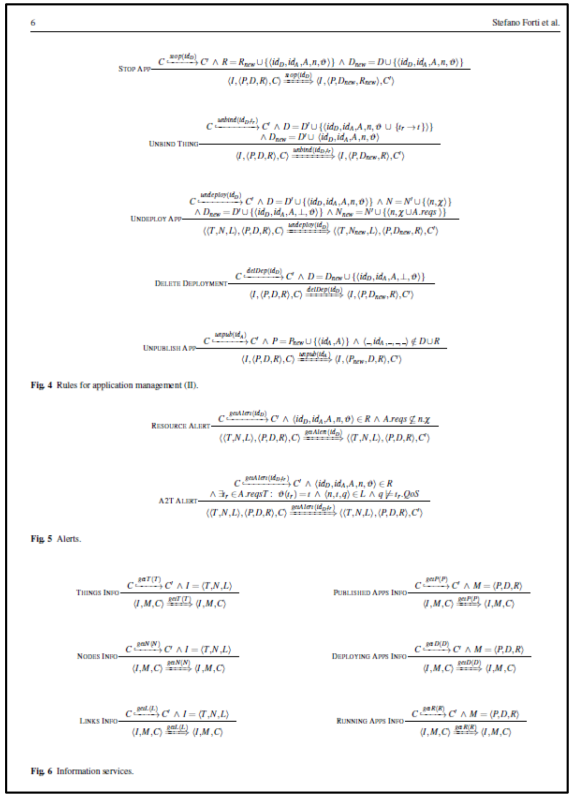
$$\textsc{New Deployment}\;\dfrac{C \xhookrightarrow{newDep(id_D,id_A)} C' \wedge \langle id_A,A\rangle \in P \wedge \langle id_D,\_,\_,\_,\_\rangle \notin D \cup R \wedge D_{new} = D \cup \{\langle id_D,id_A,A,\bot,\emptyset\rangle\}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{newDep(id_D,id_A)} \langle I,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Deploy App}\;\dfrac{\begin{array}{c}C \xhookrightarrow{deploy(id_D,n)} C' \wedge D = D' \cup \{\langle id_D,id_A,A,\bot,\vartheta\rangle\} \wedge N = N' \cup \{\langle n,\chi\rangle\} \wedge \chi \supseteq A.reqs\\ \wedge D_{new} = D' \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\} \wedge N_{new} = N' \cup \{\langle n,\chi \setminus A.reqs\rangle\}\end{array}}{\langle\langle T,N,L\rangle,\langle P,D,R\rangle,C\rangle \xRightarrow{deploy(id_D,n)} \langle\langle T,N_{new},L\rangle,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Bind Thing}\;\dfrac{\begin{array}{c}C \xhookrightarrow{bind(id_D,t_r,t)} C' \wedge D = D' \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\} \wedge n \neq \bot \wedge t_r \in A.reqsT\\ \wedge \vartheta(t_r) = \bot \wedge t \in T \wedge typeOf(t,t_r) \wedge D_{new} = D' \cup \{\langle id_D,id_A,A,\_,\vartheta \cup \{t_r \to t\}\rangle\}\end{array}}{\langle\langle T,N,L\rangle,\langle P,D,R\rangle,C\rangle \xRightarrow{bind(id_D,t_r,t)} \langle\langle T,N,L\rangle,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Start App}\;\dfrac{\begin{array}{c}C \xhookrightarrow{start(id_D)} C' \wedge D = D_{new} \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\} \wedge n \neq \bot\\ \wedge \forall t_r \in A.reqsT:\ \vartheta(t_r) \neq \bot \wedge R_{new} = R \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\}\end{array}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{start(id_D)} \langle I,\langle P,D_{new},R_{new}\rangle,C'\rangle}$$

Fig. 3 Rules for application management (I).



6 Stefano Forti et al.

$$\textsc{Stop App}\;\dfrac{C \xhookrightarrow{stop(id_D)} C' \wedge R = R_{new} \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\} \wedge D_{new} = D \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{stop(id_D)} \langle I,\langle P,D_{new},R_{new}\rangle,C'\rangle}$$

$$\textsc{Unbind Thing}\;\dfrac{\begin{array}{c}C \xhookrightarrow{unbind(id_D,t_r)} C' \wedge D = D' \cup \{\langle id_D,id_A,A,n,\vartheta \cup \{t_r \to t\}\rangle\}\\ \wedge D_{new} = D' \cup \langle id_D,id_A,A,n,\vartheta\rangle\end{array}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{unbind(id_D,t_r)} \langle I,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Undeploy App}\;\dfrac{\begin{array}{c}C \xhookrightarrow{undeploy(id_D)} C' \wedge D = D' \cup \{\langle id_D,id_A,A,n,\vartheta\rangle\} \wedge N = N' \cup \{\langle n,\chi\rangle\}\\ \wedge D_{new} = D' \cup \{\langle id_D,id_A,A,\bot,\vartheta\rangle\} \wedge N_{new} = N' \cup \{\langle n,\chi \cup A.reqs\rangle\}\end{array}}{\langle\langle T,N,L\rangle,\langle P,D,R\rangle,C\rangle \xRightarrow{undeploy(id_D)} \langle\langle T,N_{new},L\rangle,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Delete Deployment}\;\dfrac{C \xhookrightarrow{delDep(id_D)} C' \wedge D = D_{new} \cup \{\langle id_D,id_A,A,\bot,\vartheta\rangle\}}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{delDep(id_D)} \langle I,\langle P,D_{new},R\rangle,C'\rangle}$$

