Distributed Systems

Engineering the Smart Fabric of IoT, People, and Systems

29 October 2019, Dagstuhl 19442

Schahram Dustdar

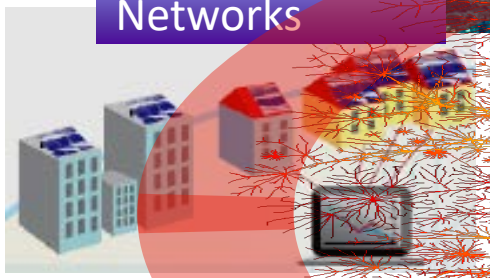
Distributed Systems Group
TU Wien

dsg.tuwien.ac.at

Smart Evolution – People, Services, and Things

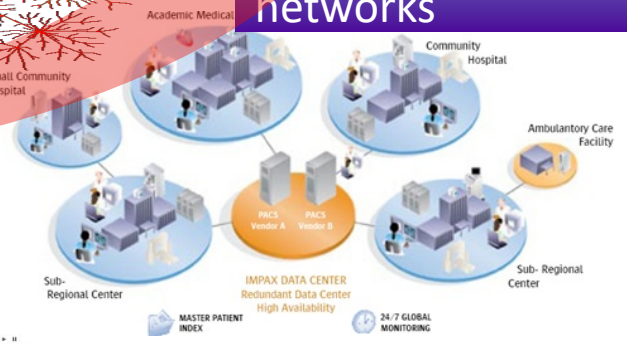
Smart Energy Networks

Smart eGovernments & eAdministrations



Elastic Systems & Processes

eHealth & Smart Health networks



Smart Homes

STB

Game Machine

TV

PC

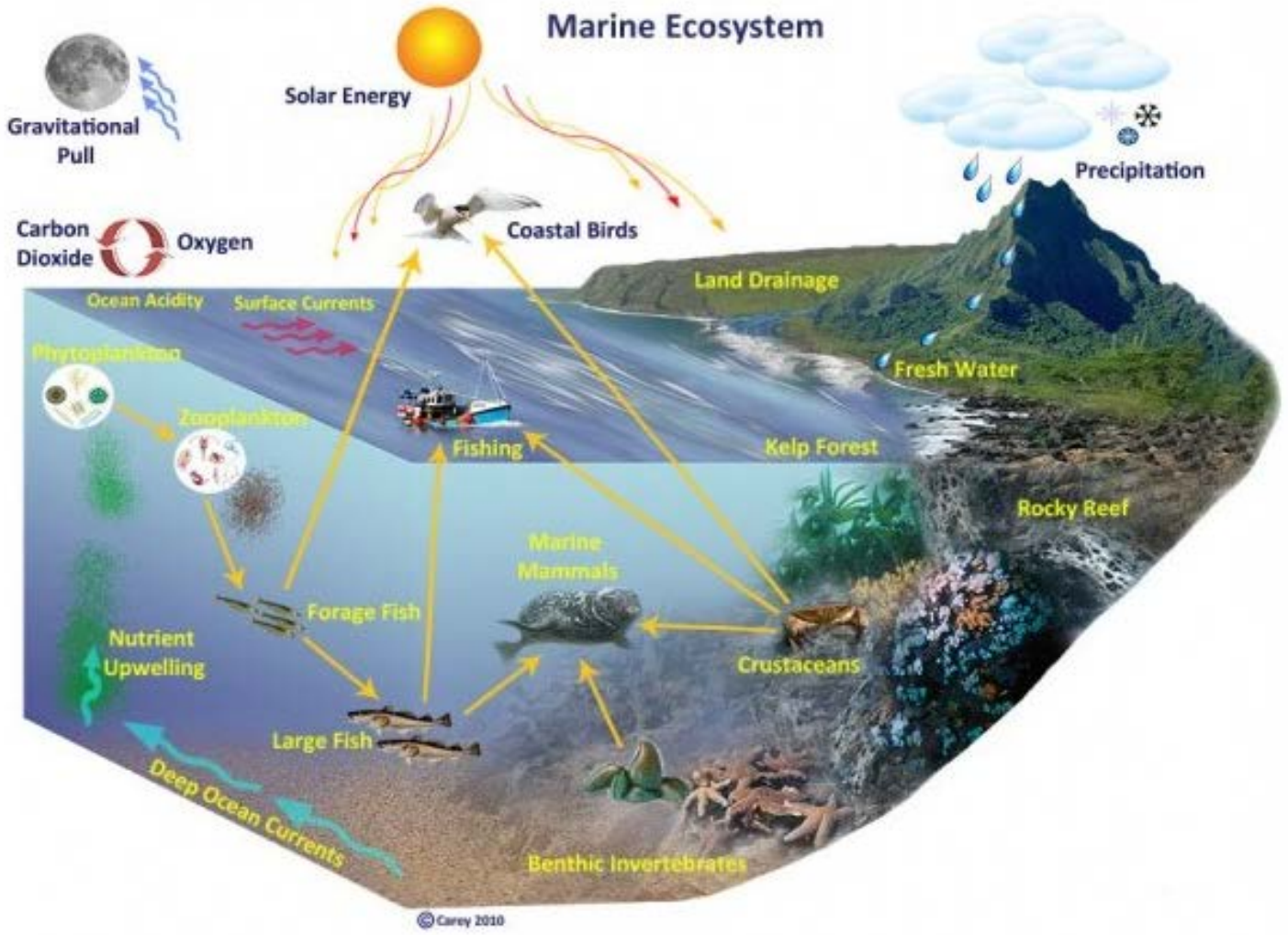
Audio

DVD

Telephone

