Variability Variations in Cyber-Physical Systems

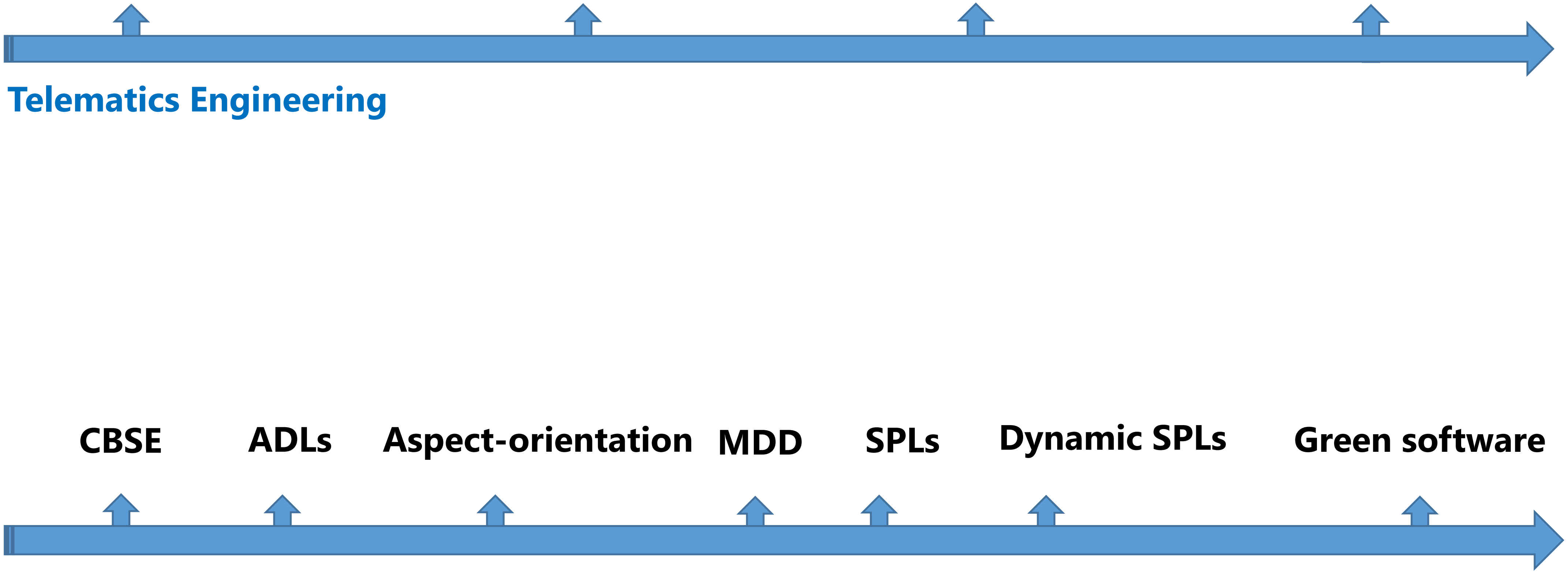
Lidia Fuentes (http://www.lcc.uma.es/~lff/)

Universidad de Málaga (Spain)

SPLC/ECSA 2019 Paris

My academic story

Teaching



Multimedia applications

Ambient Intelligence

Internet of Things

Cyber-physical systems

Telematics Engineering

Research

CBSE

ADLs

Aspect-orientation

MDD

SPLs

Dynamic SPLs

Green software

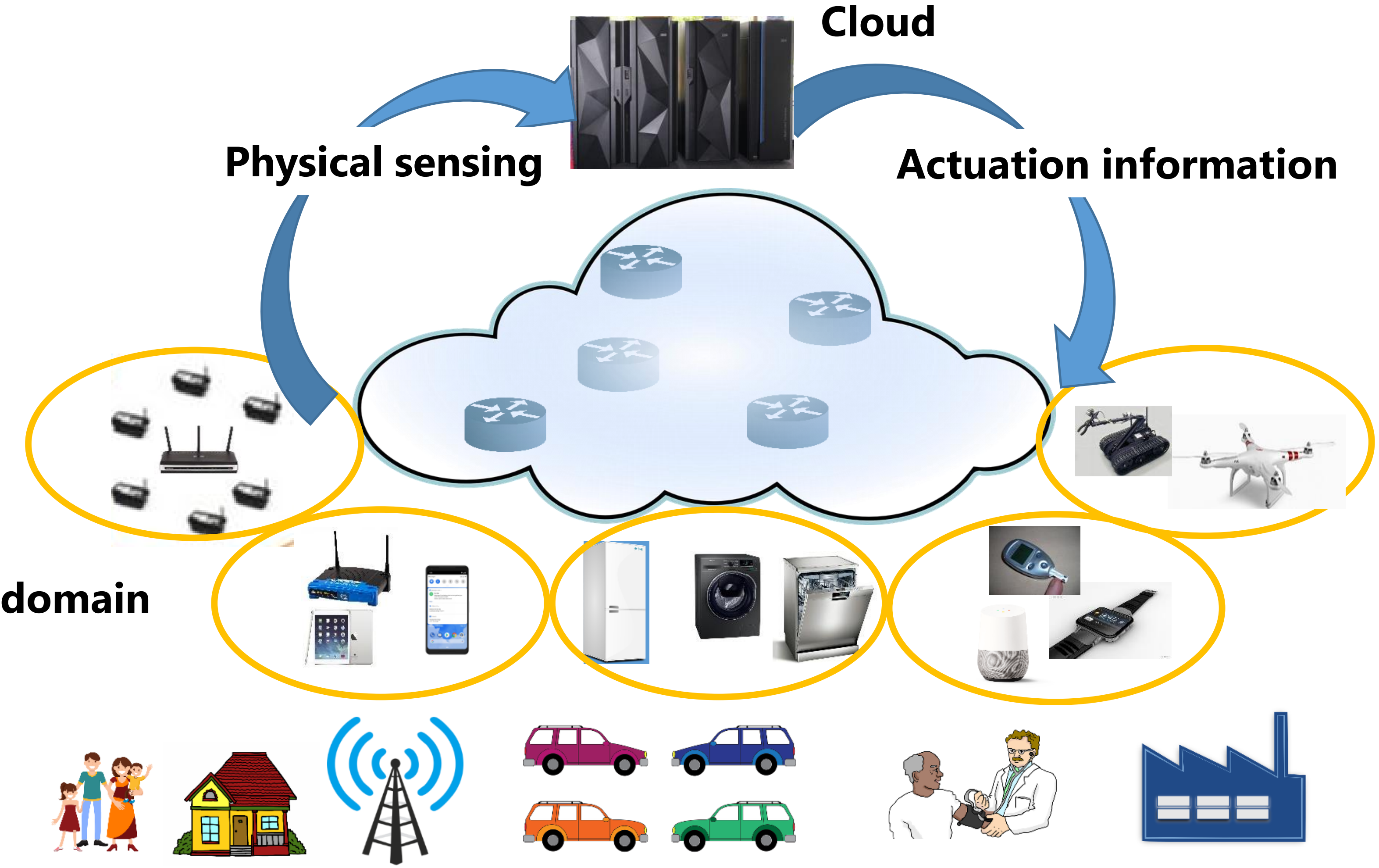
MsC, PhD in Computer Science

25 years later

What are cyber-physical systems ?

CPSs integrate computation, networking, and physical processes

- Cyber physical systems, **control the physical world**
- Physical devices interact thanks to **Internet of Thing (IoT)**
- CPSs **interact with their environment** thanks to sensors and actuators
- **CPSoS** are CPSs



Why are Cyber-Physical Systems important?

“Software is eating the world”

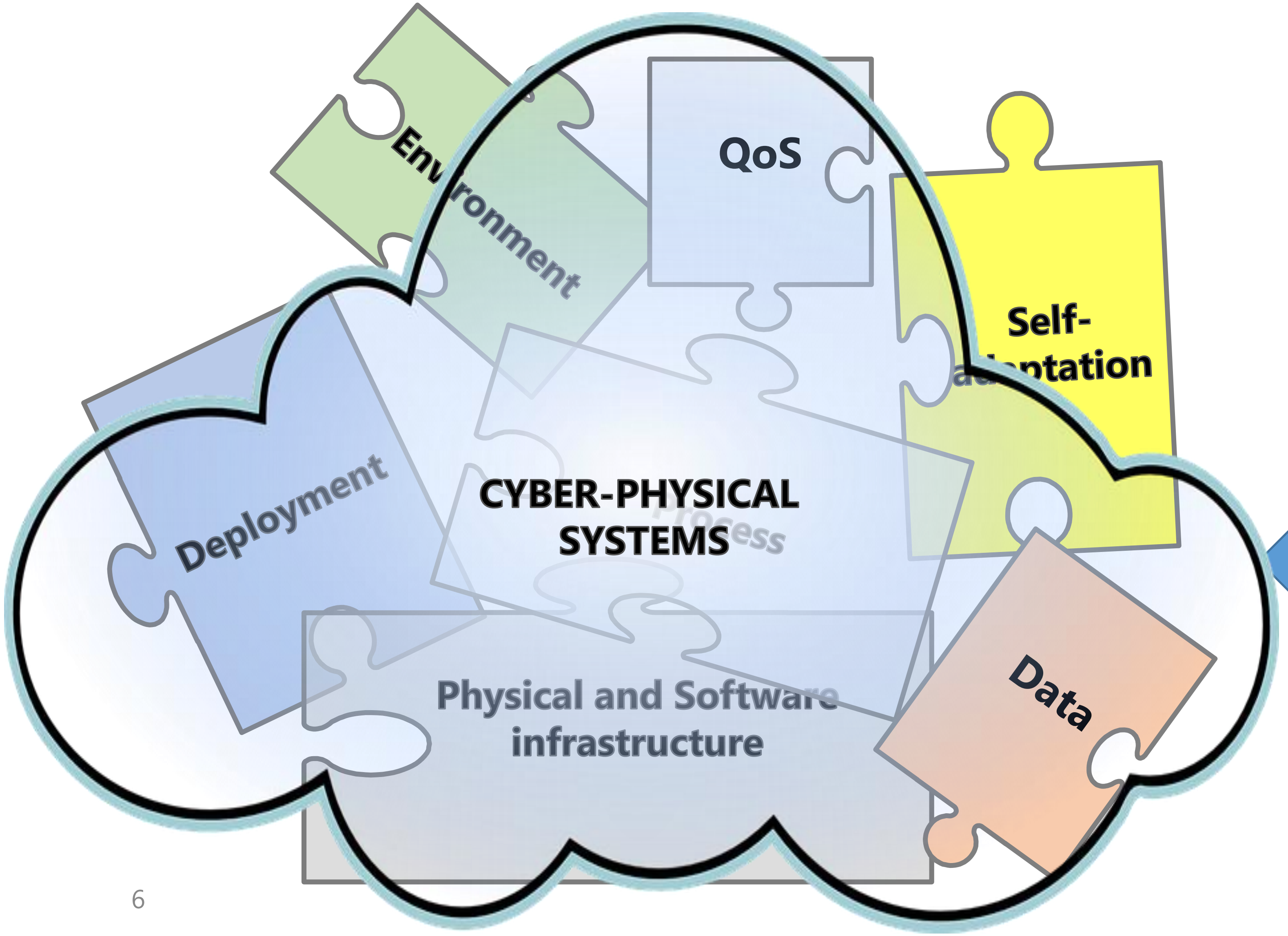
M. Andreessen

- Cyber physical systems play **an important role in future society** with an enormous **economic importance**
- CPSs **support many application domains**
  - Improve health care, address climate change, support renewable energy, autonomous driving cars, ageing population, sustainability, among others
- CPSs are present in the **Industry 4.0** providing new production methodologies

What are the challenges of CPSs?

- Software is everywhere embedded in **heterogeneous IoT devices**
- Applications are part of **CPSs** and are disperse running in the **cloud or edge**
- **Industry 4.0** describes the trend towards automation and data exchange in manufacturing **diverse technologies and processes**
- Customers demand high quality **customized services**
- Systems should cope with **unplanned and often unforeseen situations**, the known un-knowns

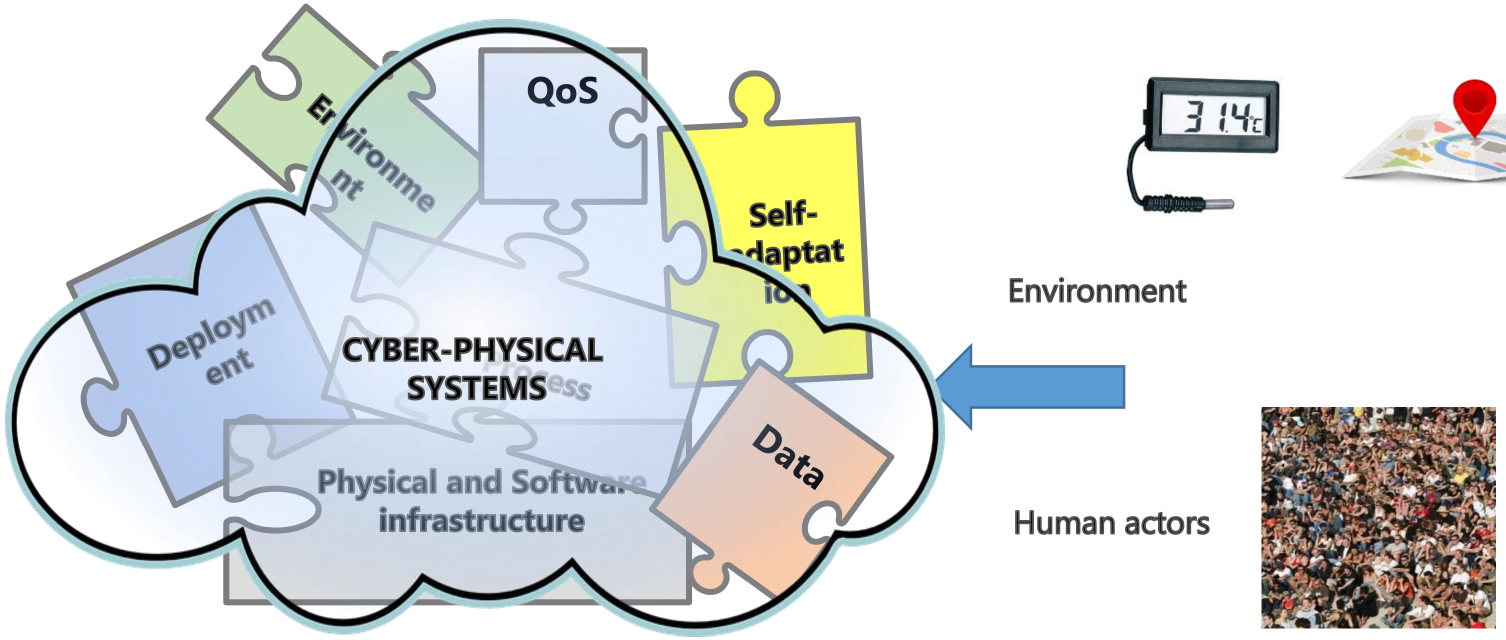
CPS can be a complex puzzle



Environment

Human actors

Complexity can be faced with variability modeling?



$$CPS\ Complexity \xRightarrow[tackled]{} Variability\ modeling$$

Variability modeling helps to deal with CPSs Complexity

Variability dimensions of CPS

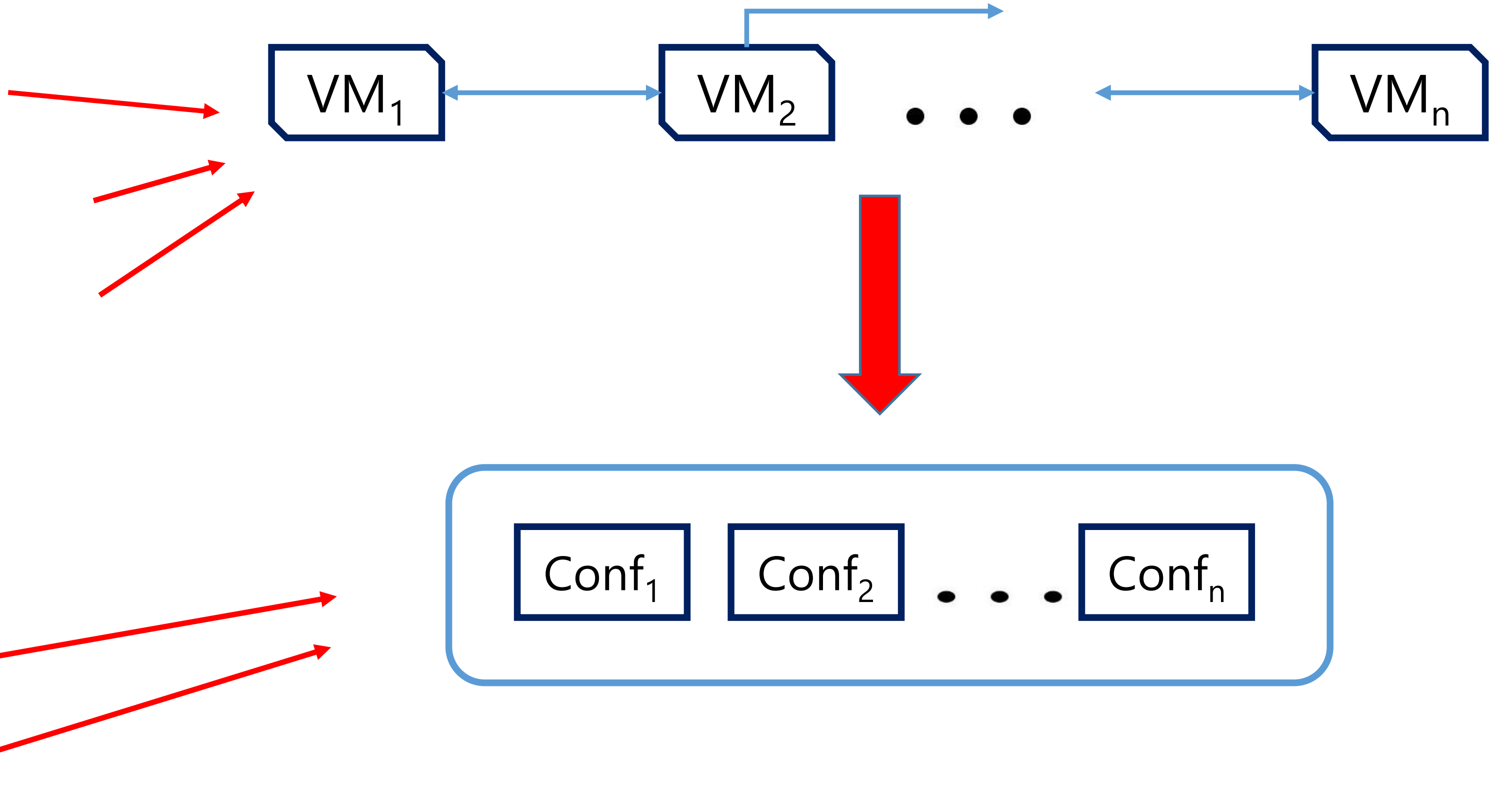
Variabilities in CPS

- Infrastructure variability
  - Physical infrastructure variability
  - Software infrastructure variability
- Data model variability
- Process variability
- Quality attributes variability
- Deployment variability
- Environment variability
- Runtime variability (self-adaptation)