$$\textsc{Unpublish App}\;\dfrac{C \xhookrightarrow{unpub(id_A)} C' \wedge P = P_{new} \cup \{\langle id_A,A\rangle\} \wedge \langle \_,id_A,\_,\_,\_\rangle \notin D \cup R}{\langle I,\langle P,D,R\rangle,C\rangle \xRightarrow{unpub(id_A)} \langle I,\langle P_{new},D,R\rangle,C'\rangle}$$

Fig. 4 Rules for application management (II).

$$\textsc{Resource Alert}\;\dfrac{C \xhookrightarrow{getAlert(id_D)} C' \wedge \langle id_D,id_A,A,n,\vartheta\rangle \in R \wedge A.reqs \not\subseteq n.\chi}{\langle\langle T,N,L\rangle,\langle P,D,R\rangle,C\rangle \xRightarrow{getAlert(id_D)} \langle\langle T,N,L\rangle,\langle P,D,R\rangle,C'\rangle}$$

$$\textsc{A2T Alert}\;\dfrac{\begin{array}{c}C \xhookrightarrow{getAlert(id_D,t_r)} C' \wedge \langle id_D,id_A,A,n,\vartheta\rangle \in R\\ \wedge \exists t_r \in A.reqsT:\ \vartheta(t_r) = t \wedge \langle n,t,q\rangle \in L \wedge q \not\models t_r.QoS\end{array}}{\langle\langle T,N,L\rangle,\langle P,D,R\rangle,C\rangle \xRightarrow{getAlert(id_D,t_r)} \langle\langle T,N,L\rangle,\langle P,D,R\rangle,C'\rangle}$$

Fig. 5 Alerts.

$$\textsc{Things Info}\;\dfrac{C \xhookrightarrow{getT(T)} C' \wedge I = \langle T,N,L\rangle}{\langle I,M,C\rangle \xRightarrow{getT(T)} \langle I,M,C\rangle} \qquad \textsc{Published Apps Info}\;\dfrac{C \xhookrightarrow{getP(P)} C' \wedge M = \langle P,D,R\rangle}{\langle I,M,C\rangle \xRightarrow{getP(P)} \langle I,M,C\rangle}$$

$$\textsc{Nodes Info}\;\dfrac{C \xhookrightarrow{getN(N)} C' \wedge I = \langle T,N,L\rangle}{\langle I,M,C\rangle \xRightarrow{getN(N)} \langle I,M,C\rangle} \qquad \textsc{Deploying Apps Info}\;\dfrac{C \xhookrightarrow{getD(D)} C' \wedge M = \langle P,D,R\rangle}{\langle I,M,C\rangle \xRightarrow{getD(D)} \langle I,M,C\rangle}$$

$$\textsc{Links Info}\;\dfrac{C \xhookrightarrow{getL(L)} C' \wedge I = \langle T,N,L\rangle}{\langle I,M,C\rangle \xRightarrow{getL(L)} \langle I,M,C\rangle} \qquad \textsc{Running Apps Info}\;\dfrac{C \xhookrightarrow{getR(R)} C' \wedge M = \langle P,D,R\rangle}{\langle I,M,C\rangle \xRightarrow{getR(R)} \langle I,M,C\rangle}$$

Fig. 6 Information services.

ONLY 2 PAGES*!
☺

Anatomy of a rule

Client Management Program

Operation and Parameters

Conditions for the operation to be successful

Error Code

$$\frac{C \xhookrightarrow{op(p_1,\cdots,p_s,e)} C' \wedge c_1 \wedge c_2 \wedge \cdots \wedge c_k \wedge e = 0}{\langle I,M,C \rangle \xRightarrow{op(p_1,\cdots,p_s,e)} \langle I',M',C' \rangle}$$

Infrastructure State

Managed Apps

FAILURE

$$\frac{C \xhookrightarrow{op(p_1,\cdots,p_s,e)} C' \wedge \neg(c_1 \wedge c_2 \wedge \cdots \wedge c_k) \wedge e = -1}{\langle I,M,C \rangle \xRightarrow{op(p_1,\cdots,p_s,e)} \langle I',M',C' \rangle}$$

Problem #2

write **correct** and **effective** management

- **Correctness** can be verified by using the semantics.
- **Effectiveness** involves considering variations in:
  - Fog node **resources** and
  - QoS of **communication links**