Smart Transport Networks

Ecosystems: People, Systems, and Things

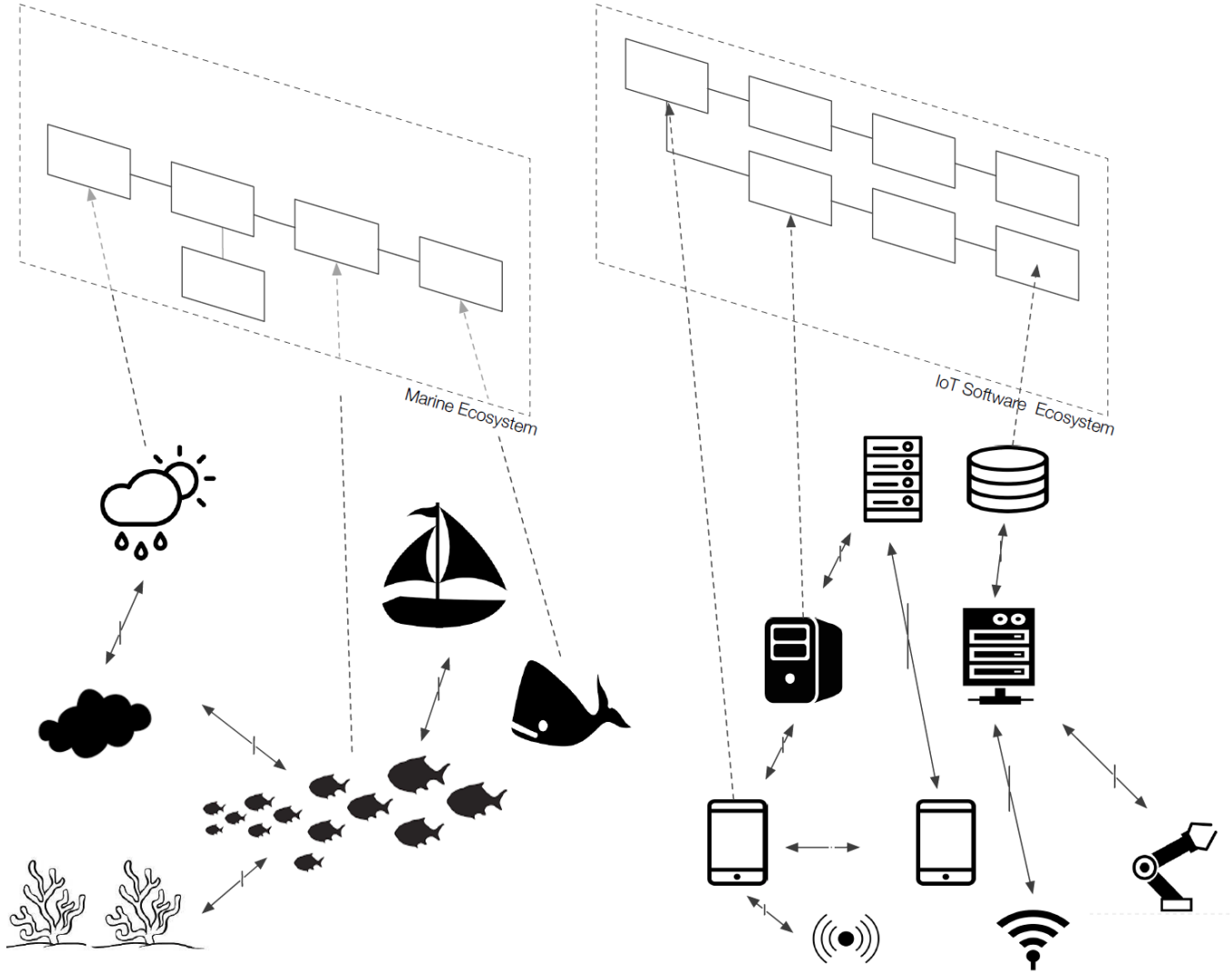


Marine Ecosystem: http://www.xbordercurrents.co.uk/wildlife/marine-ecosystem-2

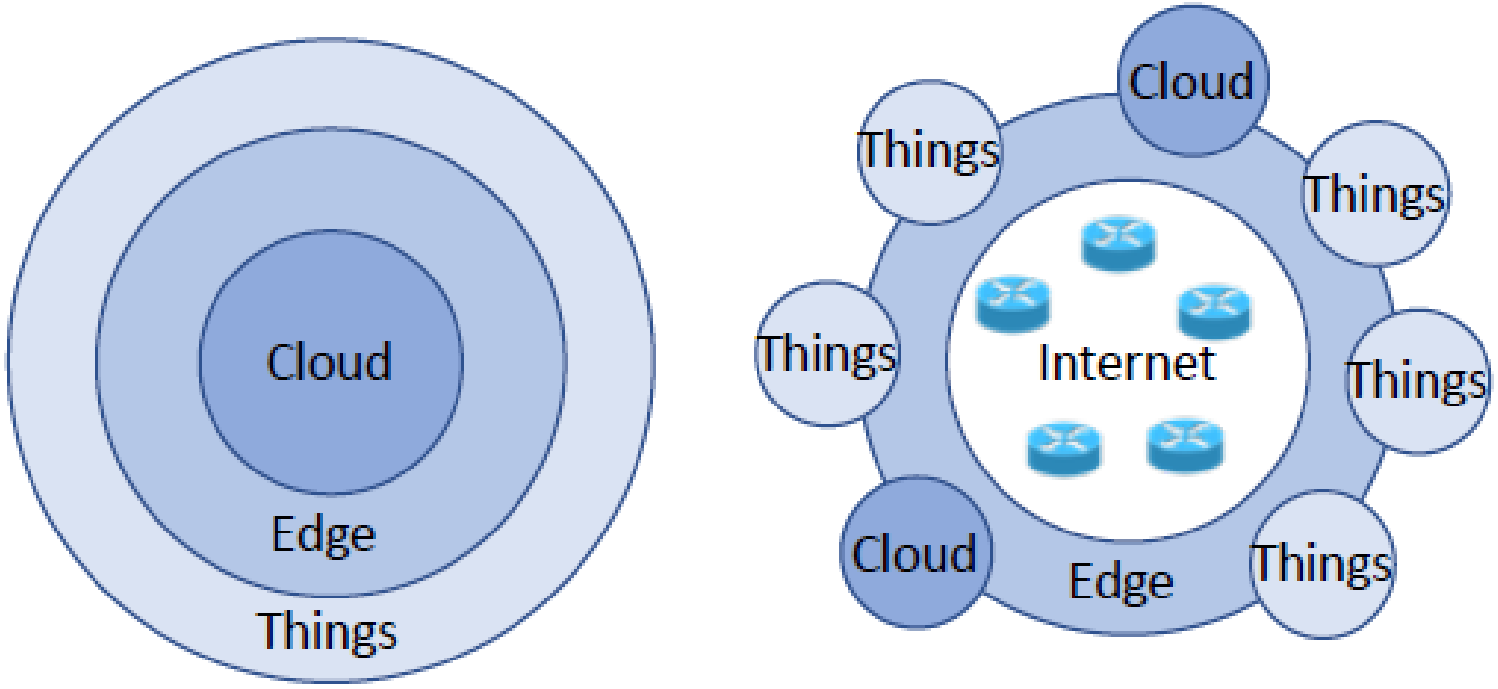
Complex system with networked dependencies and intrinsic adaptive behavior – has:

1. **Robustness & Resilience mechanisms**: achieving stability in the presence of disruption
2. **Measures of health**: diversity, population trends, other key indicators
3. **Built-in coherence**
4. **Entropy-resistence**