SPL engineers are not experts in all CPSs technologies

CPS variability modeling

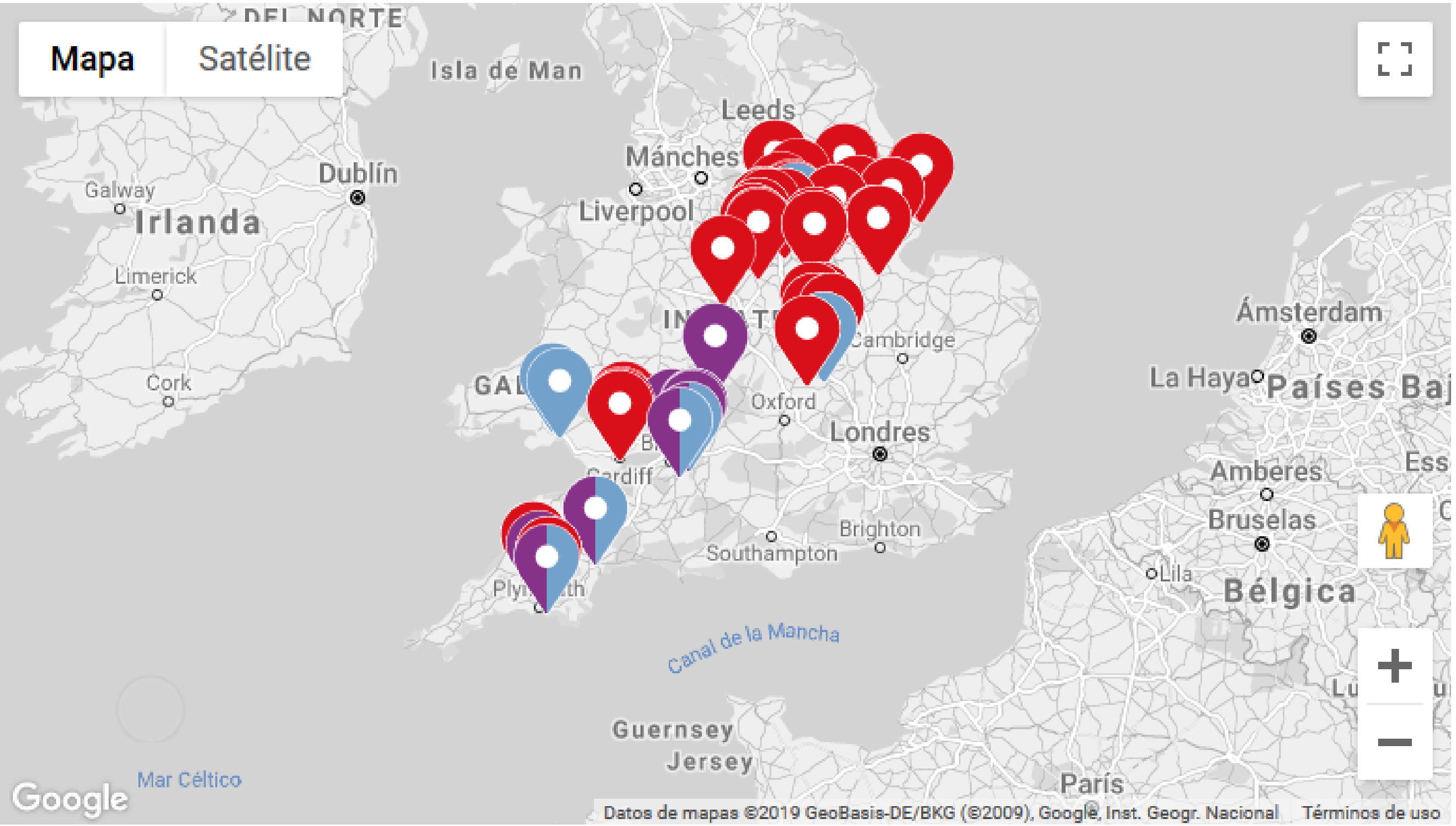
- **Multi-product line activities**
  - Mapping of variability models
  - Synchronization of variabilty models
  - Propagation of changes, consistency
  - Reduce scalability problems
- **Configuration of CPS**
  - Quality of configurations
  - CPS requirements



Automation of power distribution

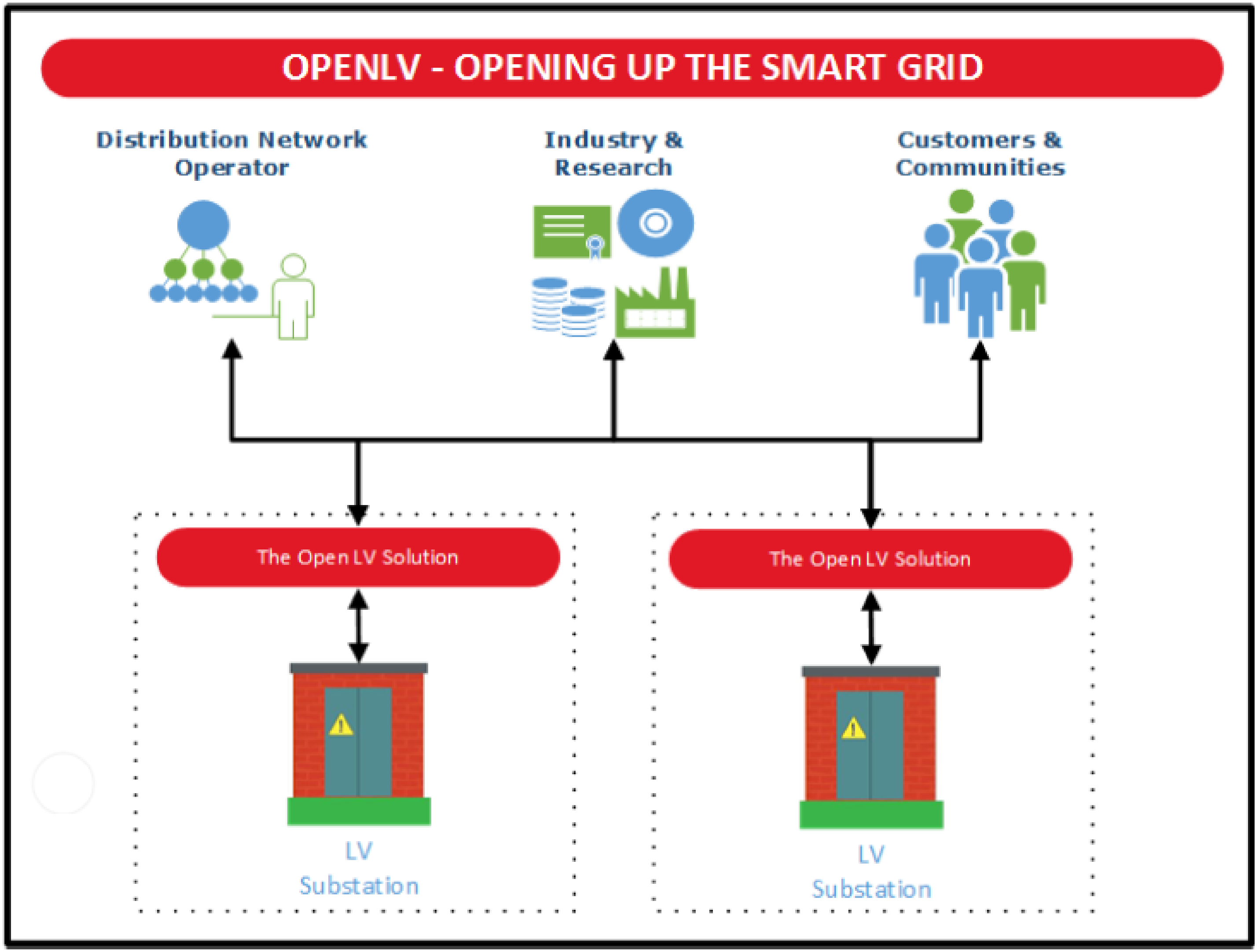
WESTERN POWER DISTRIBUTION

- Great Britain Low Voltage (LV) networks are expected to see radical change (electrical vehicles, solar panels, etc.)
- **Goal**: provision of LV network data to wider industry and research
- **Open-LV** project



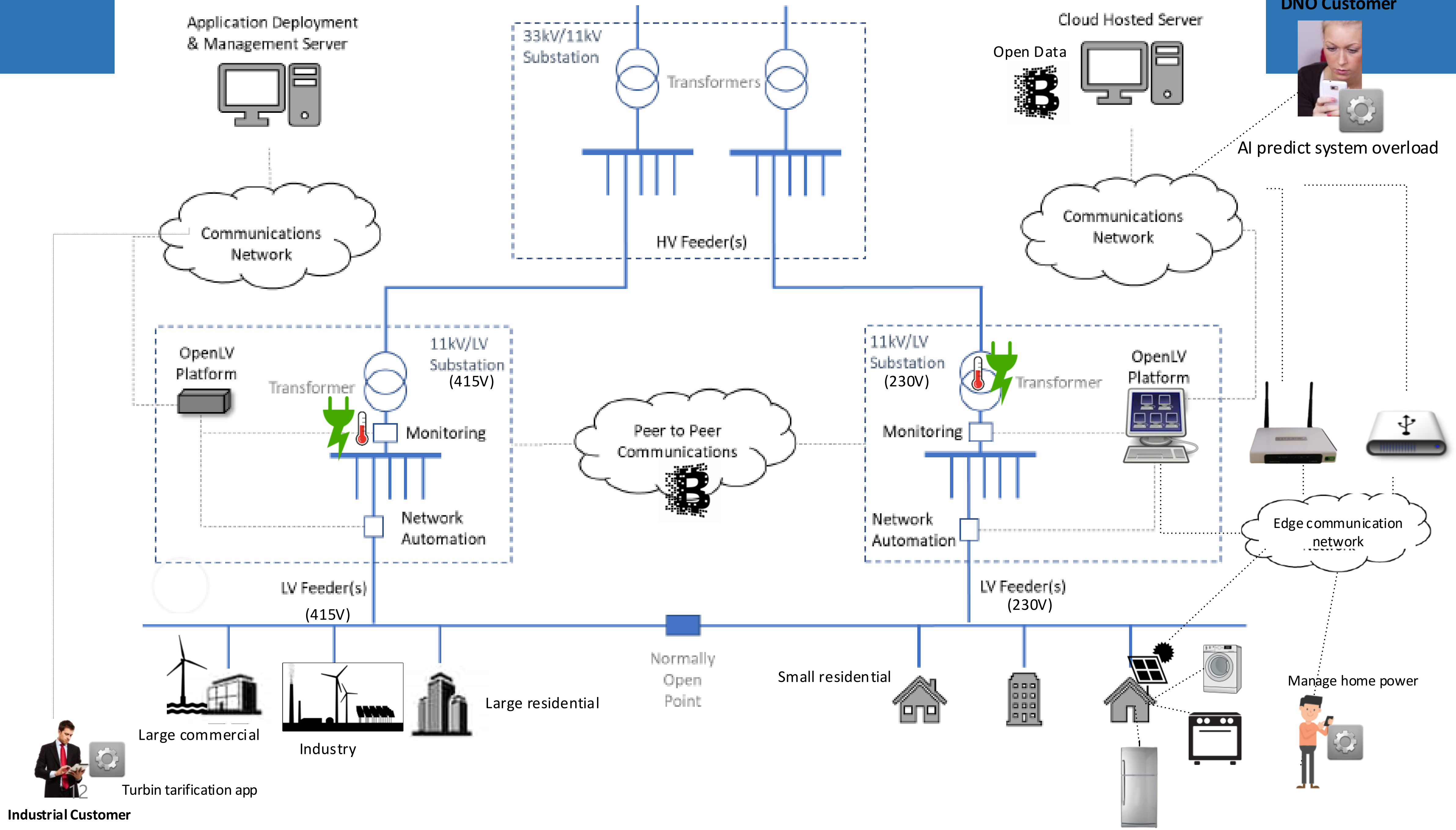
Open-LV project

- Open software platform in electricity substations that can **monitor substation electricity demand.**
- The **LV-CAP™ platform** integrates third party products to enable network control and more participation in network management.



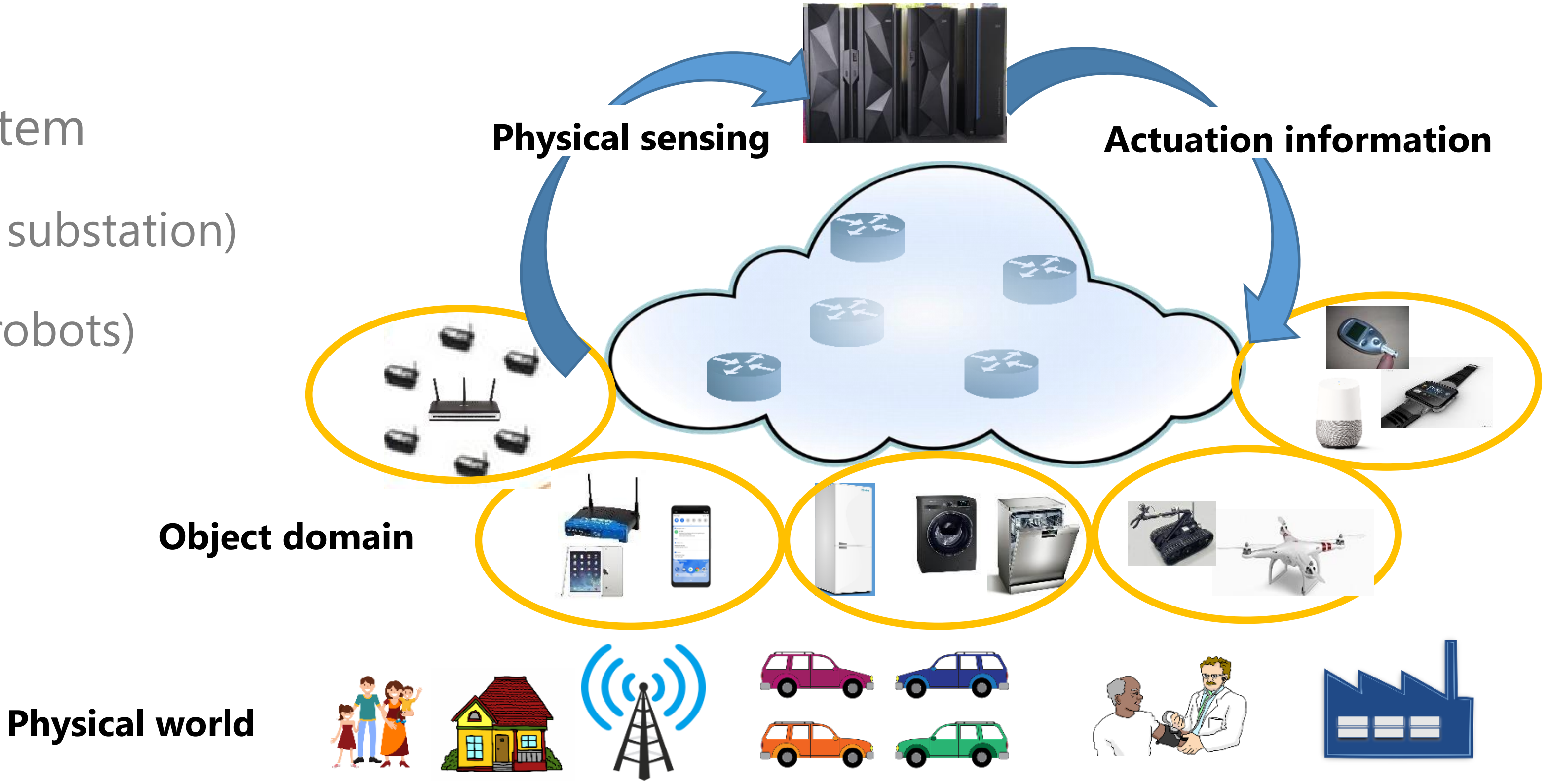
Distribution Network Operator

WESTERN POWER DISTRIBUTION



Infrastructure variability

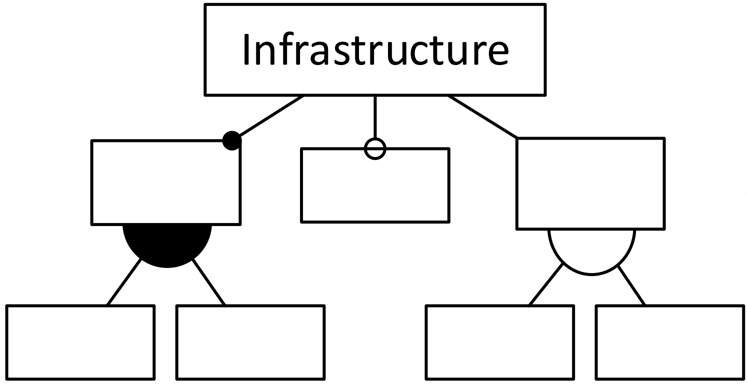
- **Physical infrastructure**
  - Physical structure of the system
    - Industry 4.0 (e.g. electrical substation)
    - Robotics (e.g. Agriculture robots)
  - Network connectivity
  - Data terminal equipments
- **Software infrastructure**
  - Operating system
  - Platforms
  - Virtualization