- **What then?**

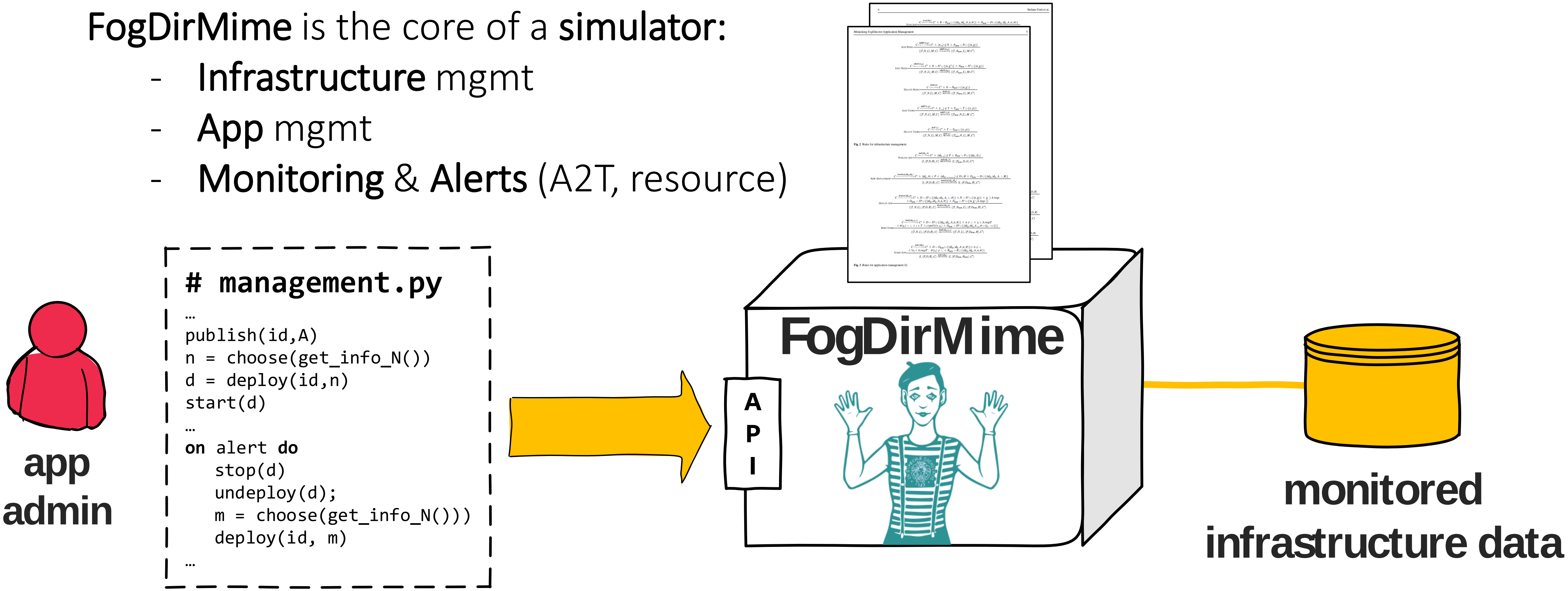
Problem #2

write **correct** and **effective** management

Solution #2

FogDirMime is the core of a **simulator:**

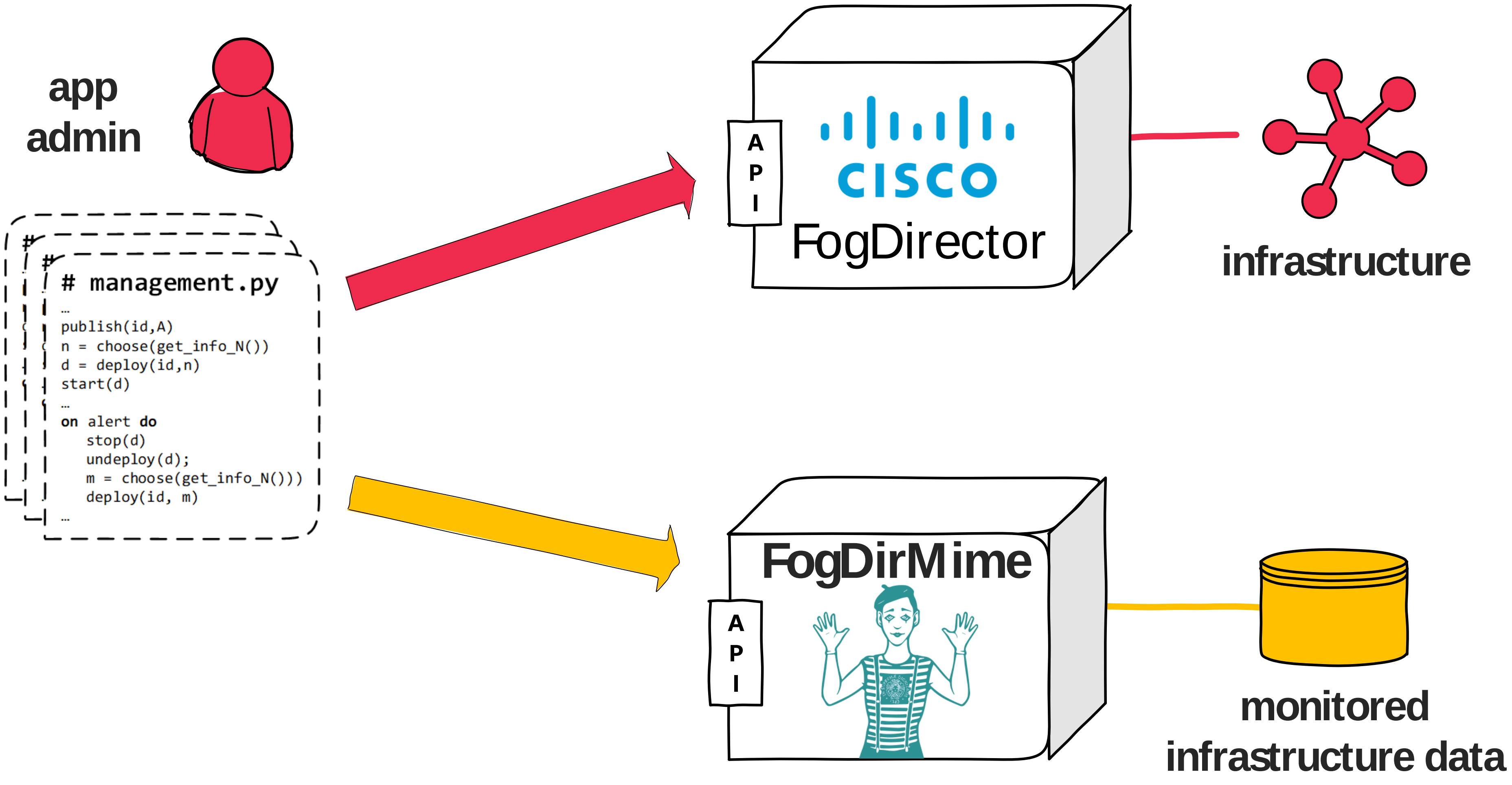
- **Infrastructure** mgmt
- **App** mgmt
- **Monitoring** & **Alerts** (A2T, resource)



```
# management.py
…
publish(id,A)
n = choose(get_info_N())
d = deploy(id,n)
start(d)
…
on alert do
    stop(d)
    undeploy(d);
    m = choose(get_info_N()))
    deploy(id, m)
…
```