Ecosystems for IoT Systems



Perspectives on the IoT: Edge, Cloud, Internet



(a) A cloud-centric perspective: Edge as "edge of the cloud"

(b) An Internet-centric perspective: Edge as "edge of the Internet"

Kim, H., Lee, E.A., Dustdar, S. (2019). Creating a Resilient IoT With Edge Computing, *IEEE Computer, 52/8, August 2019*

Cloud-centric perspective

Assumptions

- Cloud provides core services; Edge provides local proxies for the Cloud (offloading parts of the cloud's workload)

Edge Computers

- play supportive role for the IoT services and applications
- Cloud computing-based IoT solutions use cloud servers for various purposes including massive computation, data storage, communication between IoT systems, and security/privacy

Missing

- In the network architecture, the cloud is also located at the network edge, not surrounded by the edge
- Computers at the edge do not always have to depend on the cloud; they can operate autonomously and collaborate with one another directly without the help of the cloud

Internet-centric perspective

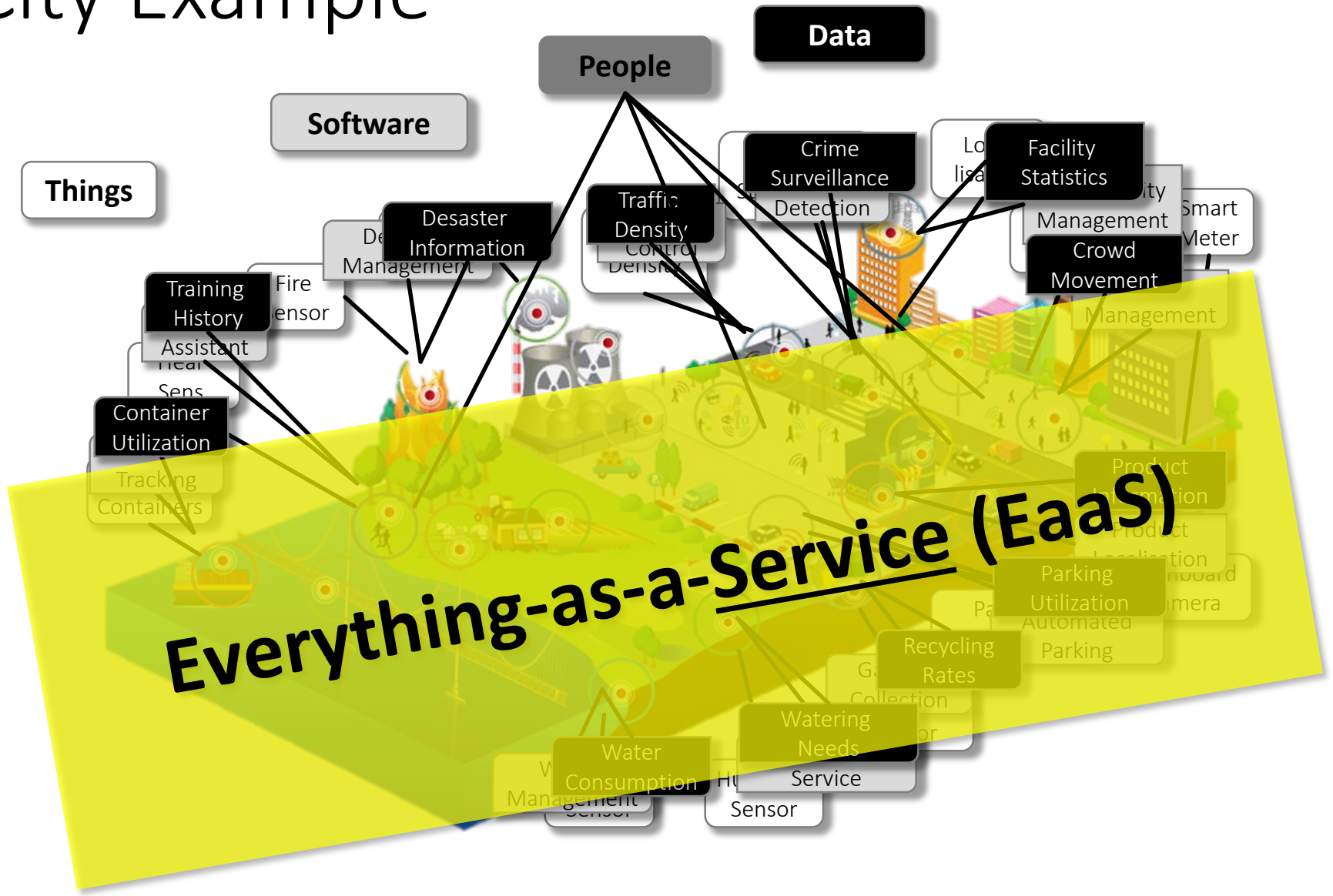
Assumptions

- Internet is center of IoT architecture; Edge devices are gateways to the Internet (not the Cloud)
- Each LAN can be organized around edge devices autonomously
- Local devices do not depend on Cloud