Physical infrastructure variability

Physical infrastructure

Models world



deployment

latency

connectivity constraints

energy consumption

position

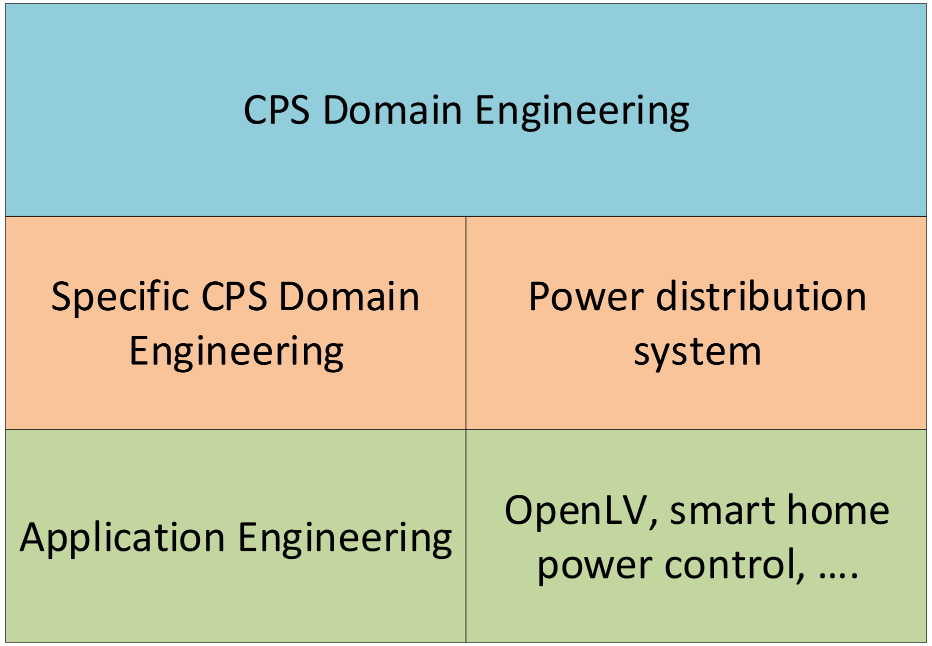
Physical world

Technical sheet

- HD, RAM, CPU
- Network cards/interfaces
- Operating system
- Connectivity
- Positioning
- Sensing units
- .....

Physical infrastructure

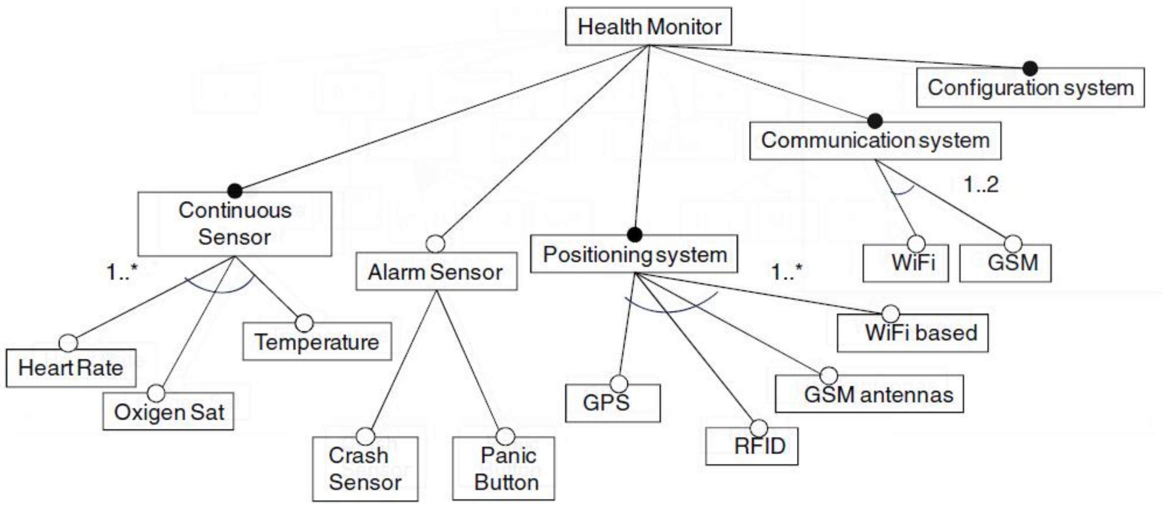
- Physical infrastructure is shared
- Physical aware domain engineering
- Configuration layer by layer



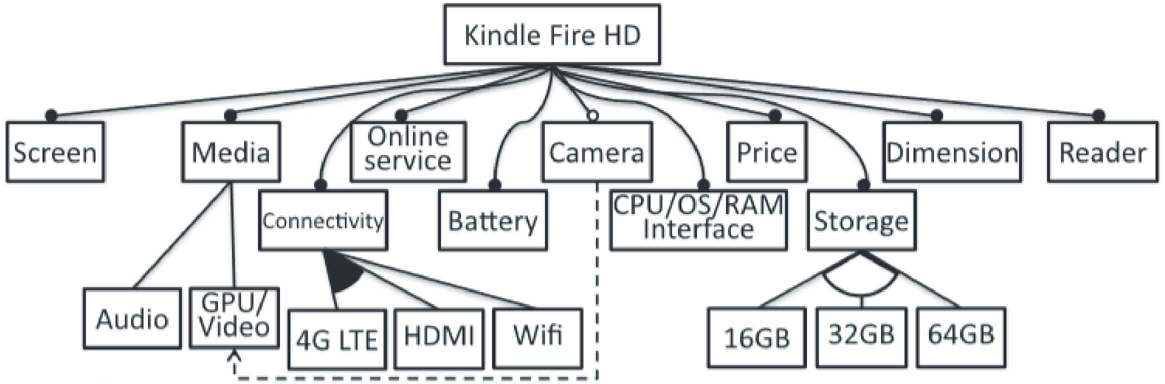
CPS Domain Engineering	
Specific CPS Domain Engineering	Power distribution system
Application Engineering	OpenLV, smart home power control, ….

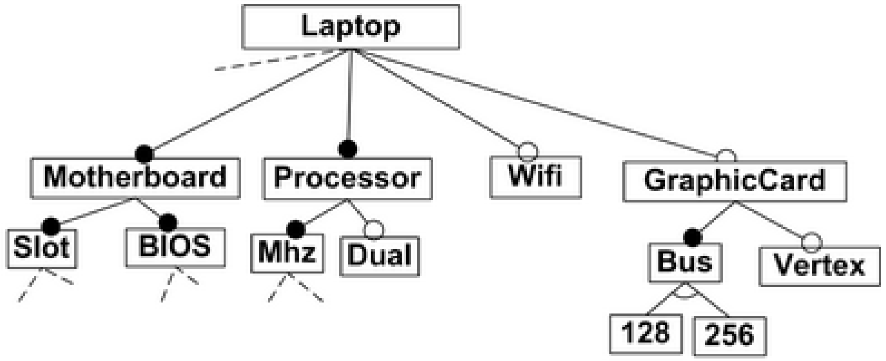
How have we modeled physical variability?

- WiFi modeled with **three different meanings**
- No semantic information



[Laguna et al. IWANN'09]



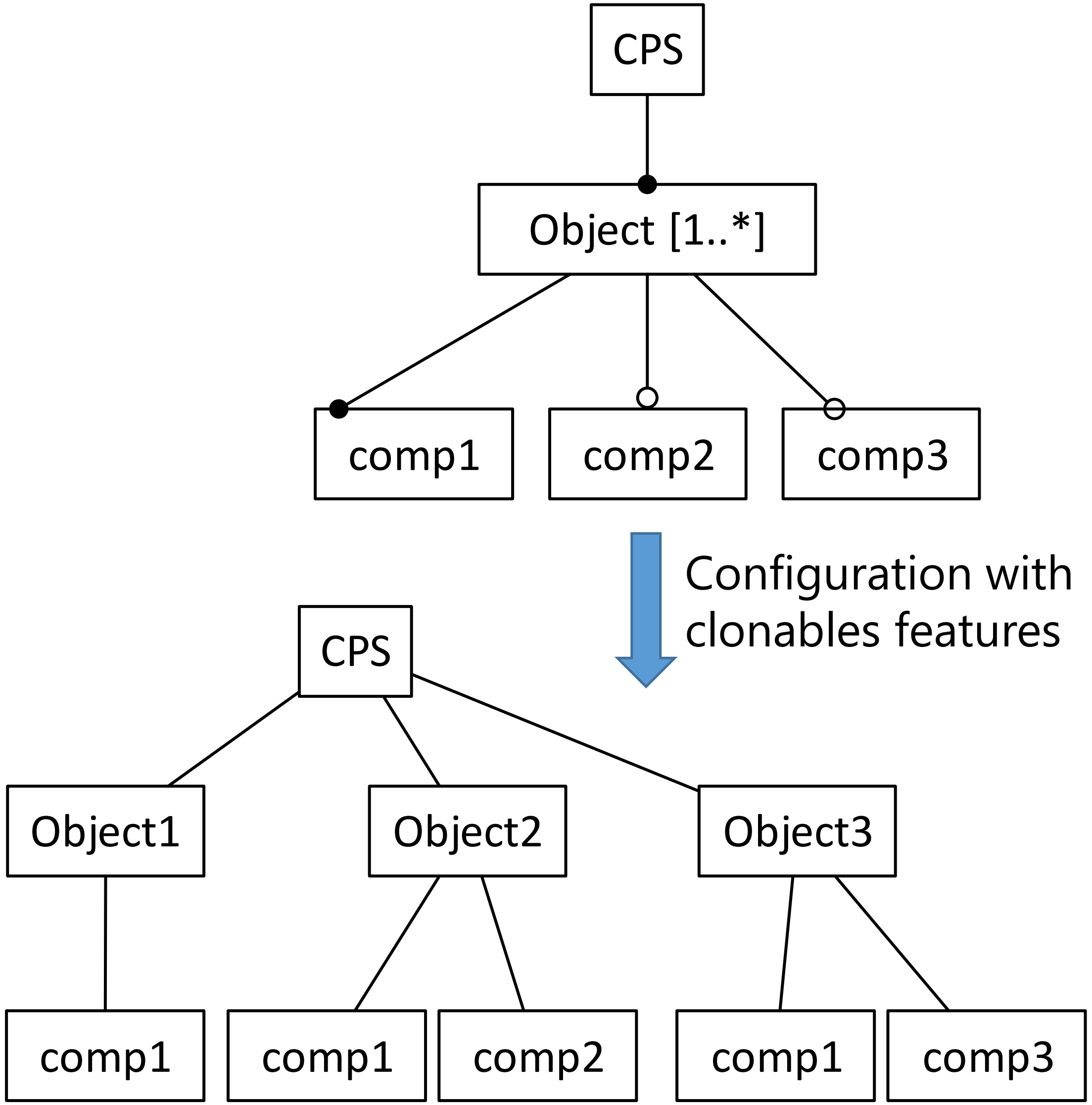


[Zhou et al, Expert Systems with Applications, 2017]

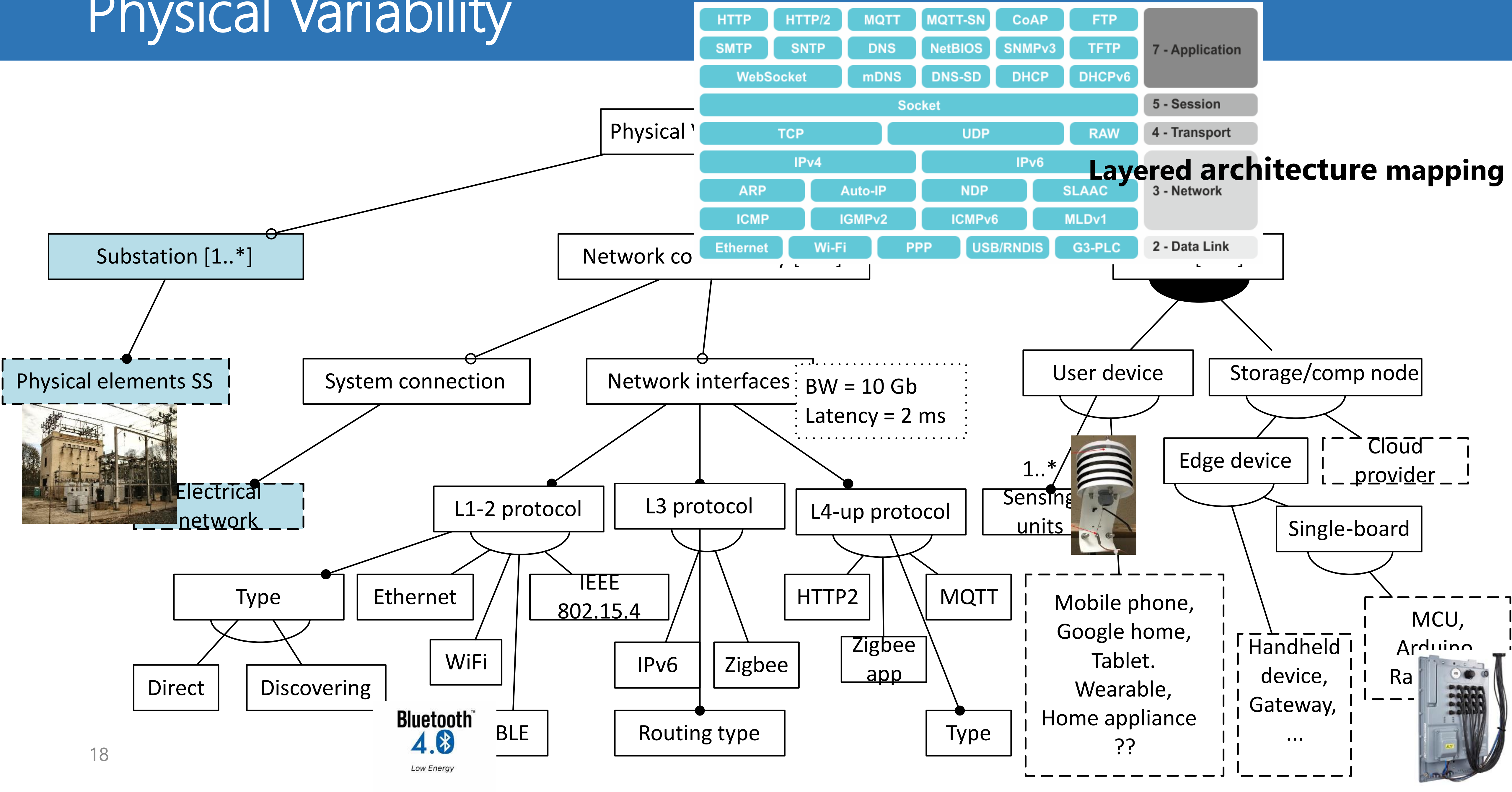
[Tutorial FAMILIAR]

Structural variability

- **Clonable features**
  - Features that represent **real world objects** (e.g. sensors, mobile phones, home appliances, etc.)
  - Each object will embed software, but adapted to the concrete role of the object inside the global system