https://github.com/di-unipi-socc/FogDirMime

The Big Picture

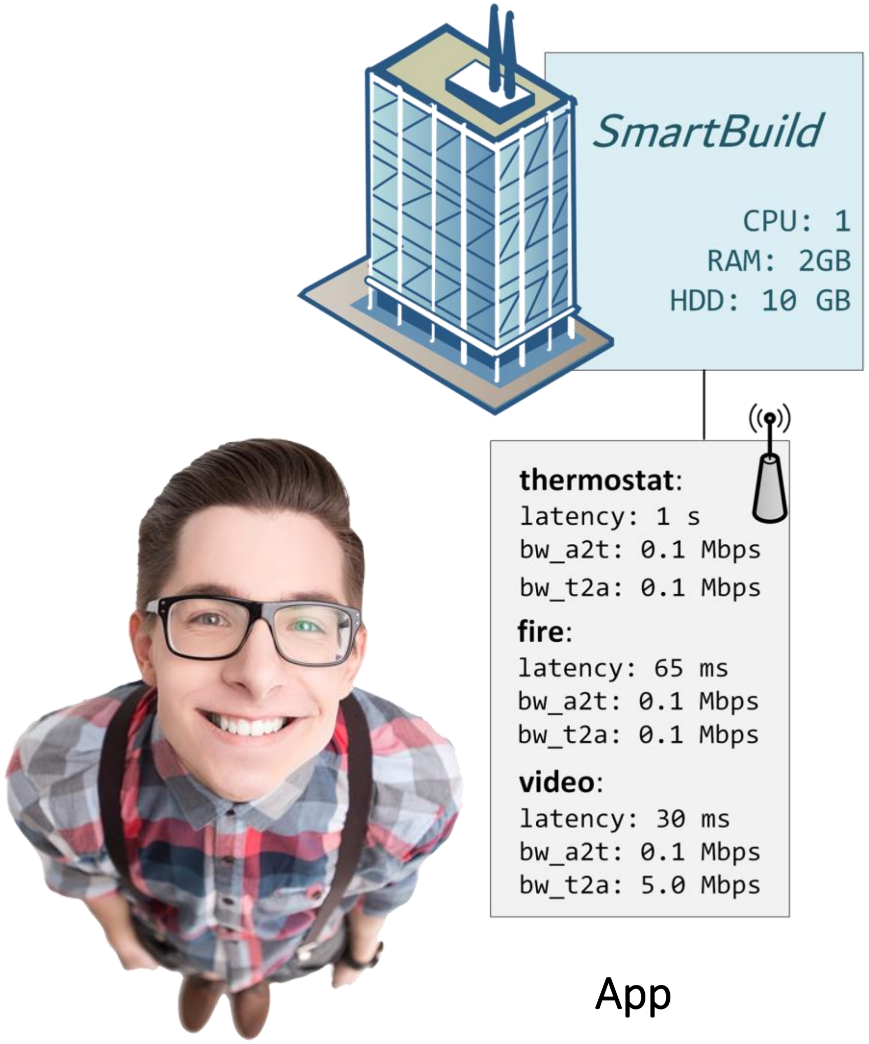


```
# management.py
…
publish(id,A)
n = choose(get_info_N())
d = deploy(id,n)
start(d)
…
on alert do
    stop(d)
    undeploy(d);
    m = choose(get_info_N()))
    deploy(id, m)
…
```