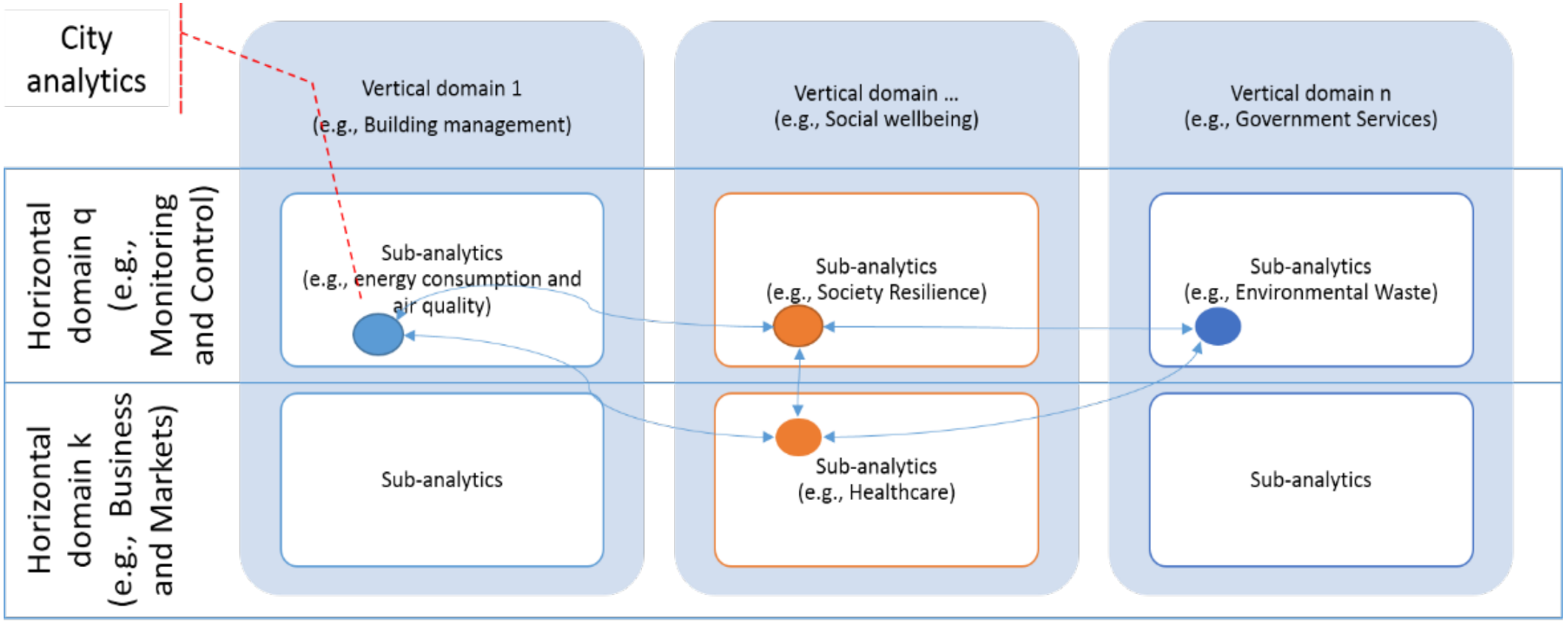
Therefore

- Things belong to partitioned subsystems and LANs rather than to a centralized system directly
- The Cloud is connected to the Internet via the edge of the network
- Remote IoT systems can be connected directly via the Internet. Communications does not have to go via the Cloud
- The Edge can connect things to the Internet and disconnect traffic outside the LAN to protect things -> IoT system must be able to act autonomously

Smart City Example



Dynamic Analytics (e.g., Smart City)

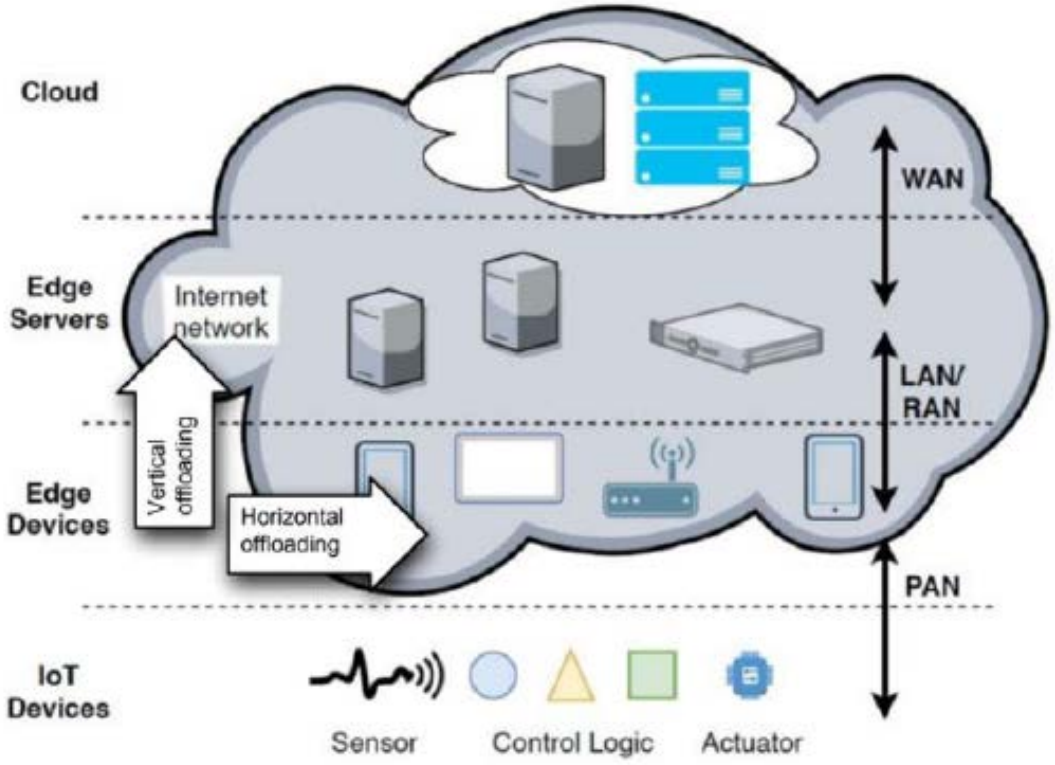


Vertical vs. Horizontal Edge Architecture

Cloud Computing

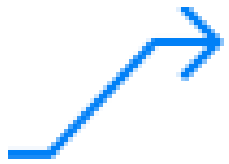
Fog Computing

Edge Computing



Paradigm 1: Elasticity (Resilience)

(Physics) The property of returning to an initial form or state following deformation



stretch when a force stresses them

e.g., ***acquire*** *new resources,* ***reduce*** *quality*

shrink when the stress is removed

e.g., ***release*** *resources,* ***increase*** *quality*

Elastic Computing > Scalability

Resource elasticity

Software / human-based computing elements, multiple clouds

Quality elasticity

Non-functional parameters e.g., performance, quality of data, service availability, human trust