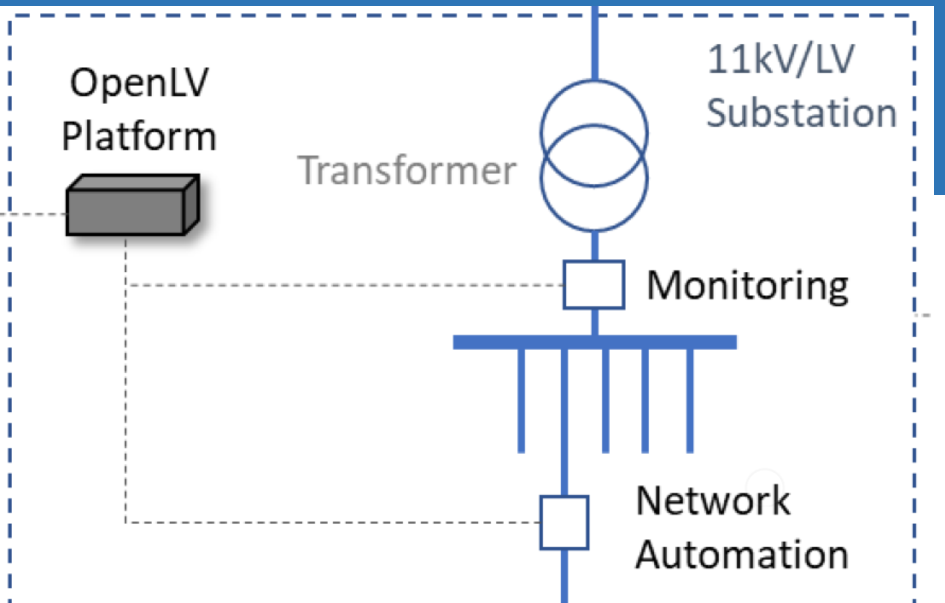
Physical Variability

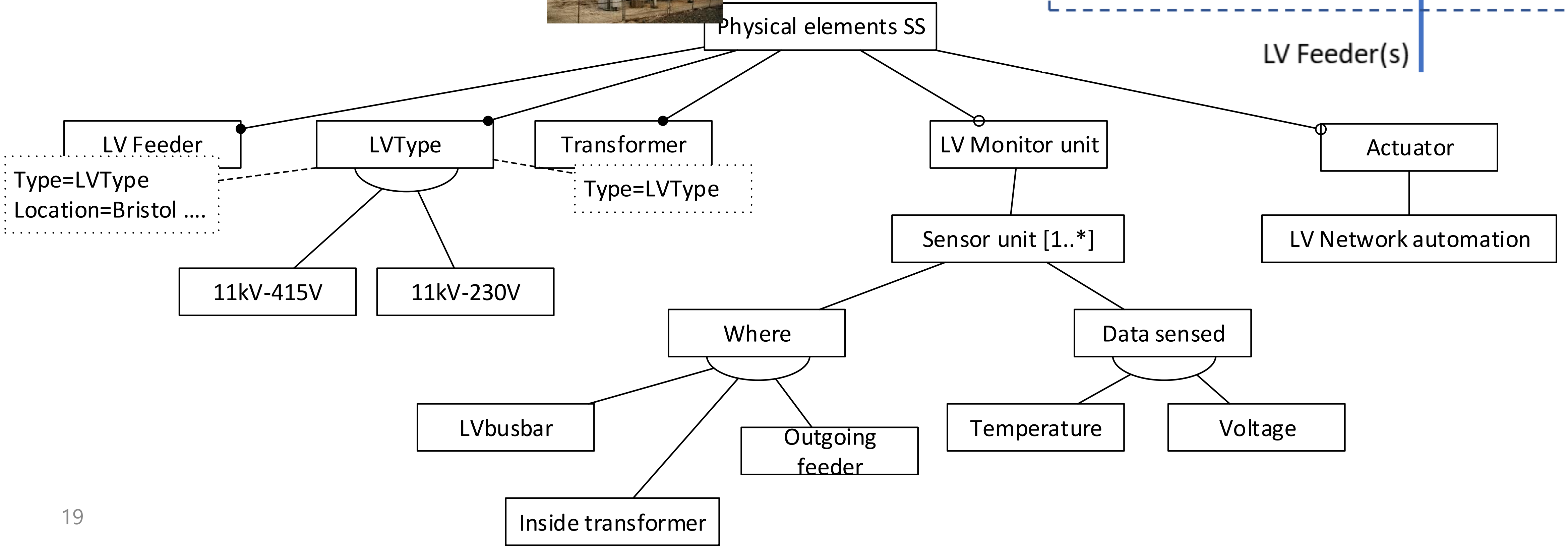


HTTP	HTTP/2	MQTT	MQTT-SN	CoAP	FTP	7 - Application
SMTP	SNTP	DNS	NetBIOS	SNMPv3	TFTP	
WebSocket		mDNS	DNS-SD	DHCP	DHCPv6	
Socket						5 - Session
TCP		UDP			RAW	4 - Transport
IPv4			IPv6			3 - Network
ARP	Auto-IP		NDP	SLAAC		
ICMP	IGMPv2		ICMPv6	MLDv1		
Ethernet	Wi-Fi	PPP	USB/RNDIS	G3-PLC		2 - Data Link

Layered architecture mapping

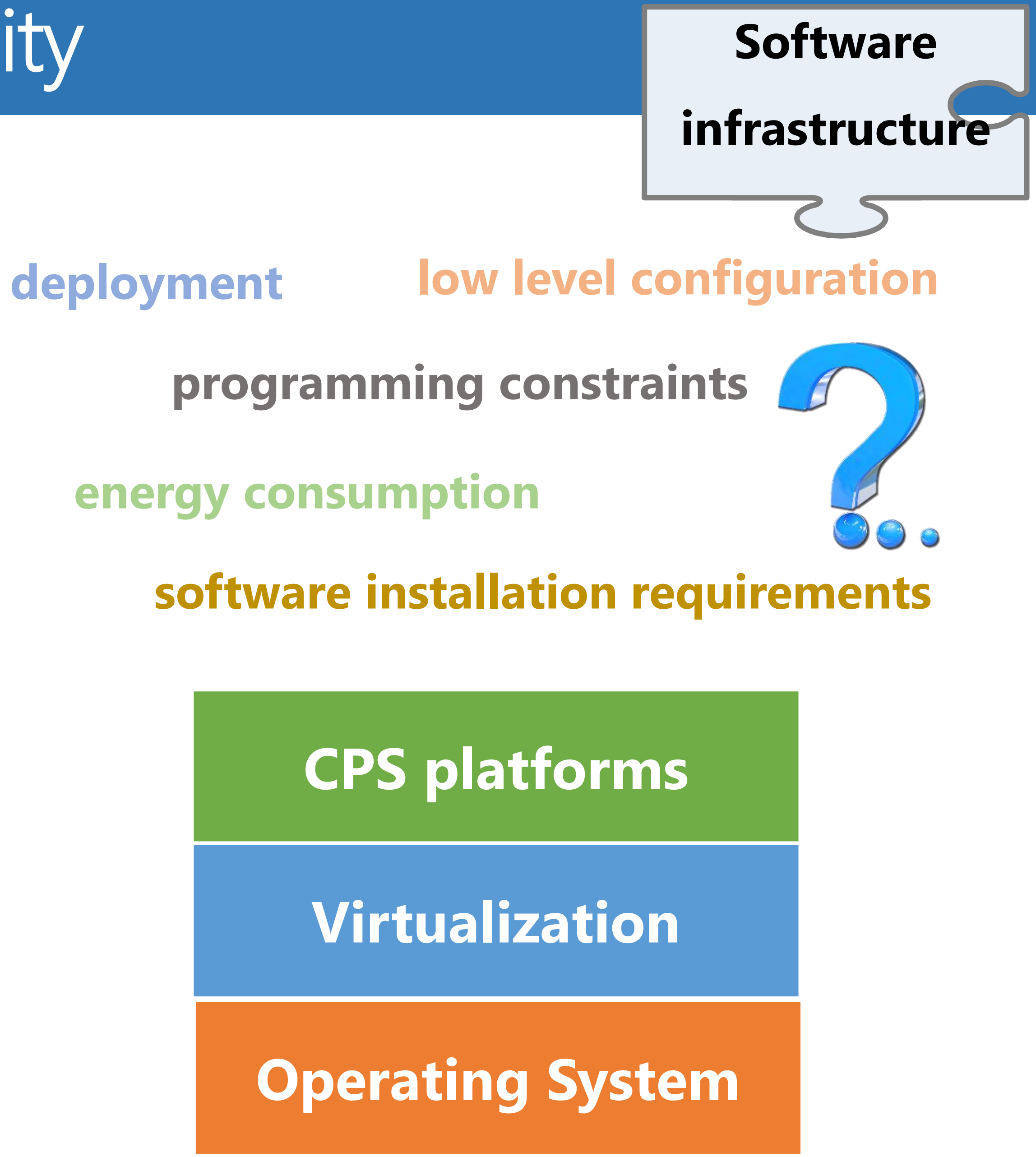
Power distribution system





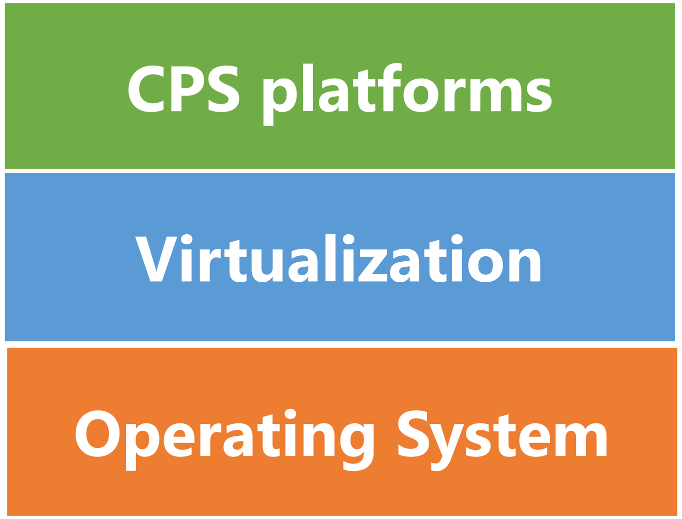
Software infrastructure variability

- **Operating system**
  - Programming model
  - Programming languages
  - Supported MCU vendors
  - License type
  - .....
- **Virtualization**
  - Configuring multiple devices: SDN and NFS
  - Virtual machines
- **Platforms**
  - Cloud and IoT platforms proliferation



Software infrastructure variability

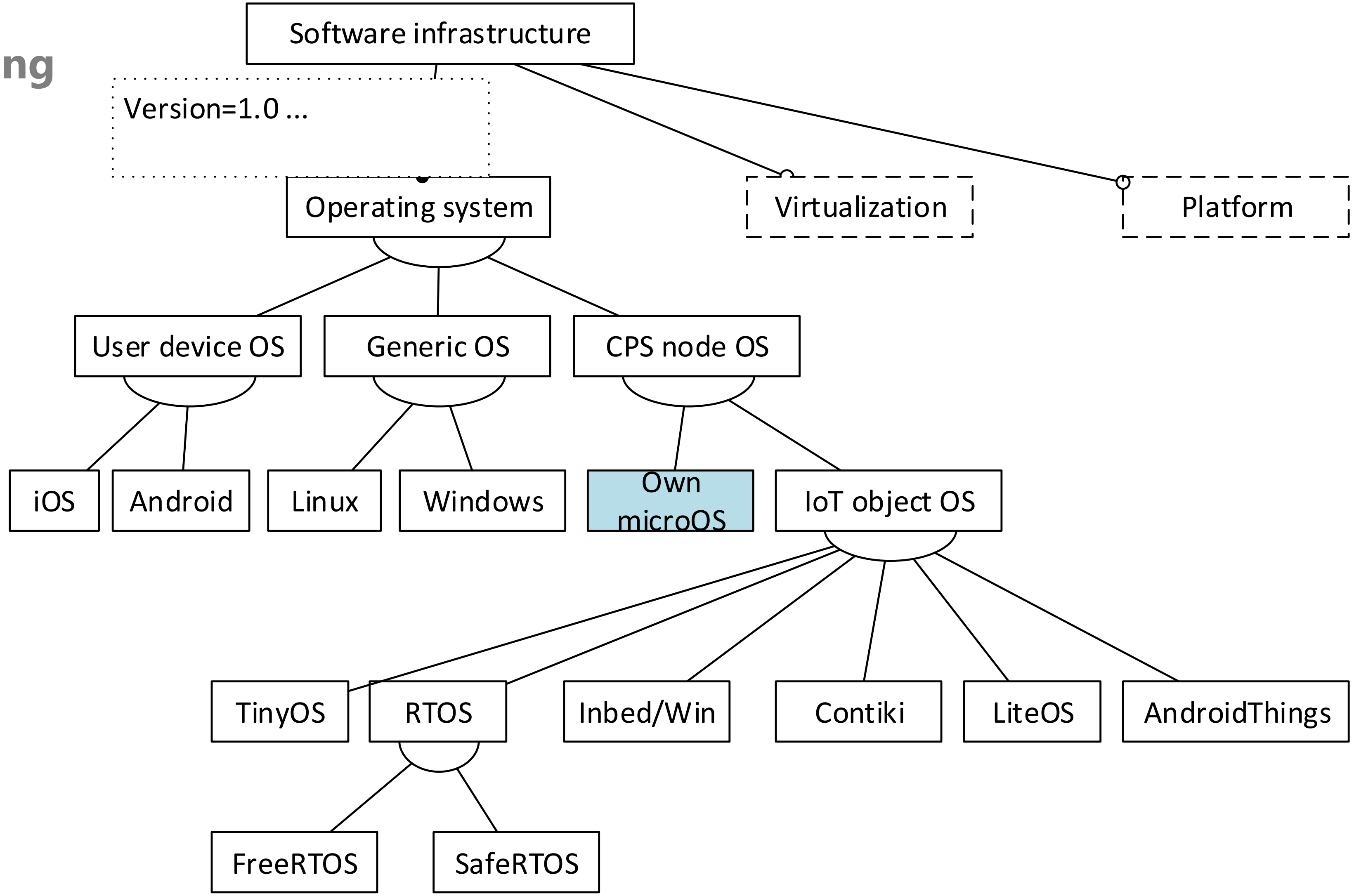
- **Layered architecture mapping**



CPS platforms

Virtualization

Operating System



Software infrastructure

Version=1.0 ...

Operating system

Virtualization

Platform

User device OS

Generic OS

CPS node OS

iOS

Android

Linux

Windows

Own microOS

IoT object OS

TinyOS

RTOS

Inbed/Win

Contiki

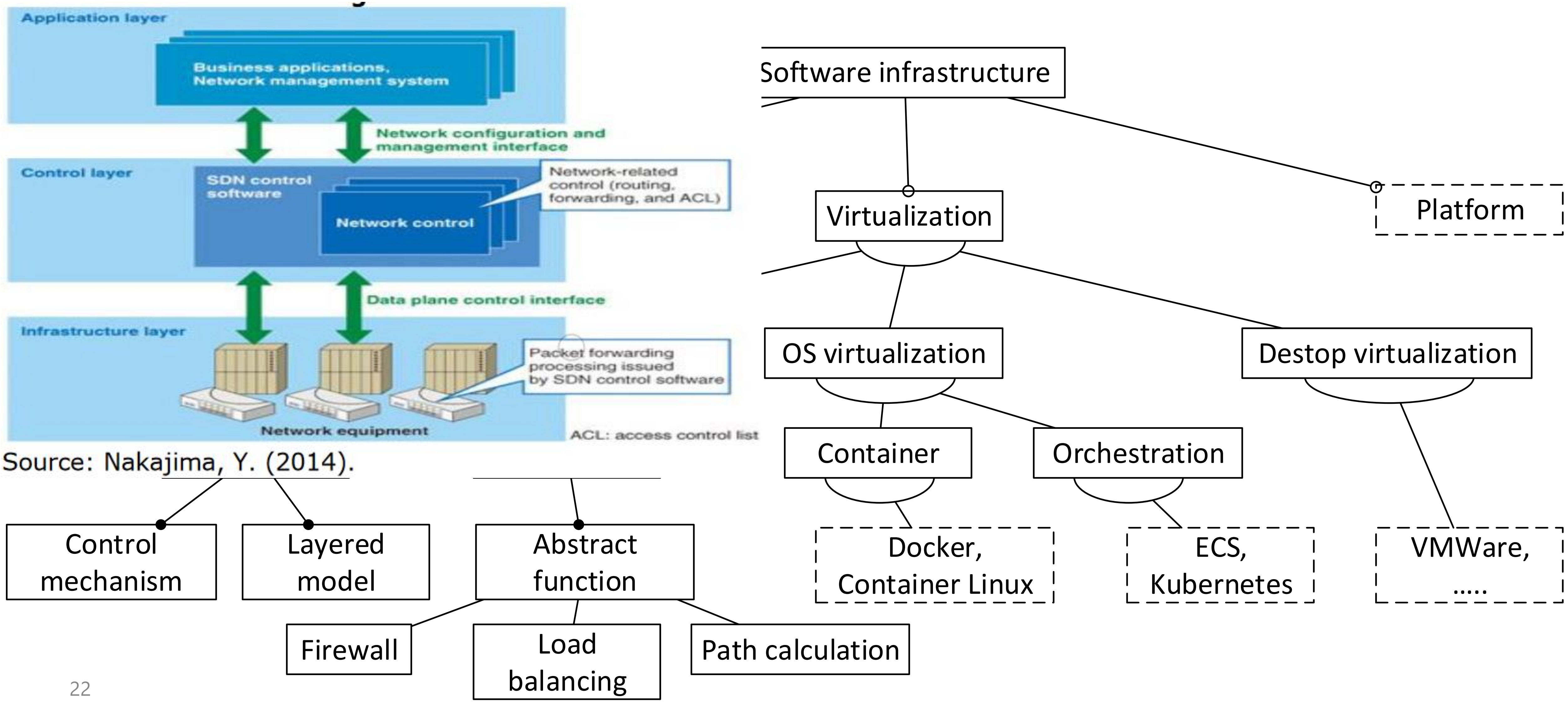
LiteOS

AndroidThings

FreeRTOS

SafeRTOS

Software infrastructure variability


Application layer
Business applications,
Network management system
Network configuration and management interface
Control layer
SDN control software
Network control
Network-related control (routing, forwarding, and ACL)
Data plane control interface
Infrastructure layer
Packet forwarding processing issued by SDN control software
Network equipment
ACL: access control list

Source: Nakajima, Y. (2014).

Software infrastructure

Virtualization

Platform

OS virtualization

Destop virtualization

Container

Orchestration

Docker, Container Linux

ECS, Kubernetes

VMWare, ...