varying network QoS and workload conditions

probabilistic sampling of latency & bandwidth and of available node resources

A (simple) example

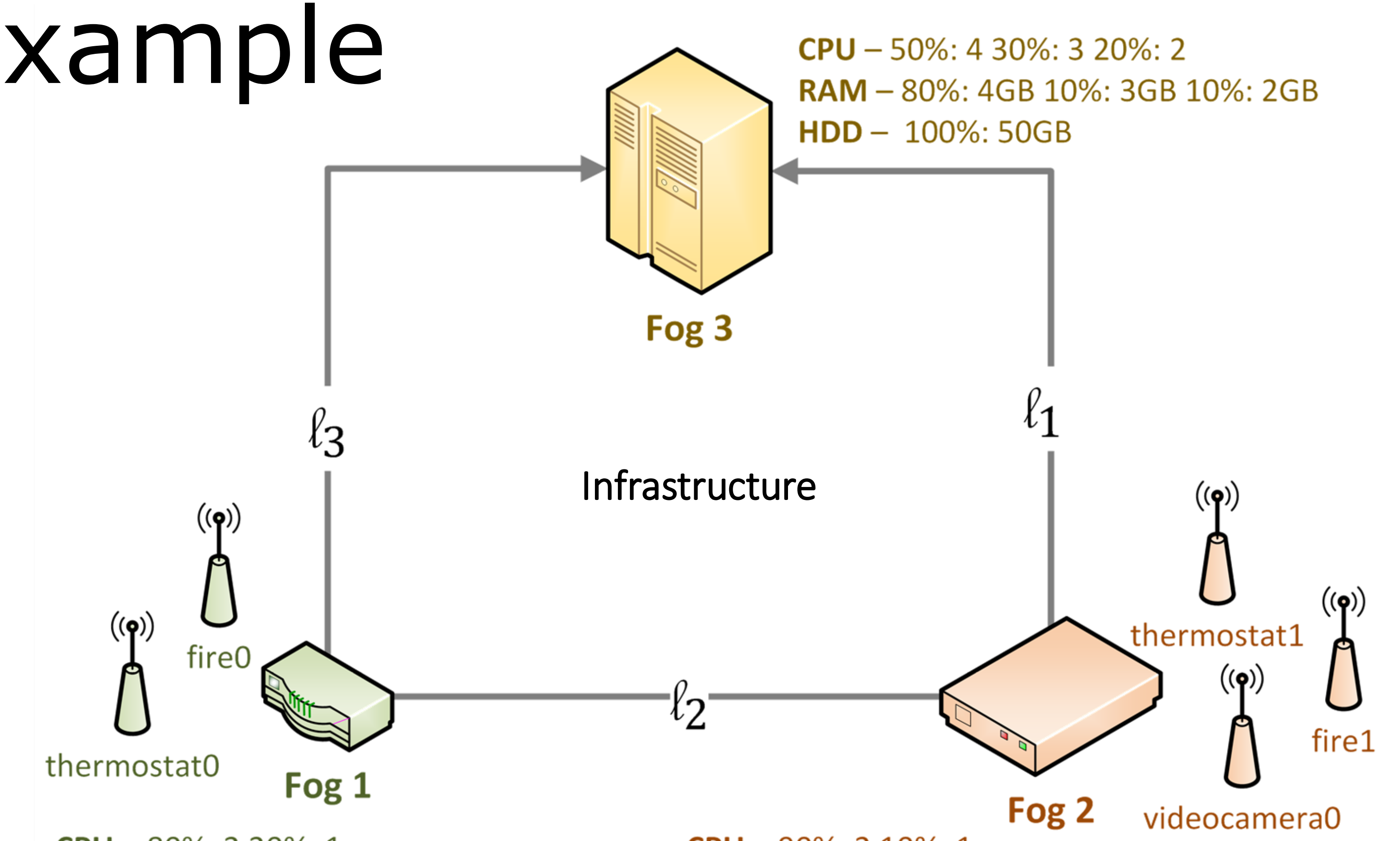


SmartBuild

```
CPU: 1
RAM: 2GB
HDD: 10 GB
```

thermostat:
```
latency: 1 s
bw_a2t: 0.1 Mbps
bw_t2a: 0.1 Mbps
```
fire:
```
latency: 65 ms
bw_a2t: 0.1 Mbps
bw_t2a: 0.1 Mbps
```
video:
```
latency: 30 ms
bw_a2t: 0.1 Mbps
bw_t2a: 5.0 Mbps
```