Costs & Benefit elasticity

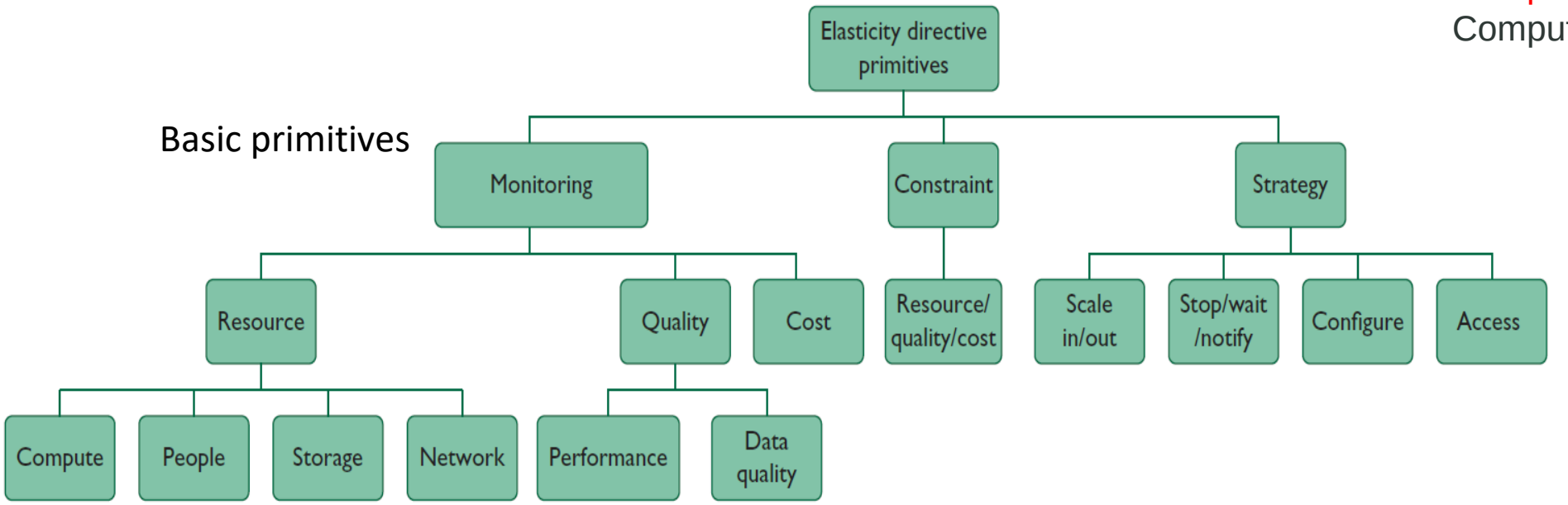
rewards, incentives

Dustdar S., Guo Y., Satzger B., Truong H. (2012) Principles of Elastic Processes, IEEE Internet Computing, Volume: 16, Issue: 6, Nov.-Dec. 2012

Specifying and controling elasticity

Dustdar, S. et al.: Programming Directives for Elastic Computing. IEEE Internet Computing 16(6): 72-77 (2012)

Basic primitives



SYBL (Simple Yet Beautiful Language) for specifying elasticity requirements

SYBL-supported requirement levels

- Cloud Service Level
- Service Topology Level
- Service Unit Level
- Relationship Level
- Programming/Code Level

Current SYBL implementation

in Java using Java annotations

```
@SYBLAnnotation(monitoring=„“,constraints=„“,strategies=„“)
```

in XML

```
<ProgrammingDirective><Constraints><Constraint
    name=c1>...</Constraint></Constraints>...</ProgrammingDirective
    >
```

as TOSCA Policies

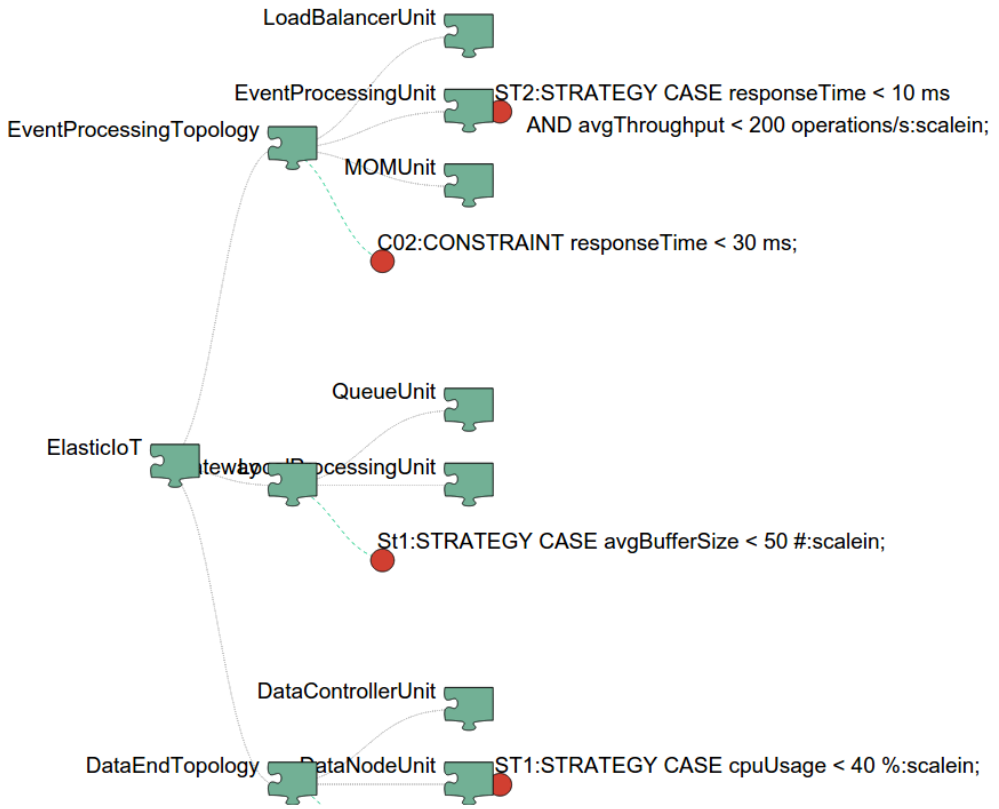
```
<tosca:ServiceTemplate name="PilotCloudService"> <tosca:Policy
    name="St1" policyType="SYBLStrategy">   St1:STRATEGY
    minimize(Cost) WHEN high(overallQuality)  </tosca:Policy>...
```

High level elasticity control

#SYBL.CloudServiceLevel
Cons1: CONSTRAINT responseTime < 5 ms
Cons2: CONSTRAINT responseTime < 10 ms
WHEN nbOfUsers > 10000
Str1: STRATEGY CASE fulfilled(Cons1) OR fulfilled(Cons2): minimize(cost)

#SYBL.ServiceUnitLevel
Str2: STRATEGY CASE ioCost < 3 Euro : maximize(dataFreshness)