Control mechanism

Layered model

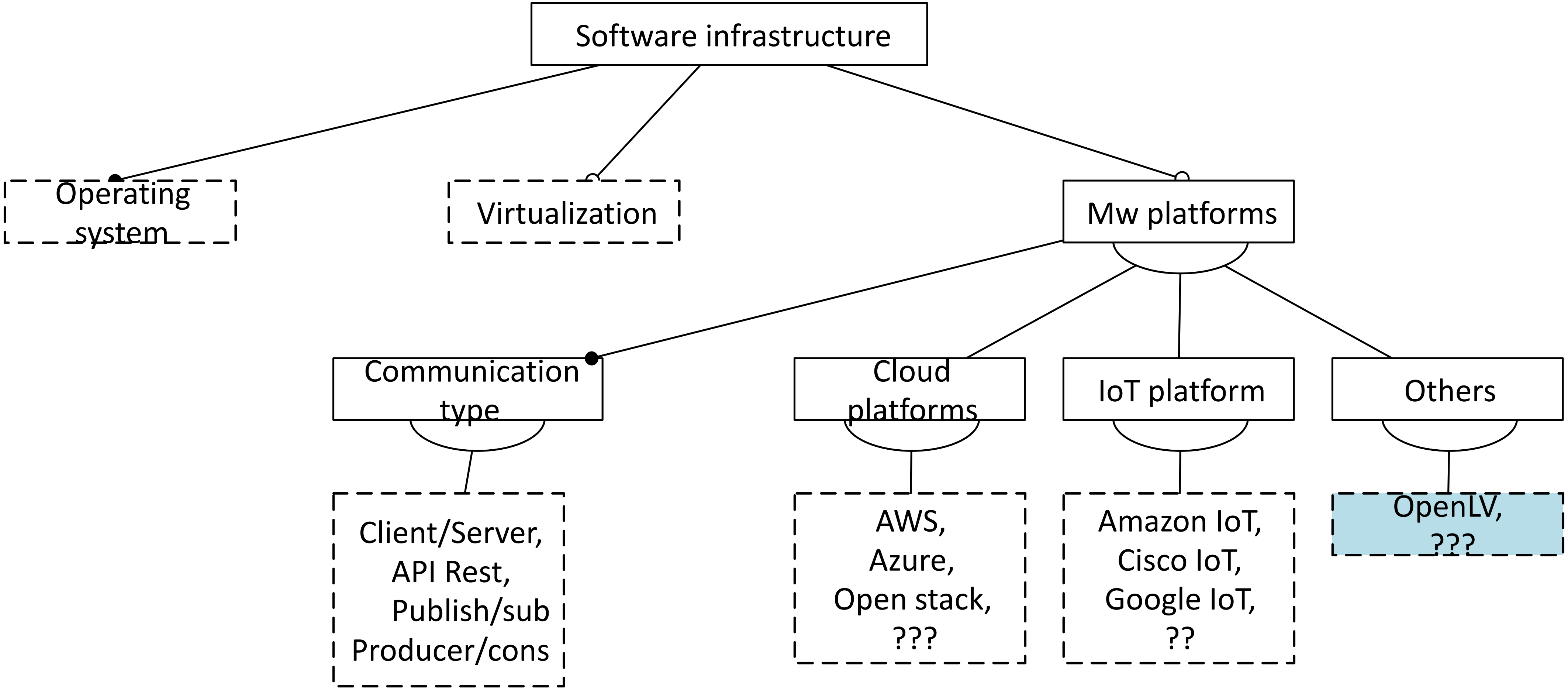
Abstract function

Firewall

Load balancing

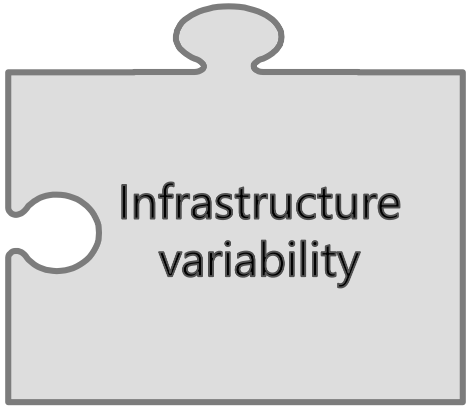
Path calculation

Software infrastructure variability



Why an infrastructure variability model?

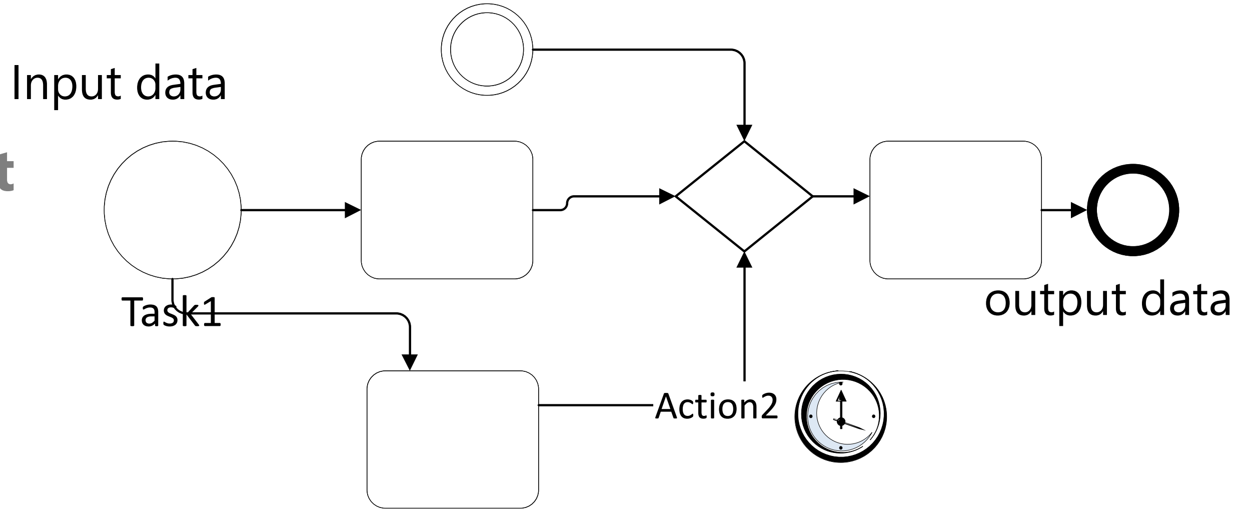
- Represent the **deployment infrastructure**
- To define CPS variability in **computational resources** and **communication protocols**
- To configure the **specific technology used** in a concrete CPS
- To **help to configure the software components/services** of the applications to be deployed there
- To represent the **logic and physical connections** among the equipments that conform a CPS
- To express that a **common infrastructure** is shared by several applications/services
- To **handle the insfrastructure evolution** by the domain engineer (only once, reuse)



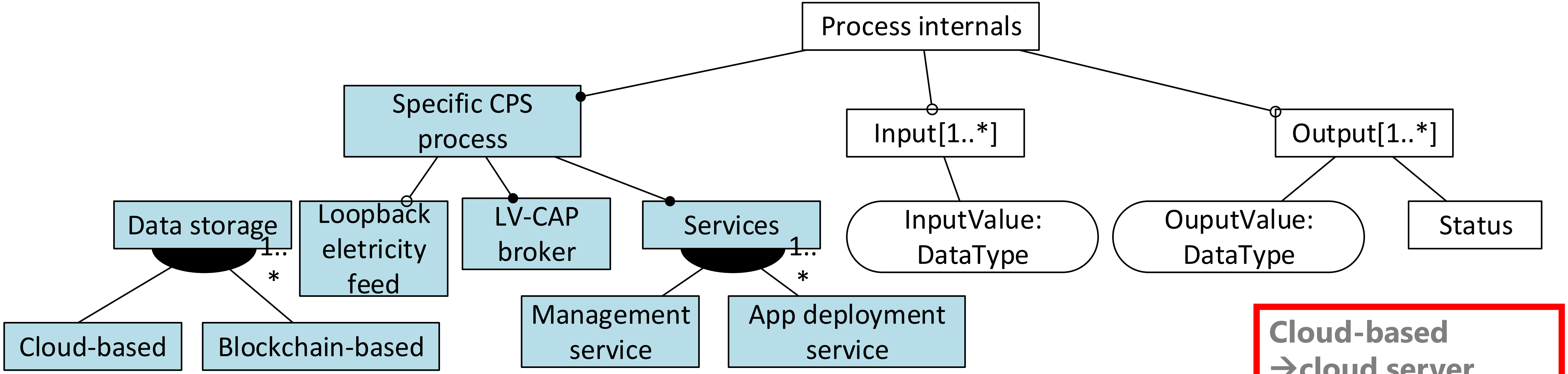
Process variability

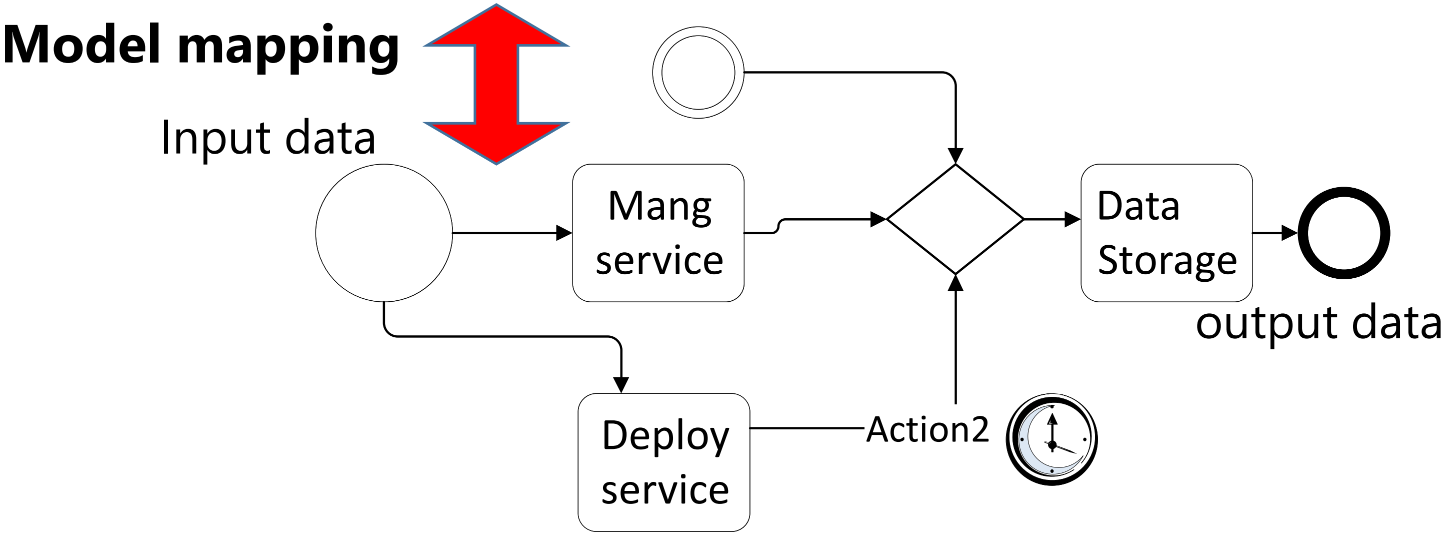
Process variability

- **Family of common processes** used in CPSs
- **Variability in tasks**, subtasks, algorithms, computation, …..
- Each task variant can **require different resources and capacity** of the devices
- Use classical models to specify which **tasks** can be implemented in **parallel, sequence**
- Tasks must be deployed in **cloud datacenter**, or can **migrate to another device, …**



Process variability



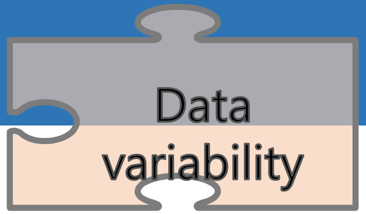


Cloud-based →cloud server

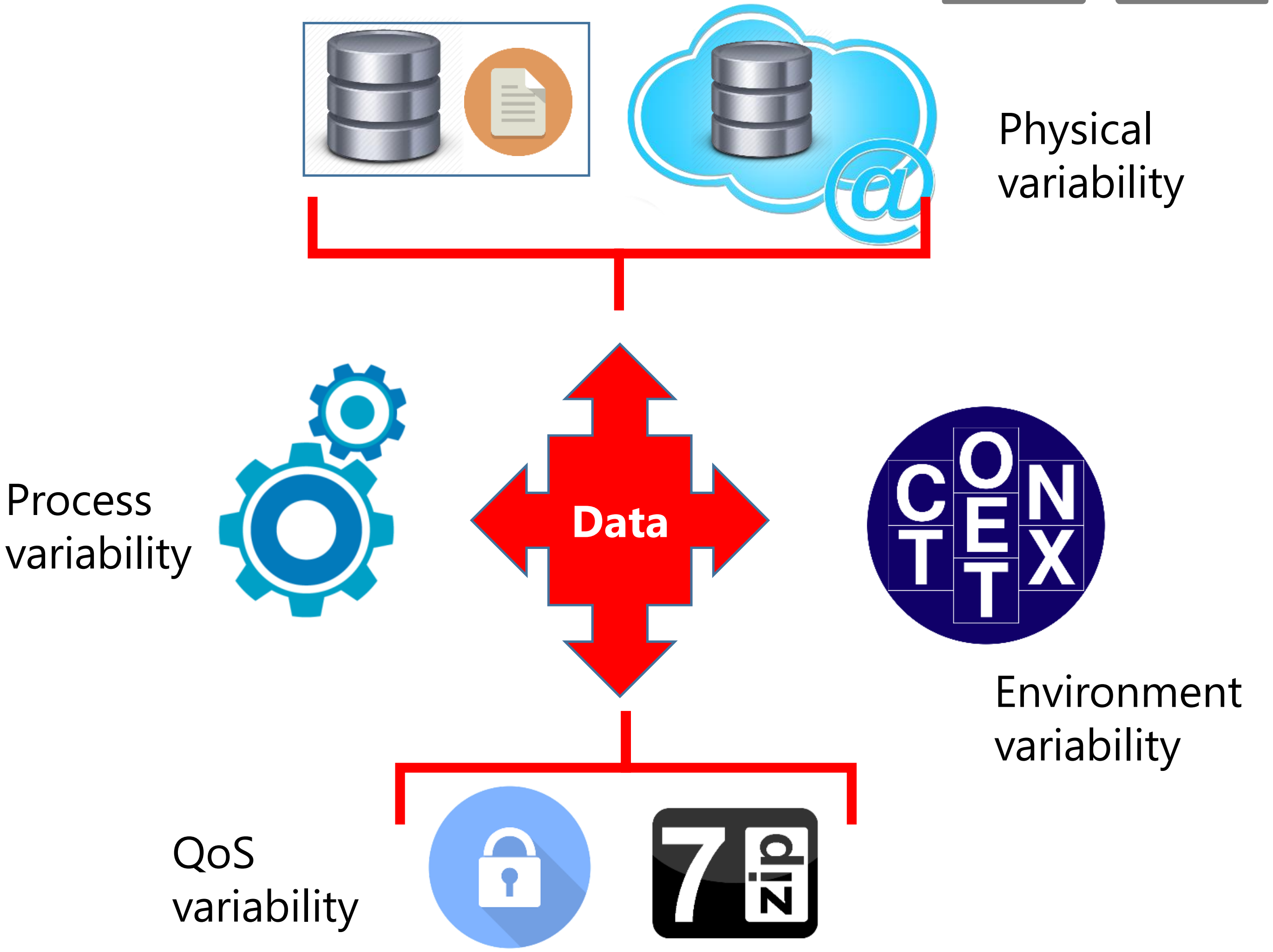
LV-CAP broker→LV machine

With CVL was possible to relate variation points and other models implemented in MOF

Data variability

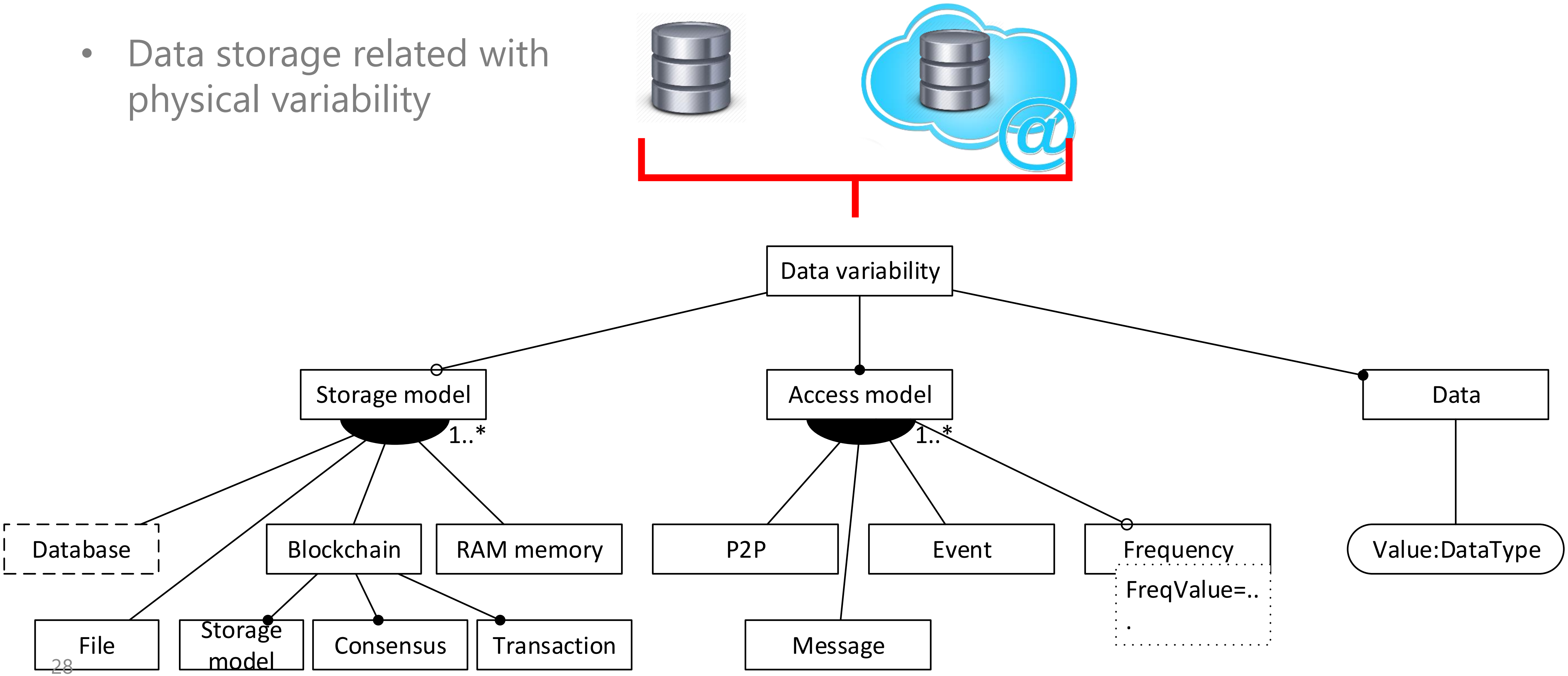


- Data storage
- Data types
- Different data access models
- The data is produced by processes
- Some of the process data is part of the context
- Relationships with quality attributes