App



Fog 3

CPU – 50%: 4 30%: 3 20%: 2
RAM – 80%: 4GB 10%: 3GB 10%: 2GB
HDD – 100%: 50GB

Fog 1

CPU – 80%: 2 20%: 1
RAM – 80%: 2GB 10%: 1GB 10%:0.5GB
HDD – 80%: 16GB 10%: 8GB 10%: 4GB

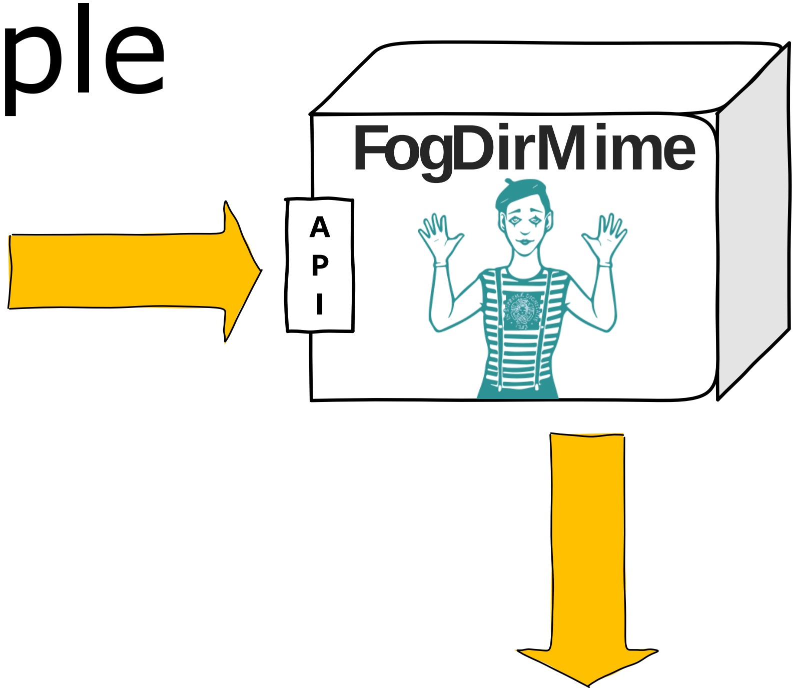
Fog 2

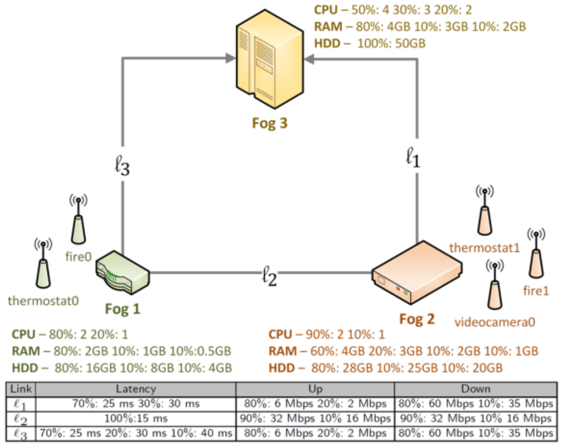
CPU – 90%: 2 10%: 1
RAM – 60%: 4GB 20%: 3GB 10%: 2GB 10%: 1GB
HDD – 80%: 28GB 10%: 25GB 10%: 20GB

Link	Latency	Up	Down
ℓ_1	70%: 25 ms 30%: 30 ms	80%: 6 Mbps 20%: 2 Mbps	80%: 60 Mbps 10%: 35 Mbps
ℓ_2	100%:15 ms	90%: 32 Mbps 10% 16 Mbps	90%: 32 Mbps 10% 16 Mbps
ℓ_3	70%: 25 ms 20%: 30 ms 10%: 40 ms	80%: 6 Mbps 20%: 2 Mbps	80%: 60 Mbps 10%: 35 Mbps

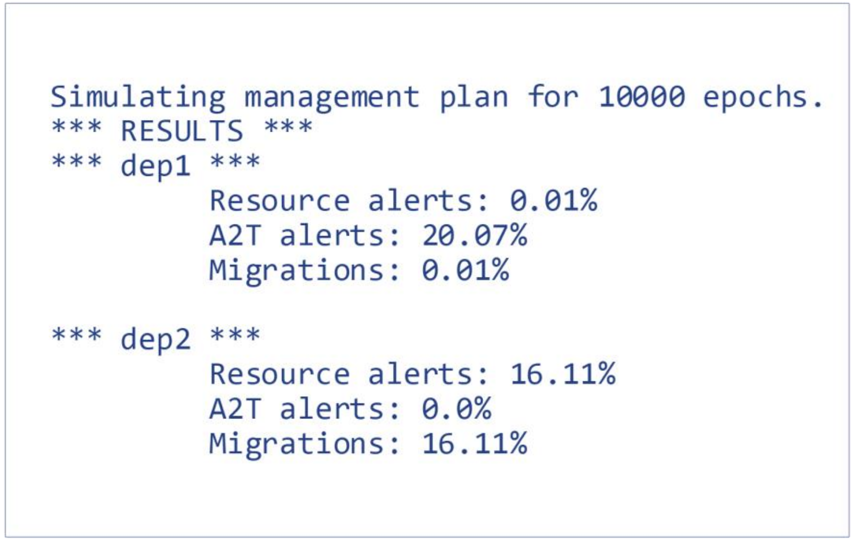
A (simple) example

```
for i in range(0, epochs):
    alerts1=fd.get_alert('dep1')
    alerts2=fd.get_alert('dep2')
    #[...] collecting user defined stats about the alerts [...]
    #In case of low resources at Fog2, move to Fog3
    for alert in alerts1:
        if alert['alert_type'] =='resources' and not(moved1):
            migrations1 +=1
            fd.stop_app('dep1')
            fd.undeploy_app('dep1', 'SmartBuild')
            while fd.deploy_app('dep1', 'SmartBuild', 'fog_3')
                 !=1:
                continue
            fd.start_app('dep1')
            moved1 =True
            break
    #In case of low resources at Fog1, move to Fog2 and
         viceversa
    for alert in alerts2:
        if alert['alert_type'] =='resources':
            migrations2 +=1
            fd.stop_app('dep2')
            fd.undeploy_app('dep2', 'SmartBuild')
            if not(moved2):
                fog_node ='fog_2'
            else:
                fog_node ='fog_1'
            while fd.deploy_app('dep2', 'SmartBuild',
                 fog_node) !=1:
                continue
            fd.start_app('dep2')
            moved2 =not(moved2)
            break
    alerts1, alerts2 =[], []
```