#SYBL.CodeRegionLevel
Cons4: CONSTRAINT dataAccuracy>90% AND cost<4 Euro



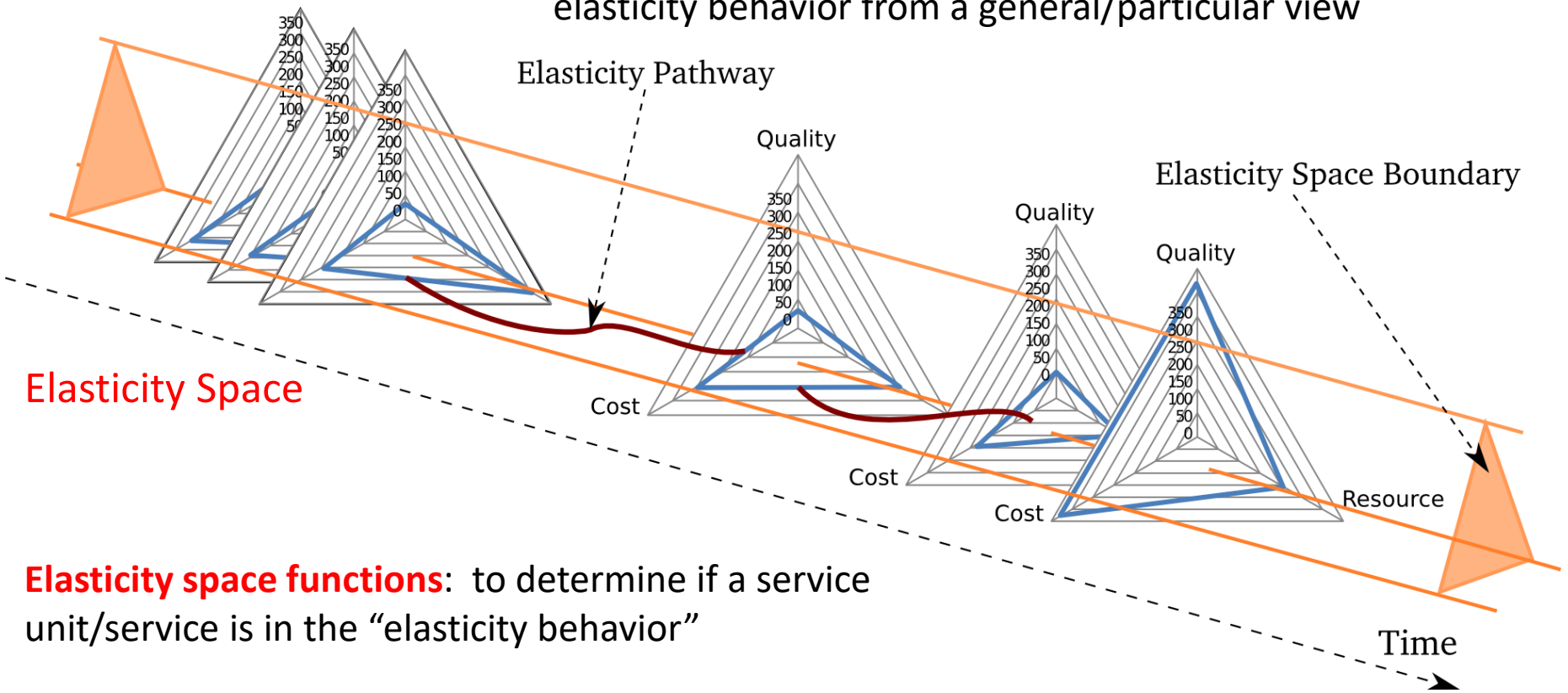
Georgiana Copil, Daniel Moldovan, Hong-Linh Truong, Schahram Dustdar, **"SYBL: an Extensible Language for Controlling Elasticity in Cloud Applications"**, 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), May 14-16, 2013, Delft, Netherlands

Copil G., Moldovan D., Truong H.-L., Dustdar S. (2016). **rSYBL: a Framework for Specifying and Controlling Cloud Services Elasticity**. *ACM Transactions on Internet Technology*

Elasticity Model for Cloud Services

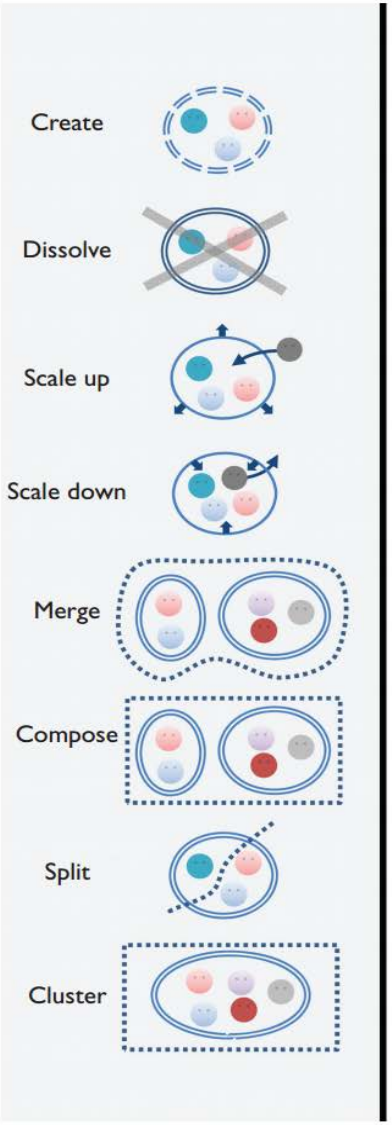
Moldovan D., G. Copil,Truong H.-L., Dustdar S. (2013). **MELA: Monitoring and Analyzing Elasticity of Cloud Service. CloudCom 2013**