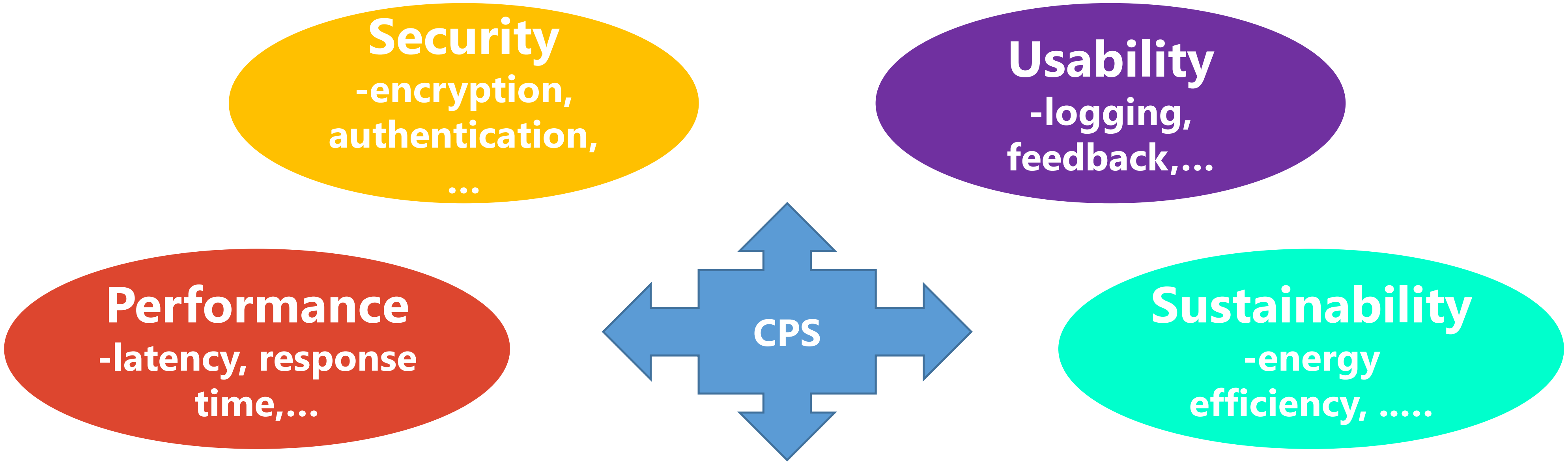
Data variability

- Data storage related with physical variability



Quality attributes variability

QA variabiity



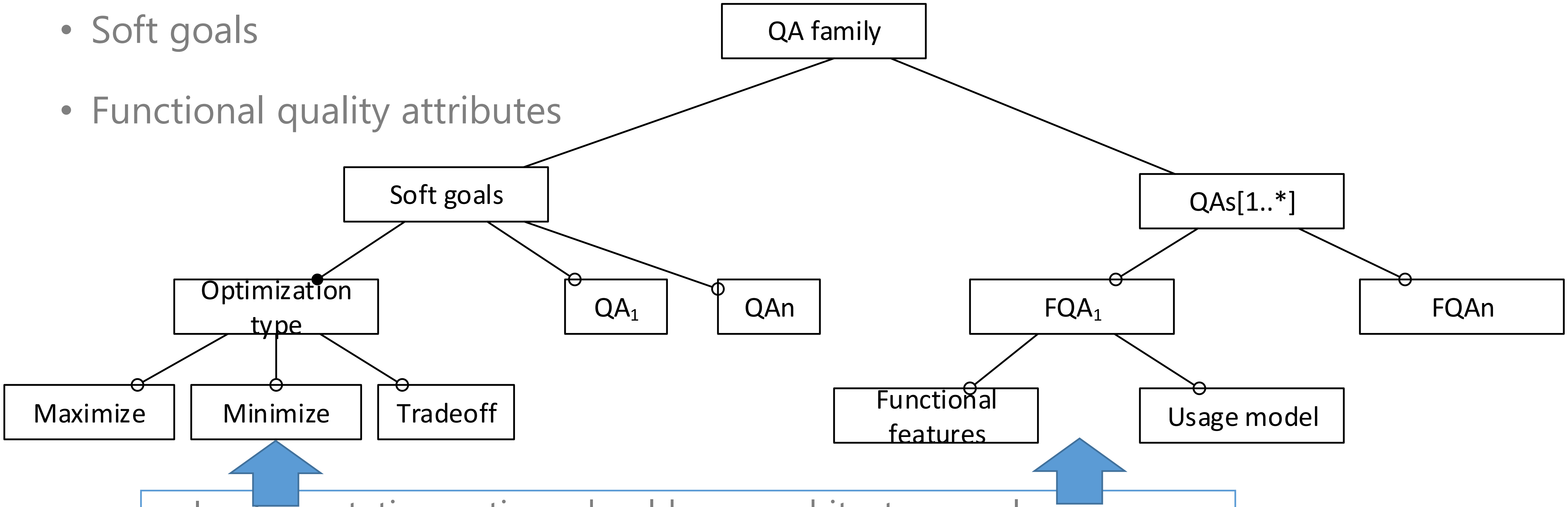
Functional QAs ⇒ **Software architecture**

Measure QAs ⇒ **Optimization**

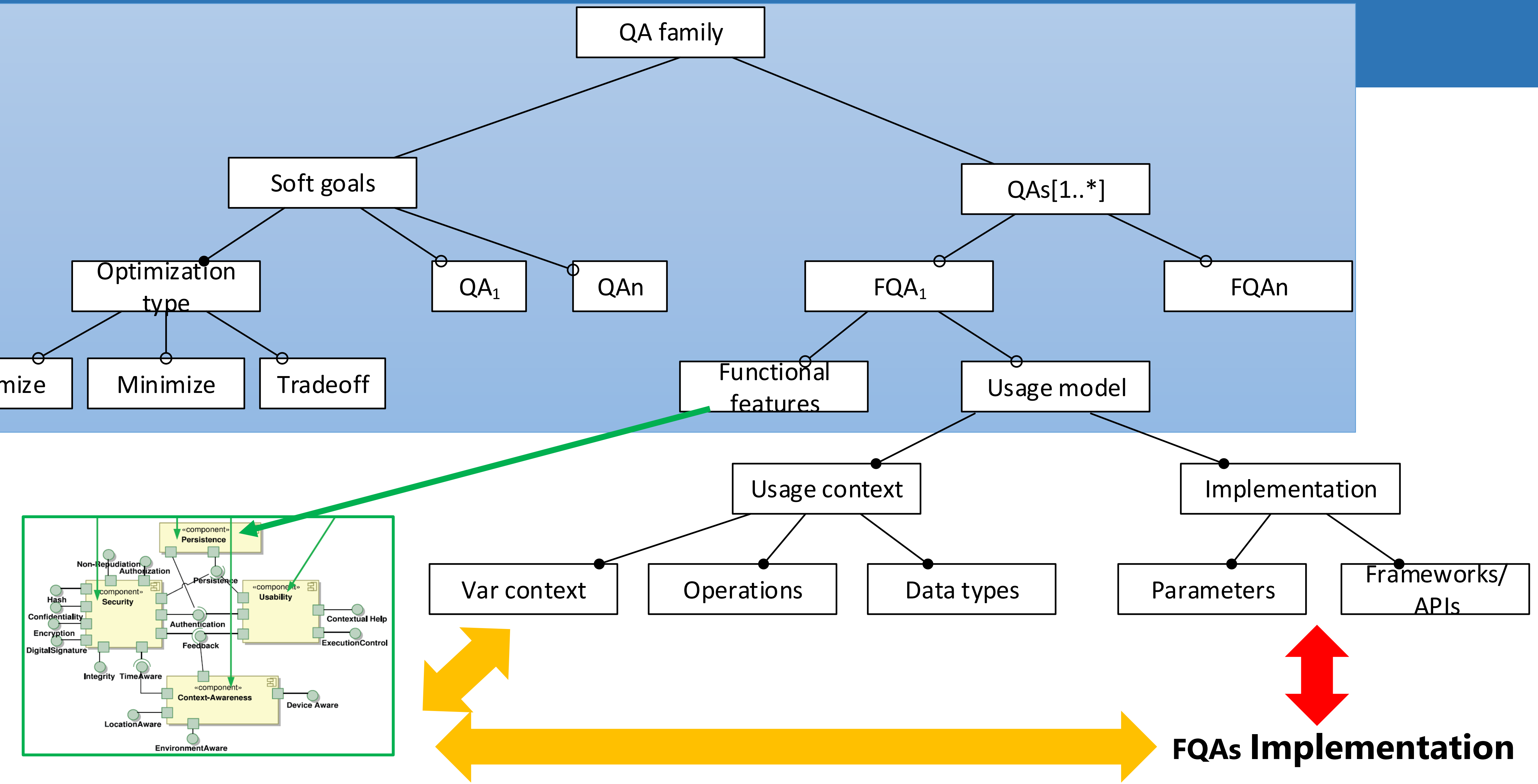
$- \cdots +$

Quality attributes variability

- CPS architecture should meet
  - Soft goals
  - Functional quality attributes



- Implementation options should map architecture goals
  - High diversity of implementation options
  - Need to quantify quality attributes for each implementation option



FQAs Implementation

Functional quality attributes in Open LV

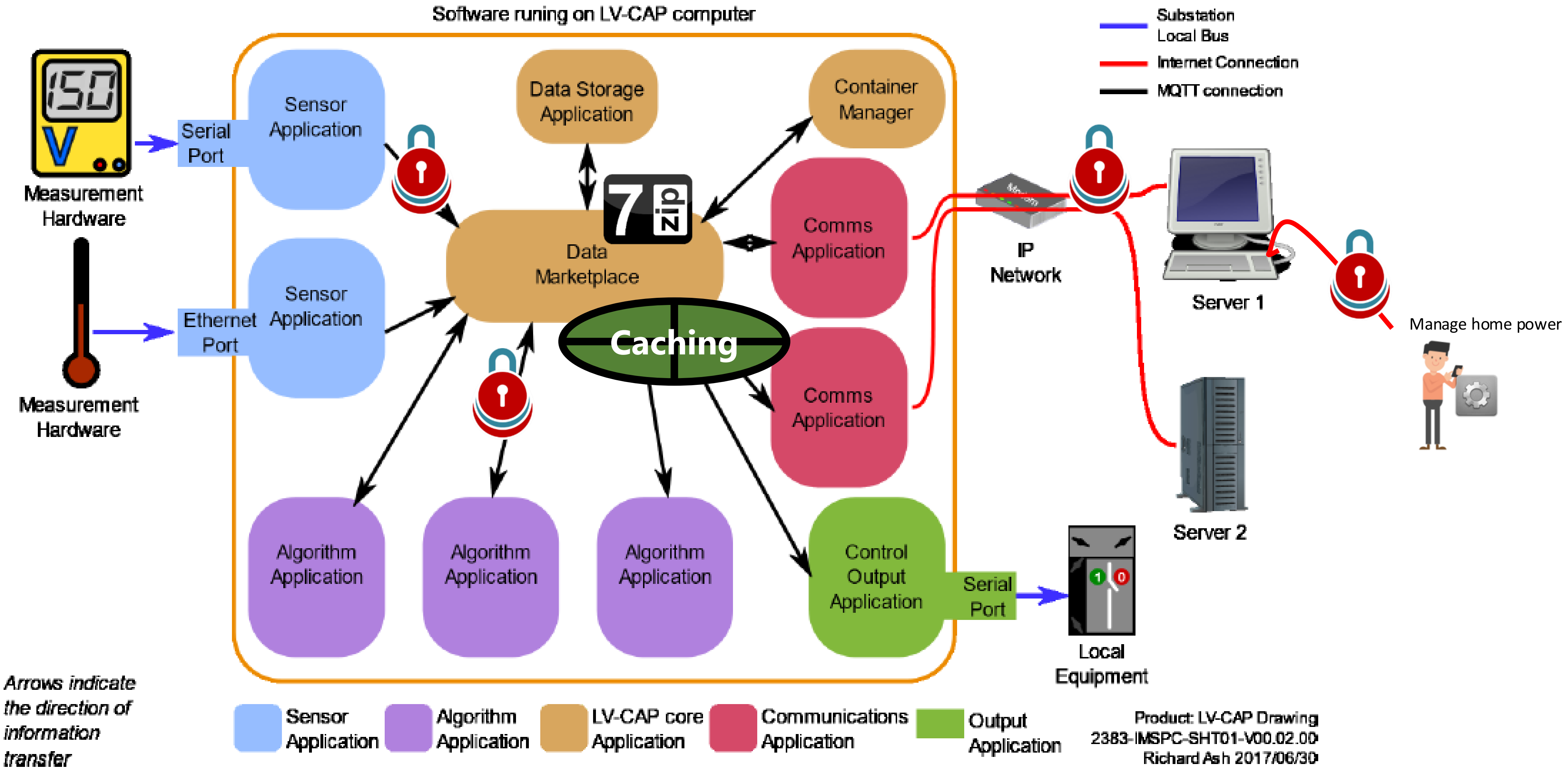
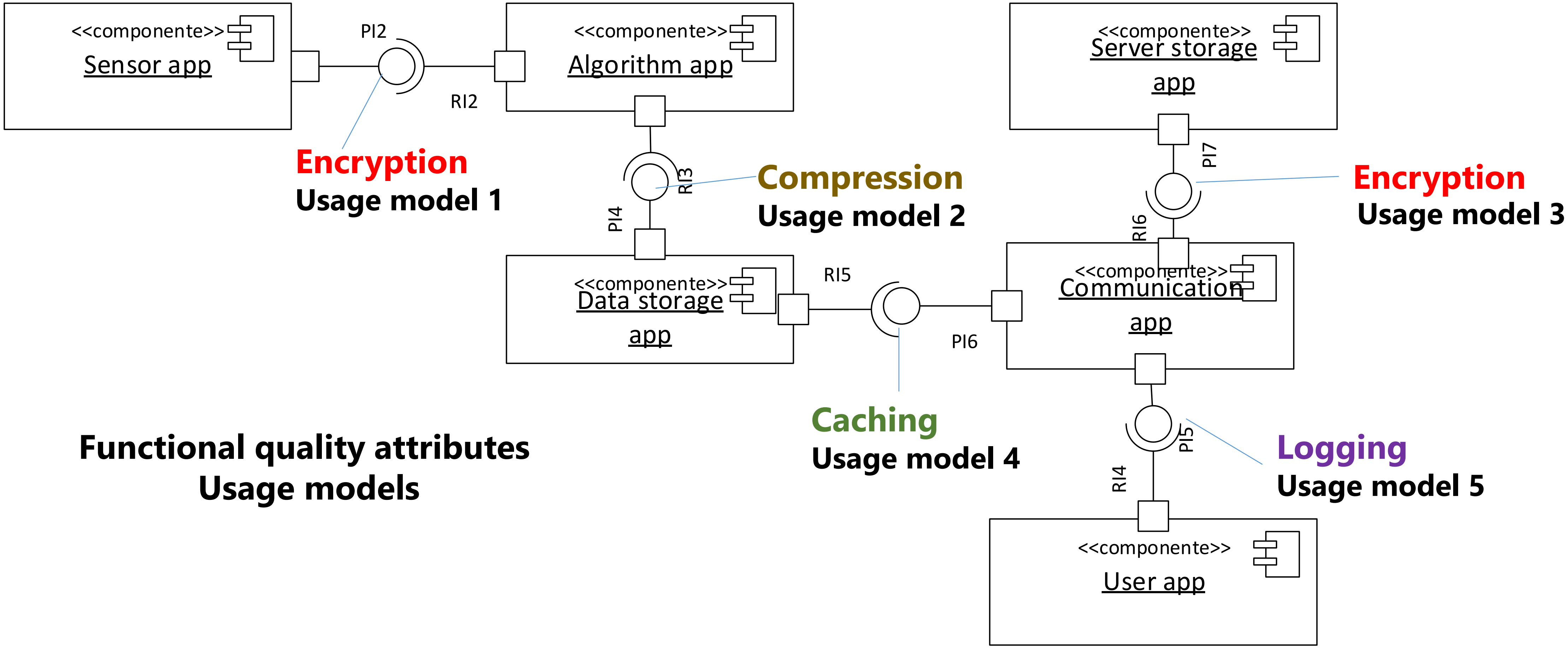