CPU – 50%: 4 30%: 3 20%: 2
RAM – 80%: 4GB 10%: 3GB 10%: 2GB
HDD – 100%: 50GB
Fog 3
ℓ3
ℓ1
ℓ2
fire0
thermostat0
Fog 1
thermostat1
fire1
Fog 2
videocamera0
CPU – 80%: 2 20%: 1
RAM – 80%: 2GB 10%: 1GB 10%:0.5GB
HDD – 80%: 16GB 10%: 8GB 10%: 4GB
CPU – 90%: 2 10%: 1
RAM – 60%: 4GB 20%: 3GB 10%: 2GB 10%: 1GB
HDD – 80%: 28GB 10%: 25GB 10%: 20GB

Link	Latency	Up	Down
ℓ_1	70%: 25 ms 30%: 30 ms	80%: 6 Mbps 20%: 2 Mbps	80%: 60 Mbps 10%: 35 Mbps
ℓ_2	100%:15 ms	90%: 32 Mbps 10% 16 Mbps	90%: 32 Mbps 10% 16 Mbps
ℓ_3	70%: 25 ms 20%: 30 ms 10%: 40 ms	80%: 6 Mbps 20%: 2 Mbps	80%: 60 Mbps 10%: 35 Mbps



```
Simulating management plan for 10000 epochs.
*** RESULTS ***
*** dep1 ***
        Resource alerts: 0.01%
        A2T alerts: 20.07%
        Migrations: 0.01%

*** dep2 ***
        Resource alerts: 16.11%
        A2T alerts: 0.0%
        Migrations: 16.11%
```

Conclusions

concise and **unambiguous reference** for **FogDirector**

validation of management scripts **at design time**

performance prediction and **tuning** of management

Future Work

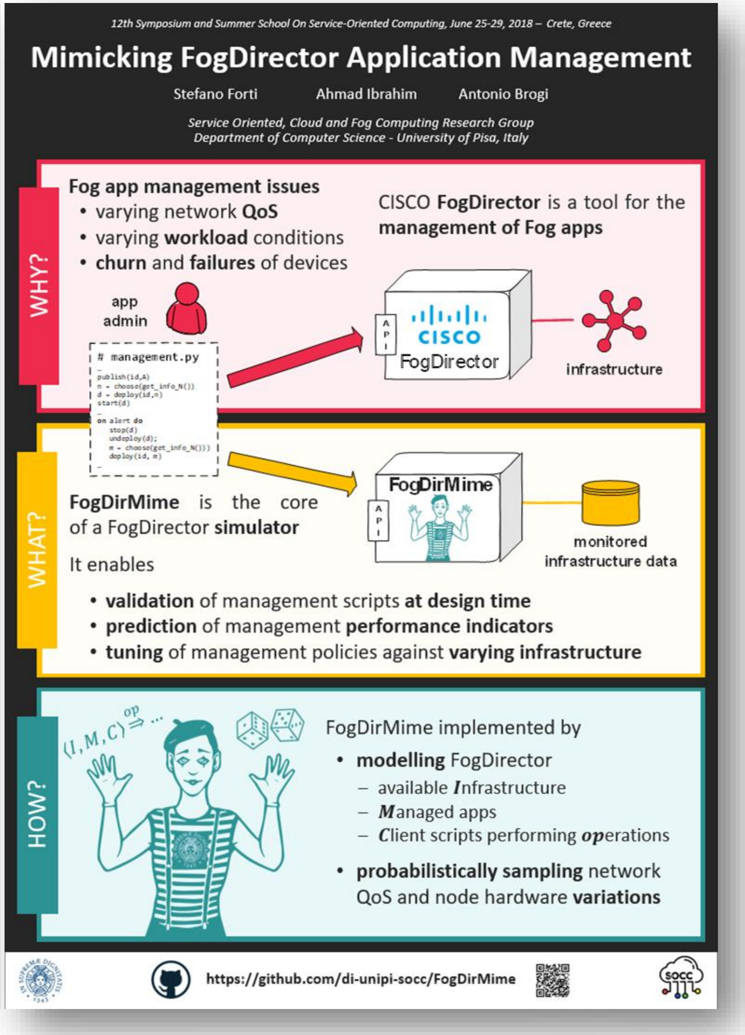
include **other functionalities** and **QoS-aware management**

implement a full-fledged **simulation environment** for FogDirector

consider **scaling** and **osmotic computing** for **multi-component applications**

study other recent tools such as **EdgeX Foundry**$^{\text{TM}}$

Q&A → Poster Session



12th Symposium and Summer School On Service-Oriented Computing, June 25-29, 2018 – Crete, Greece

Mimicking FogDirector Application Management

Stefano Forti Ahmad Ibrahim Antonio Brogi

Service Oriented, Cloud and Fog Computing Research Group
Department of Computer Science - University of Pisa, Italy

WHY?