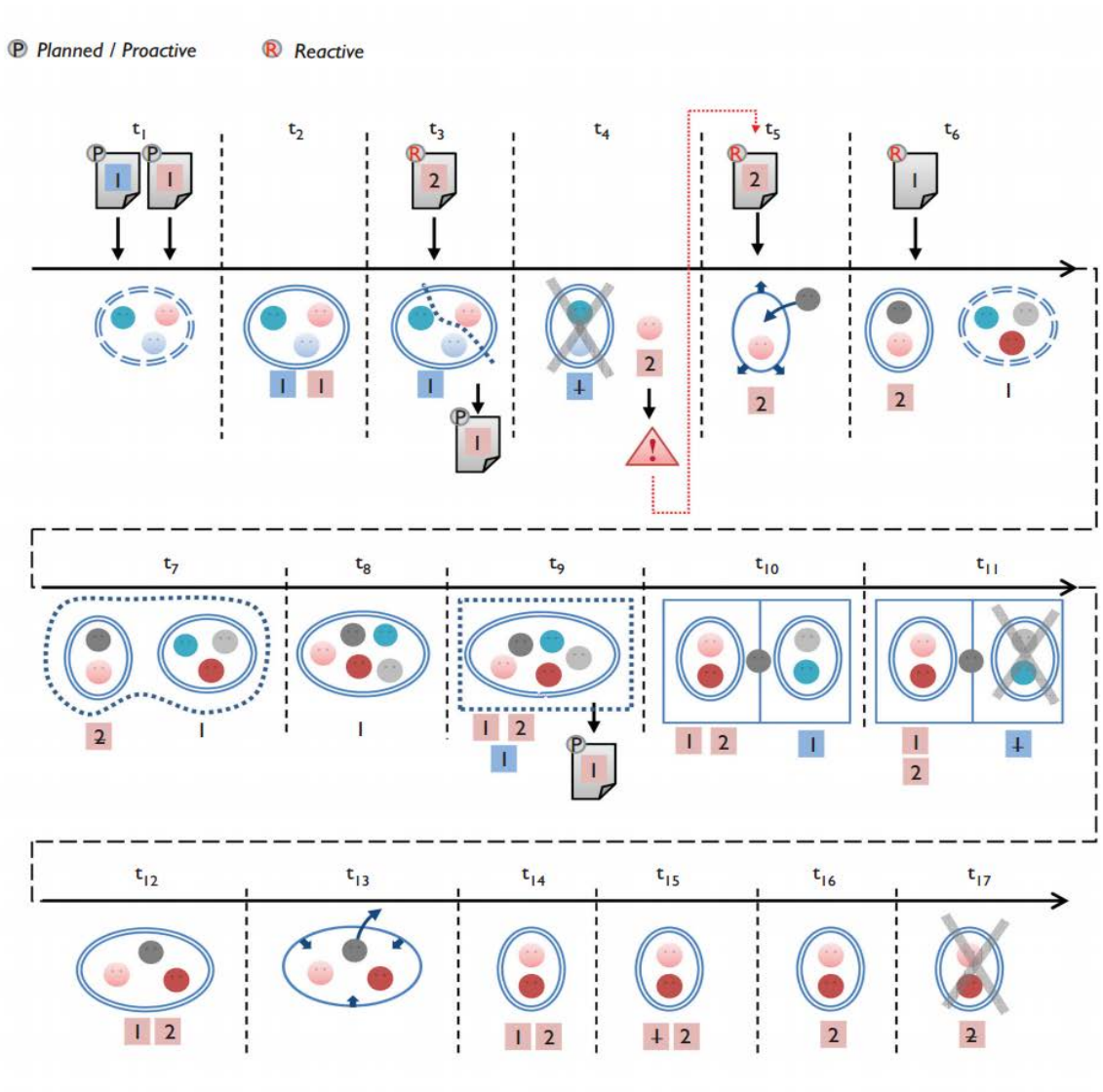
Elasticity Pathway functions: to characterize the elasticity behavior from a general/particular view



Elasticity space functions: to determine if a service unit/service is in the "elasticity behavior"

Paradigm 2: Social Compute Units (SCUs)

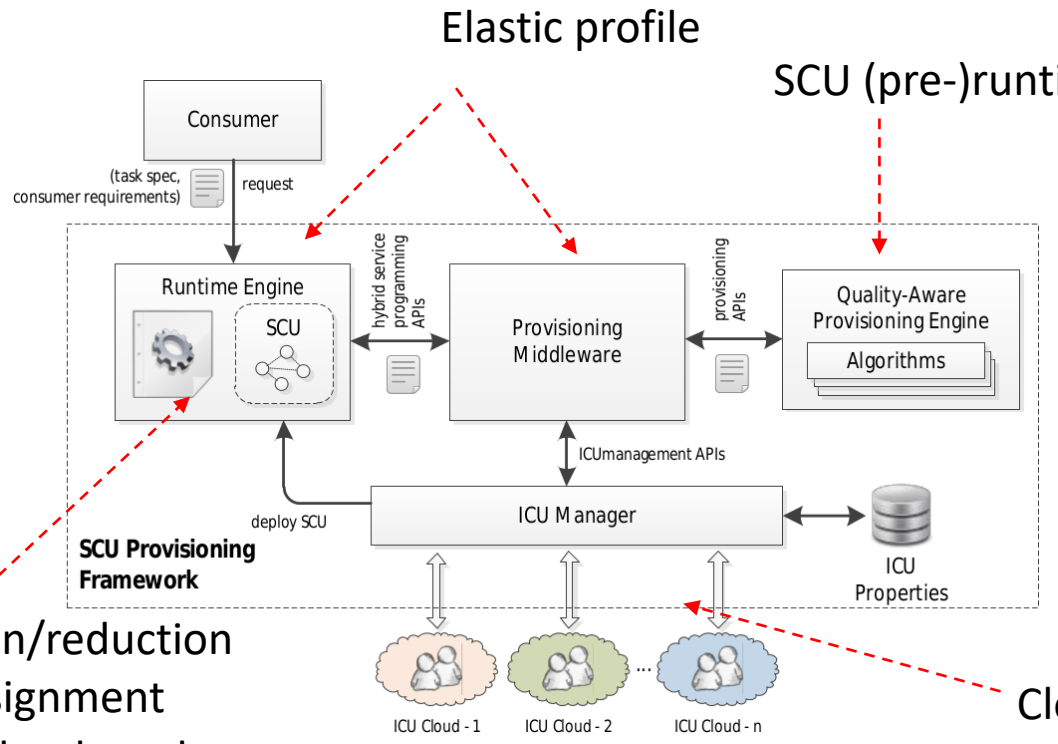




Dustdar S., Bhattacharya K. (2011). The Social Compute Unit, *IEEE Internet Computing,* Volume 15, Issue 3; pp. 64 - 69.

Fernández P., Truong H.-L., Dustdar S., Ruiz-Cortés A. (2015). Programming Elasticity and Commitment in Dynamic Processes. IEEE Internet Computing, Volume 19, Number 2, pp. 68 - 74

Elastic SCU provisioning (Paradigms 1 and 2 together)