Figure 2 - LV-CAP Software Architecture

Open LV software architecture and FQAs



Encryption
Usage model 1

Compression
Usage model 2

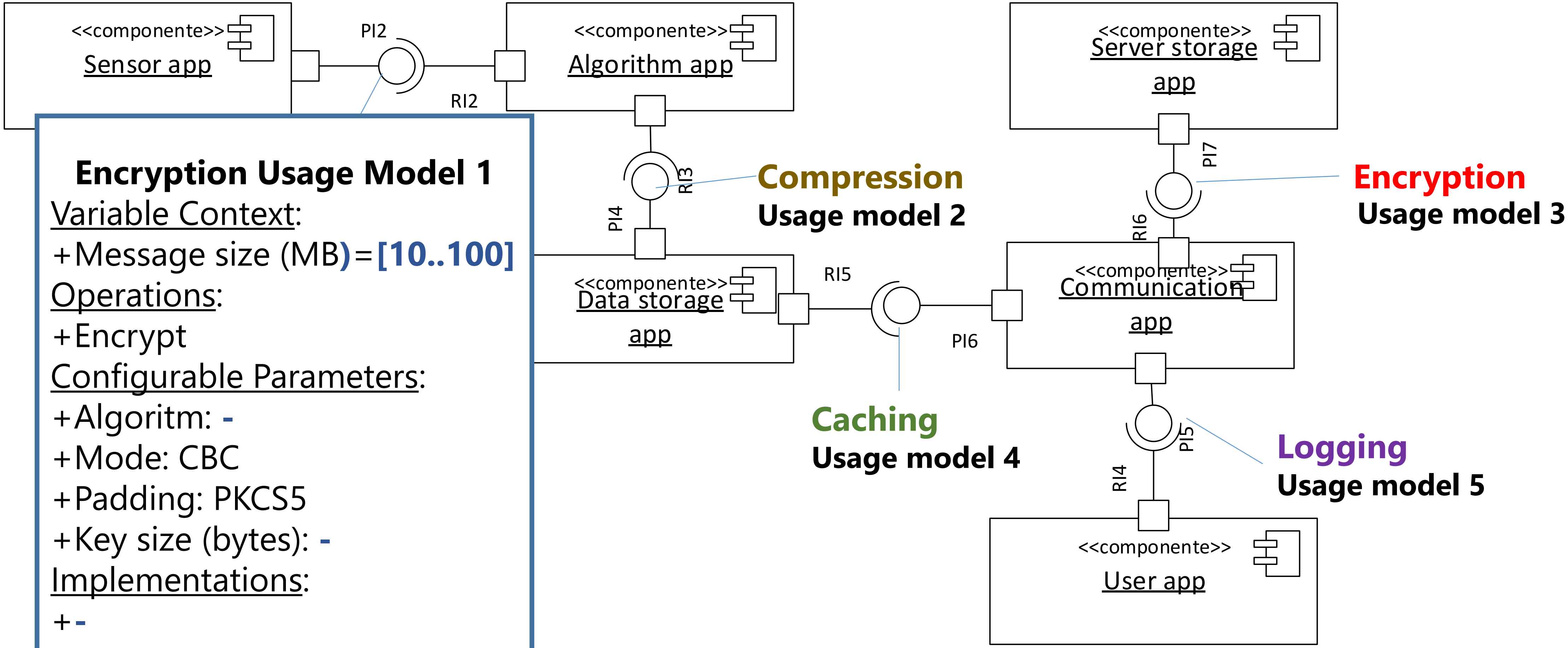
Encryption
Usage model 3

Caching
Usage model 4

Logging
Usage model 5

Functional quality attributes
Usage models

Open LV software architecture and FQAs



Encryption Usage Model 1

Variable Context:

+Message size (MB)=**[10..100]**

Operations:

+Encrypt

Configurable Parameters:

+Algoritm: **-**

+Mode: CBC

+Padding: PKCS5

+Key size (bytes): **-**

Implementations:

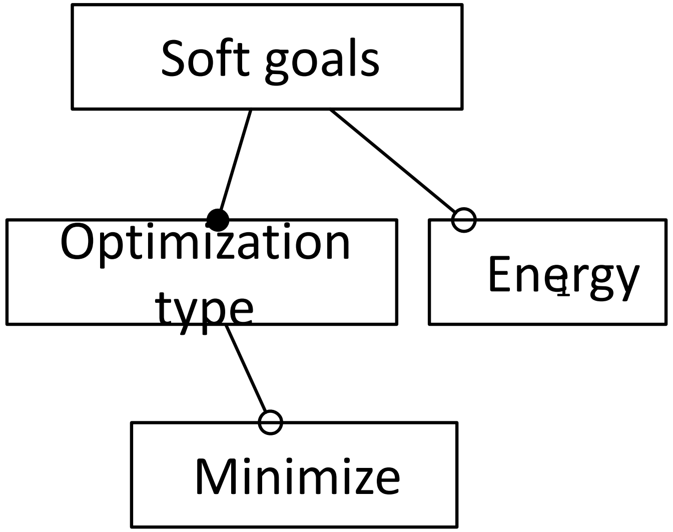
+**-**

QAs and feature models

- Extend FMs to model attributes

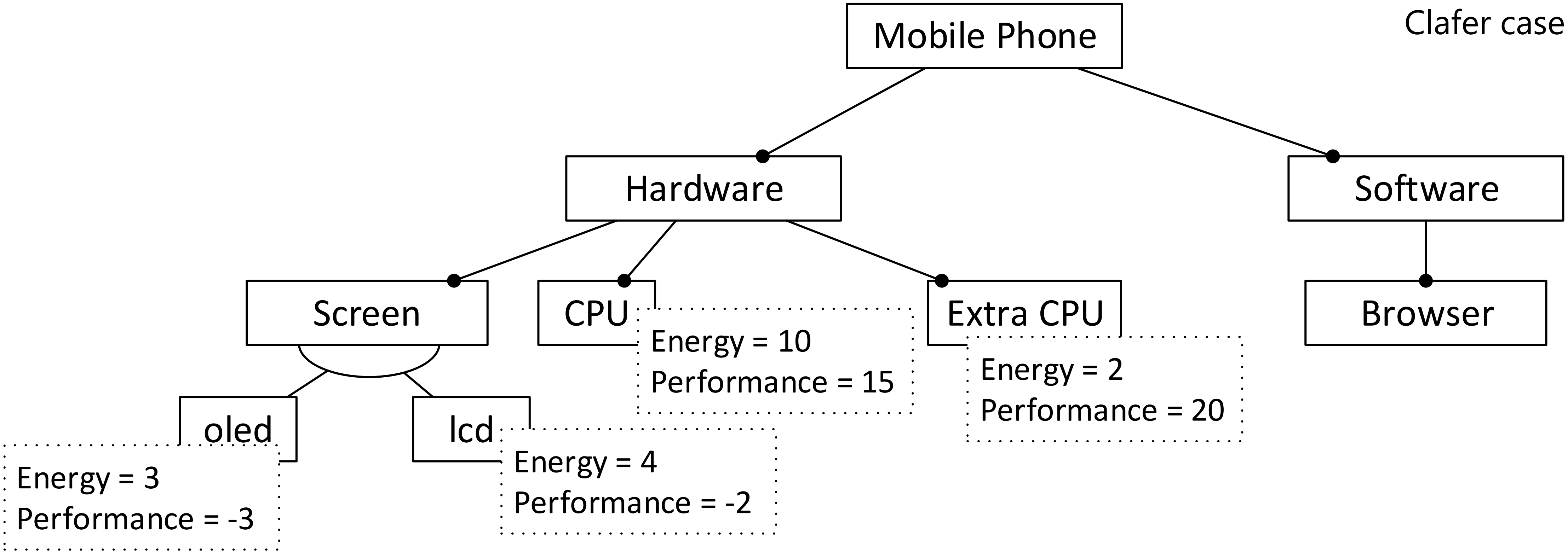
- Represent the generation of the best configuration as a optimization problem

  - E.g. **what is the configuration of the encryption component that consumes the least power?**



QAs and feature models

Clafer case study



Total performance -> sum c.performance

Total energy -> sum c.energy

<< min aPhone.total energy >>

<< max aPhone.performance >>

- **One feature ->qa value (limited)**

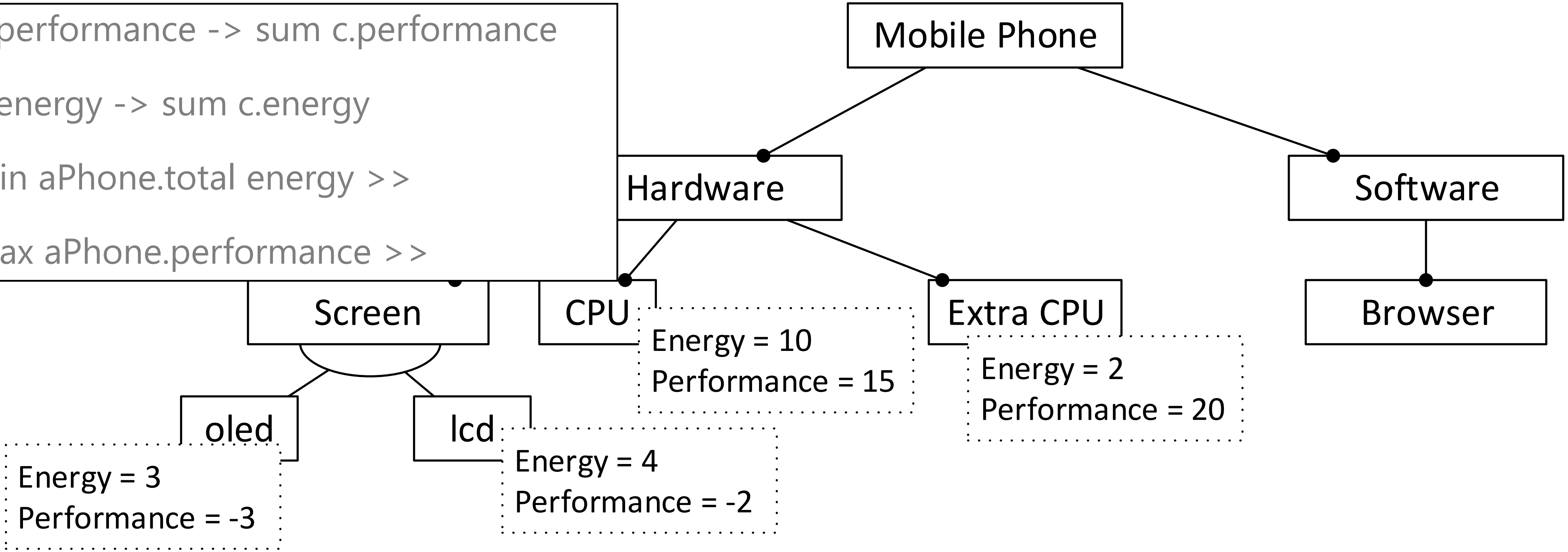
QAs and feature models

Total performance -> sum c.performance

Total energy -> sum c.energy

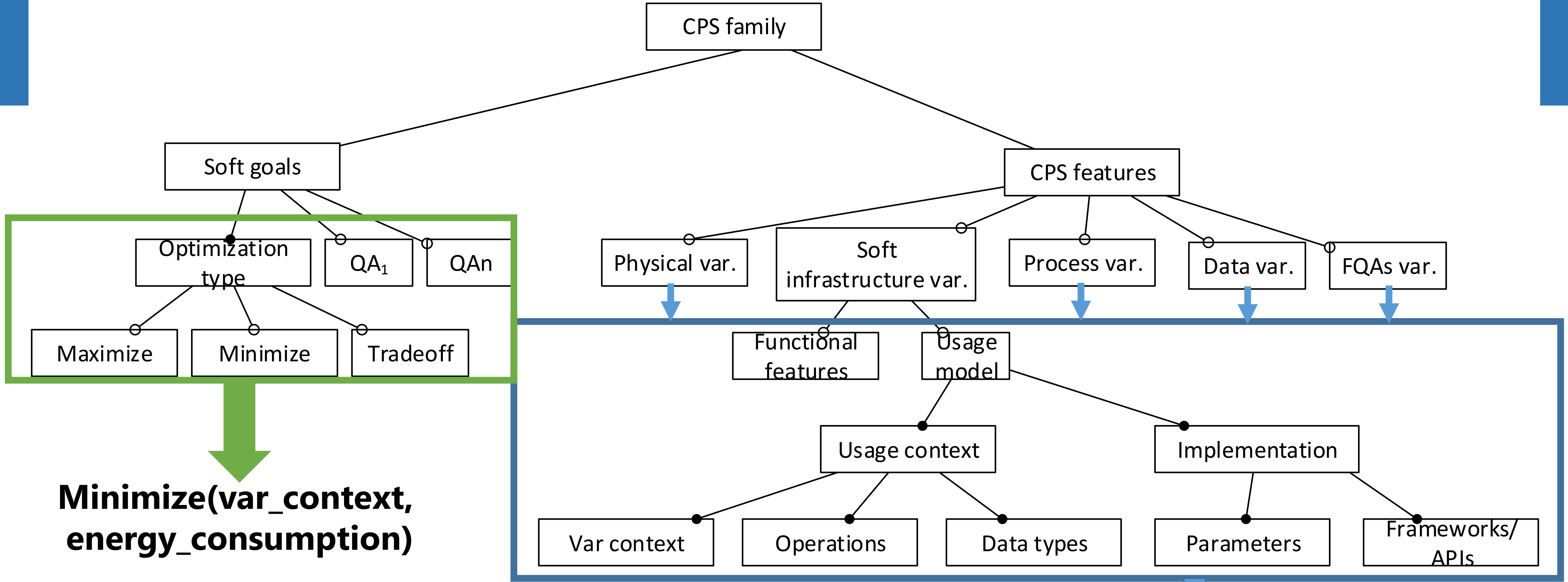
<< min aPhone.total energy >>

<< max aPhone.performance >>

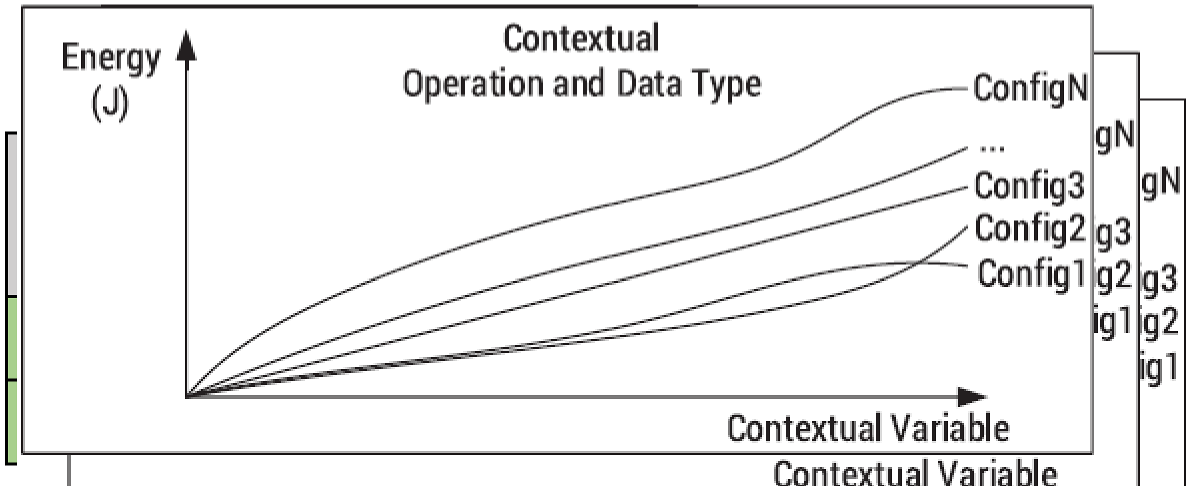


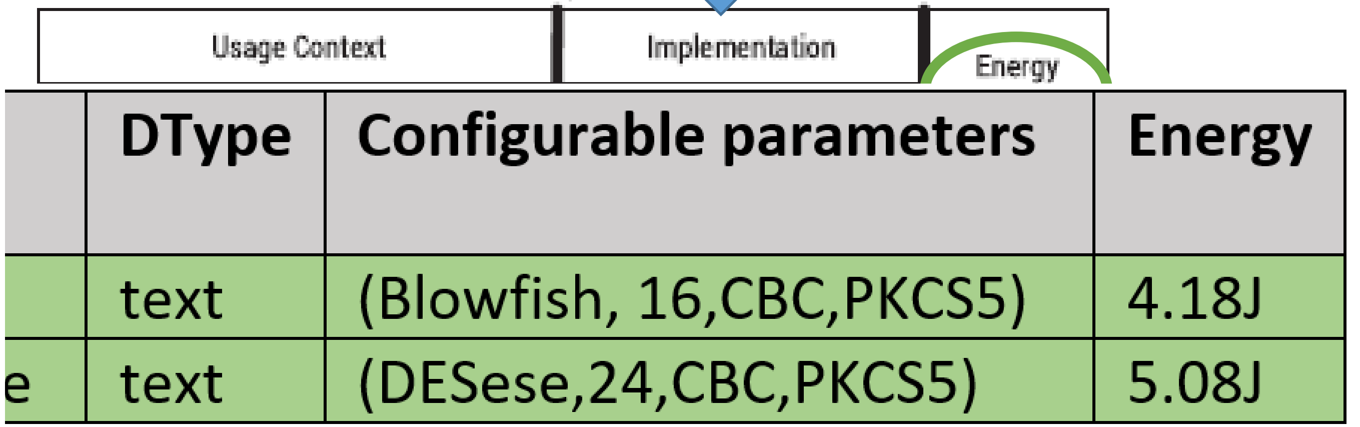
- **Energy depends on**
  - usage frequency of browser
  - Graphics displayed, ….