Fog app management issues

- varying network **QoS**
- varying **workload** conditions
- **churn** and **failures** of devices

CISCO **FogDirector** is a tool for the **management of Fog apps**

```
# management.py
…
publish(id,A)
n - choose(get_info_N())
d - deploy(id,n)
start(d)
…
on alert do
    stop(d)
    undeploy(d);
    m - choose(get_info_N()))
    deploy(id, m)
…
```

WHAT?

FogDirMime is the core of a FogDirector **simulator**

It enables

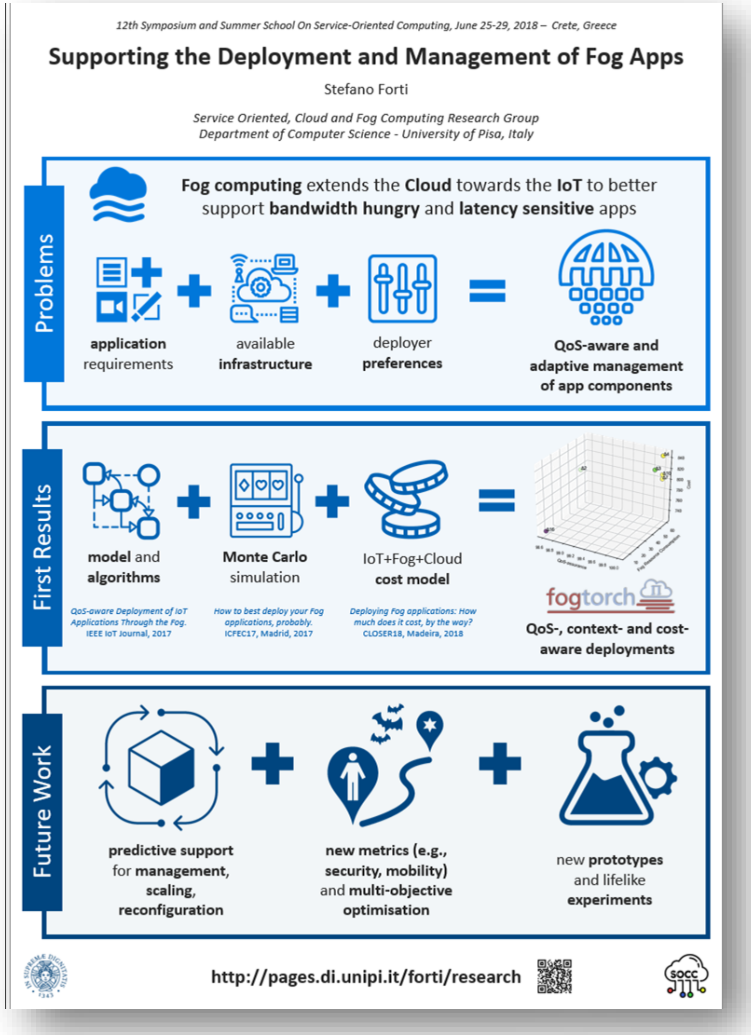
- **validation** of management scripts **at design time**
- **prediction** of **management performance indicators**
- **tuning** of management policies against **varying infrastructure**

HOW?

FogDirMime implemented by

- **modelling** FogDirector
  - available ***I***nfrastructure
  - ***M***anaged apps
  - ***C***lient scripts performing ***op***erations
- **probabilistically sampling** network QoS and node hardware **variations**

https://github.com/di-unipi-socc/FogDirMime



12th Symposium and Summer School On Service-Oriented Computing, June 25-29, 2018 – Crete, Greece

Supporting the Deployment and Management of Fog Apps

Stefano Forti

Service Oriented, Cloud and Fog Computing Research Group
Department of Computer Science - University of Pisa, Italy

Problems

Fog computing extends the **Cloud** towards the **IoT** to better support **bandwidth hungry** and **latency sensitive** apps

application requirements + available **infrastructure** + deployer **preferences** = **QoS-aware and adaptive management of app components**

First Results

model and **algorithms** + **Monte Carlo** simulation + IoT+Fog+Cloud **cost model** = fogtorch QoS-, context- and cost-aware deployments

QoS-aware Deployment of IoT Applications Through the Fog. IEEE IoT Journal, 2017

How to best deploy your Fog applications, probably. ICFEC17, Madrid, 2017

Deploying Fog applications: How much does it cost, by the way? CLOSER18, Madeira, 2018

Future Work

predictive support for **management, scaling, reconfiguration** + **new metrics** (e.g., **security, mobility**) and **multi-objective optimisation** + new **prototypes** and lifelike **experiments**

http://pages.di.unipi.it/forti/research

Mimicking FogDirector Application Management

Stefano Forti, Ahmad Ibrahim and Antonio Brogi

Service-oriented, Cloud and Fog Computing Research Group

Department of Computer Science

University of Pisa, Italy

12th Symposium and Summer School On Service-Oriented Computing, June 25 – June 29, 2018 in Crete, Greece