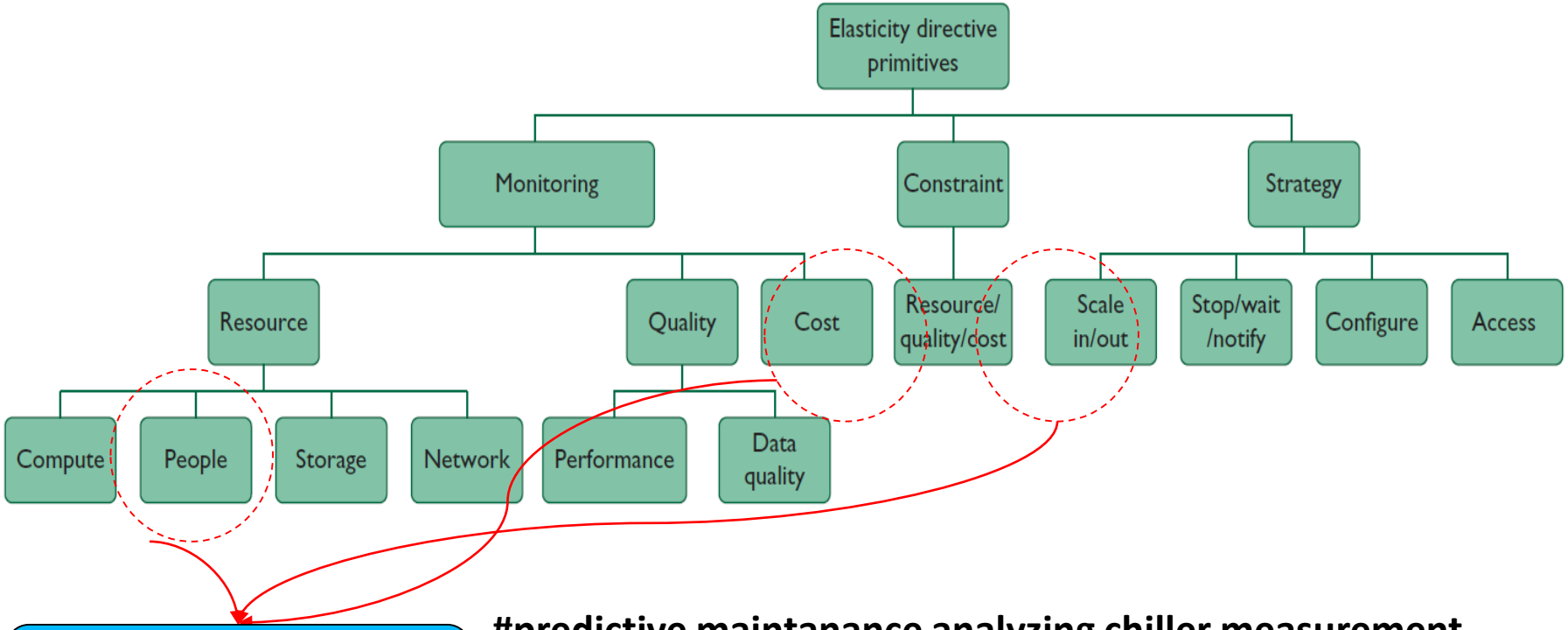
Algorithms

- Ant Colony Optimization variants
- FCFS
- Greedy

Mirela Riveni, Hong-Linh Truong, and Schahram Dustdar, **On the Elasticity of Social Compute Units, CAISE 2014**

Muhammad Z.C. Candra, Hong-Linh Truong, and Schahram Dustdar, **Provisioning Quality-aware Social Compute Units in the Cloud, ICSOC 2013.**

Specifying and controling elasticity of human-based services

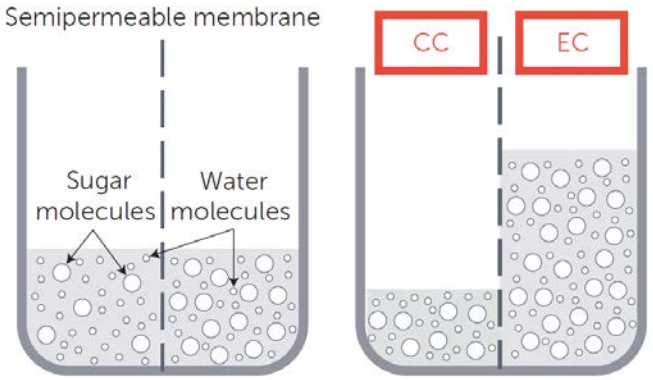


What if we need to "invoke" humans?

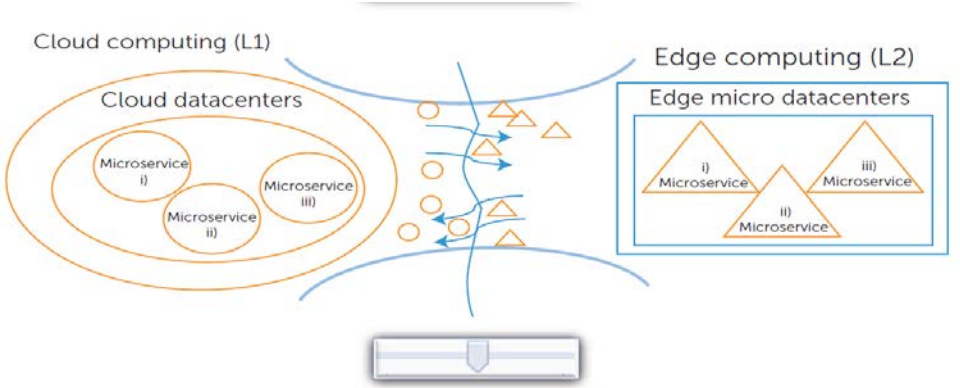
#predictive maintanance analyzing chiller measurement
#SYBL.ServiceUnitLevel
Mon1 MONITORING accuracy = Quality.Accuracy
Cons1 CONSTRAINT accuracy < 0.7
Str1 STRATEGY CASE Violated(Cons1):
Notify(Incident.DEFAULT, ServiceUnitType.HBS)

Paradigm 3: Osmotic Computing

- In chemistry, “osmosis” represents the seamless diffusion of molecules from a higher to a lower concentration solution.



- Dynamic management of (micro)services across cloud and edge datacenters
  - deployment, networking, and security, …
  - providing reliable IoT support with specified levels of QoS.



Villari M., Fazio M., Dustdar S., Rana O., Ranjan R. (2016). Osmotic Computing: A New Paradigm for Edge/Cloud Integration. *IEEE Cloud Computing*, Volume 3, Issue 6, pp. 76-83

IoT Computational Units (abstractions)

1. **MicroServices** (MS), which implement specific functionalities and can be deployed and migrated across different virtualized and/or containerized infrastructures (e.g., Docker) available across Cloud, Edge, and Things layers

2. **MicroData** (MD), encodes the contextual information about (a) the sensors, actuators, edge devices, and cloud resources it needs to collect data from or send data to, (b) the specific type of data (e.g., temperature, vibration, pollution, pH, humidity) it needs to process, and (c) other data manipulation operations such as where to store data, where to forward data, and where to store results

3. **MicroComputing** (MC), executing specific types of computational tasks (machine learning, aggregation, statistical analysis, error checking, and format translation) based on a mix of historic and real-time MD data in heterogeneous formats. These MCs could be realized using a variety of data storage and analytics programming models (SQL, NoSQL, stream processing, batch processing, etc.)

4. **MicroActuator** (MA), implementing programming interfaces (e.g., for sending commands) with actuator devices for changing or controlling object states in the IoT environment

Villari M., Fazio M., Dustdar S., Rana O., Jha D. N., Ranjan R. (2019). Osmosis: The Osmotic Computing Platform for Microelements in the Cloud, Edge, and Internet of Things. *IEEE Computer*, Volume 52, Issue 8, pp. 14-26

Thanks for your attention

Schahram Dustdar

Distributed Systems Group
TU Wien

dsg.tuwien.ac.at