- **One feature ->qa value (limited)**
- **One configuration -> qa value**



Minimize(var_context, energy_consumption)

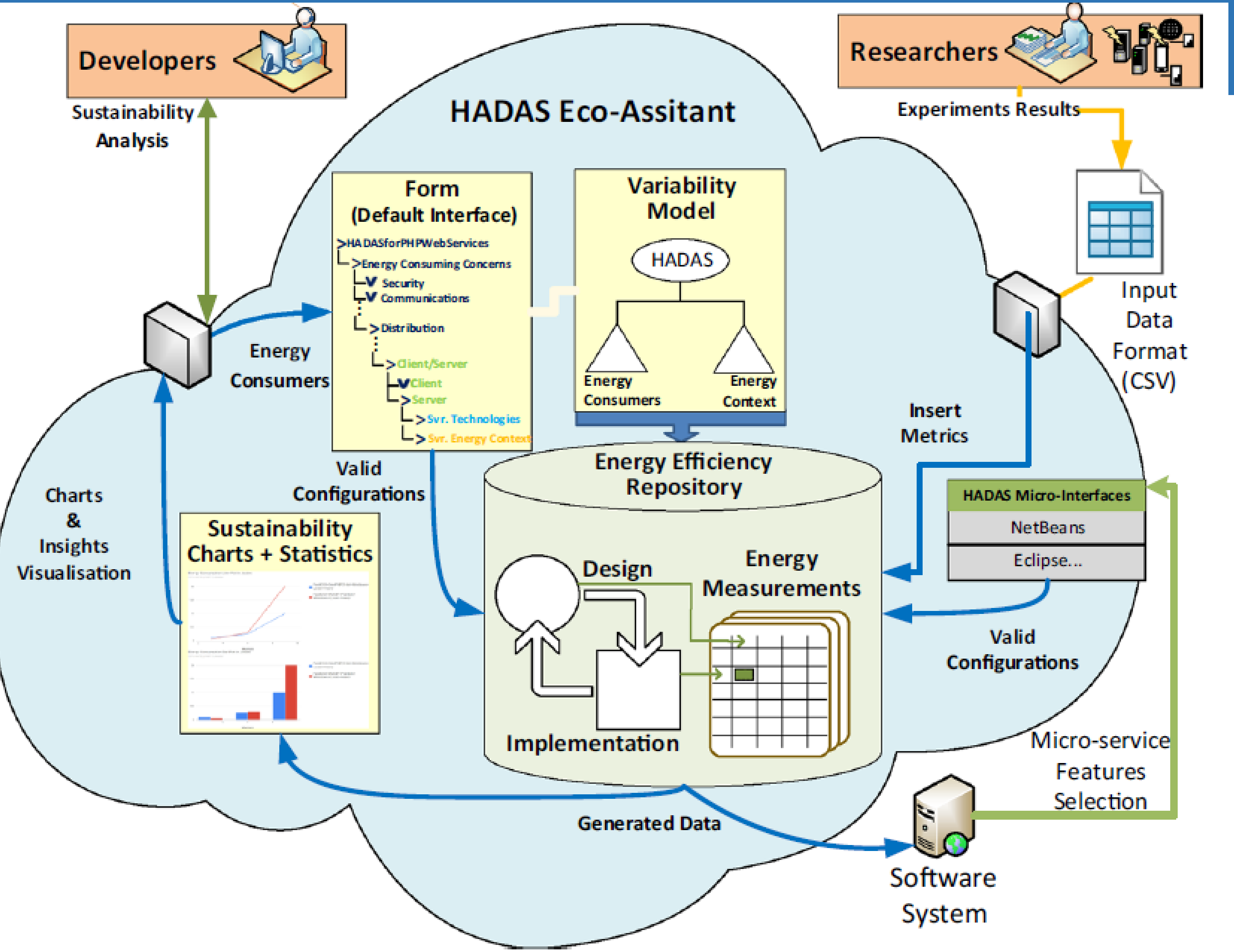




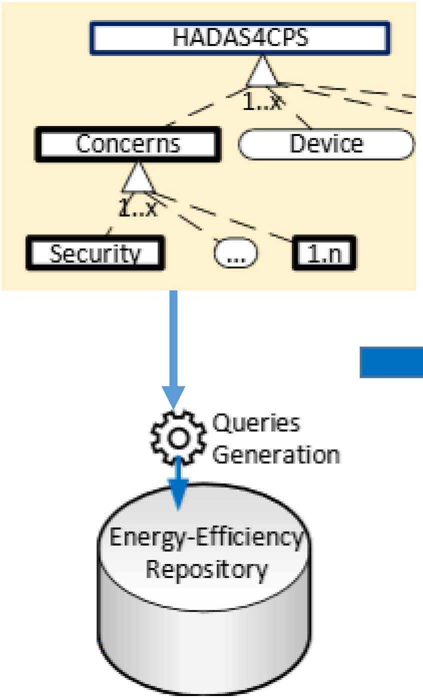
Usage Context		Implementation	Energy
	DType	Configurable parameters	Energy
	text	(Blowfish, 16,CBC,PKCS5)	4.18J
e	text	(DESese,24,CBC,PKCS5)	5.08J

HADAS

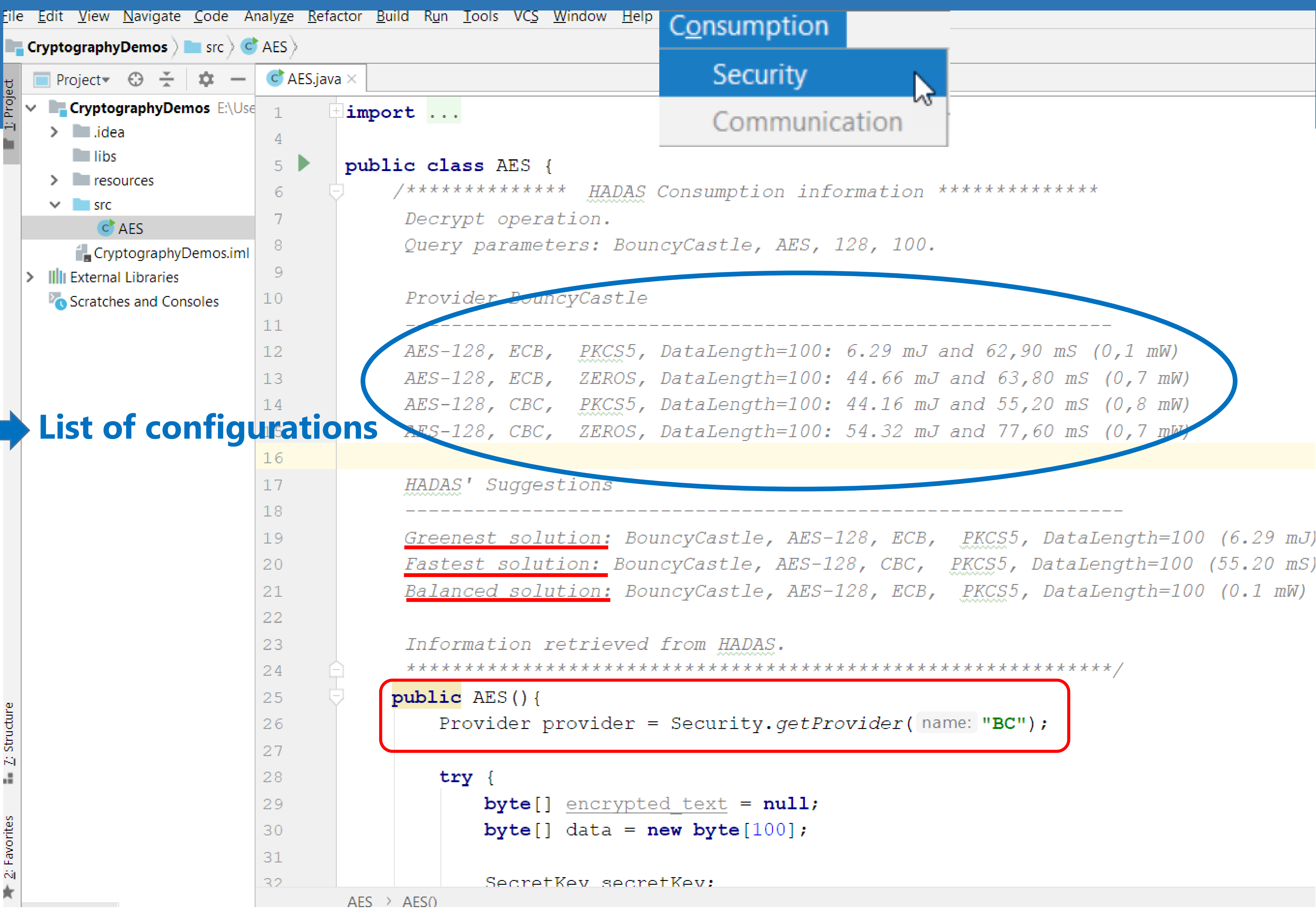
- Sustainability of CPSs
- **Energy consuming concerns** are common to many apps
- Choose the **most energy efficient implementation**
- Store energy and performance information in a **repository**
- Make **sustainability analysis**
- **Include it in IDEs** typical of CPSs



HADAS

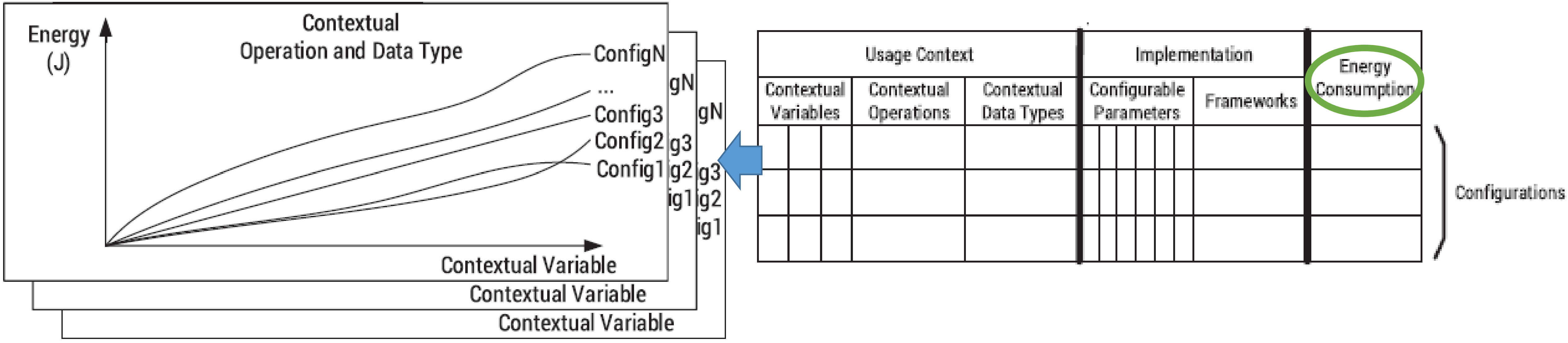


List of configurations



```java
import ...

public class AES {
    /**************  HADAS Consumption information **************
     Decrypt operation.
     Query parameters: BouncyCastle, AES, 128, 100.

     Provider BouncyCastle
     ----------------------------------------------------------------
     AES-128, ECB,  PKCS5, DataLength=100: 6.29 mJ and 62,90 mS (0,1 mW)
     AES-128, ECB,  ZEROS, DataLength=100: 44.66 mJ and 63,80 mS (0,7 mW)
     AES-128, CBC,  PKCS5, DataLength=100: 44.16 mJ and 55,20 mS (0,8 mW)
     AES-128, CBC,  ZEROS, DataLength=100: 54.32 mJ and 77,60 mS (0,7 mW)

     HADAS' Suggestions
     ----------------------------------------------------------------
     Greenest solution: BouncyCastle, AES-128, ECB,  PKCS5, DataLength=100 (6.29 mJ
     Fastest solution: BouncyCastle, AES-128, CBC,  PKCS5, DataLength=100 (55.20 mS
     Balanced solution: BouncyCastle, AES-128, ECB,  PKCS5, DataLength=100 (0.1 mW)

     Information retrieved from HADAS.
     ****************************************************************/
    public AES(){
        Provider provider = Security.getProvider( name: "BC");

        try {
            byte[] encrypted_text = null;
            byte[] data = new byte[100];

            SecretKey secretKey;
```

Limitations of HADAS approach



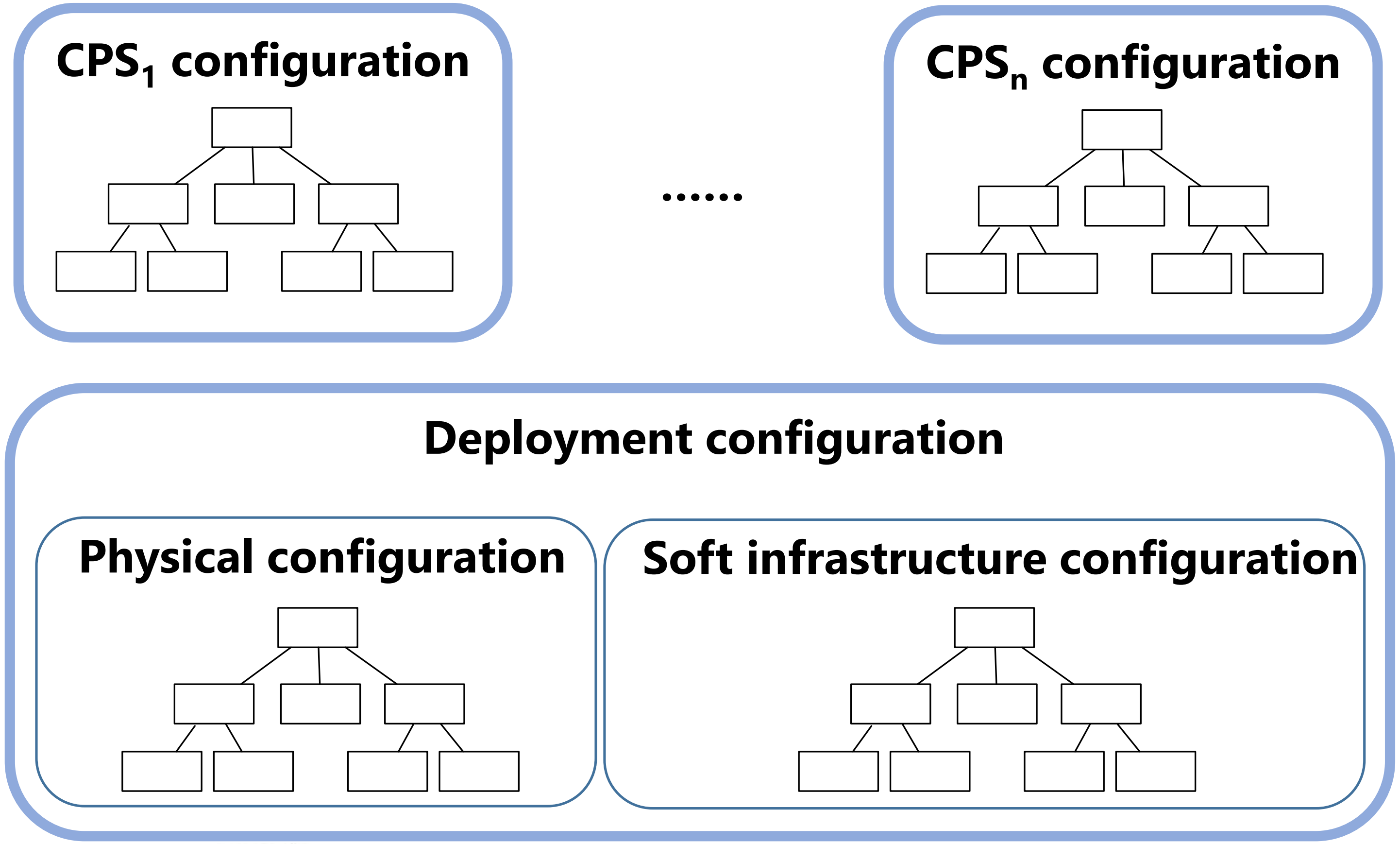
Challenges:

- **Measure QAs of every configuration is an intractable task**
- **Numerical features support of automated solvers is limited**

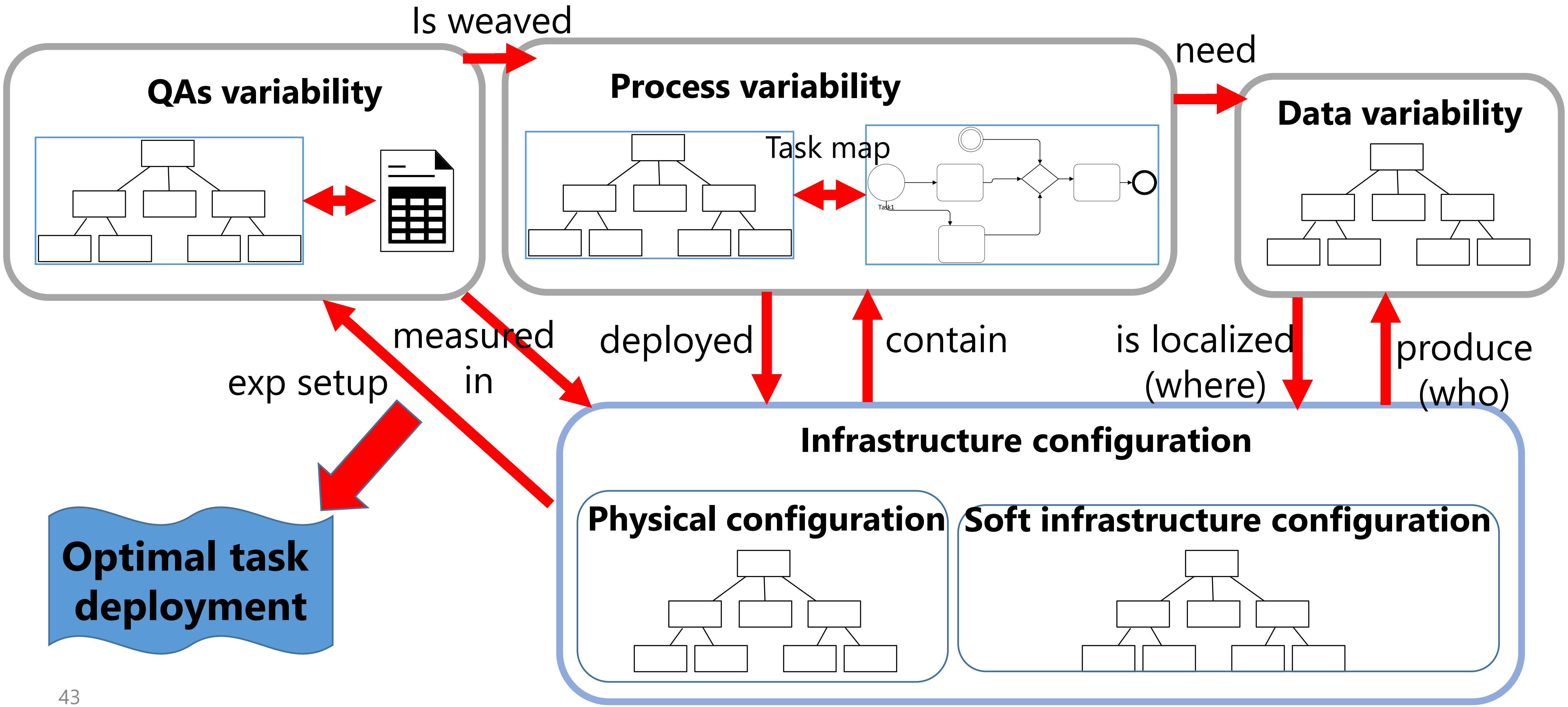
Deployment variability

Deployment variabilty

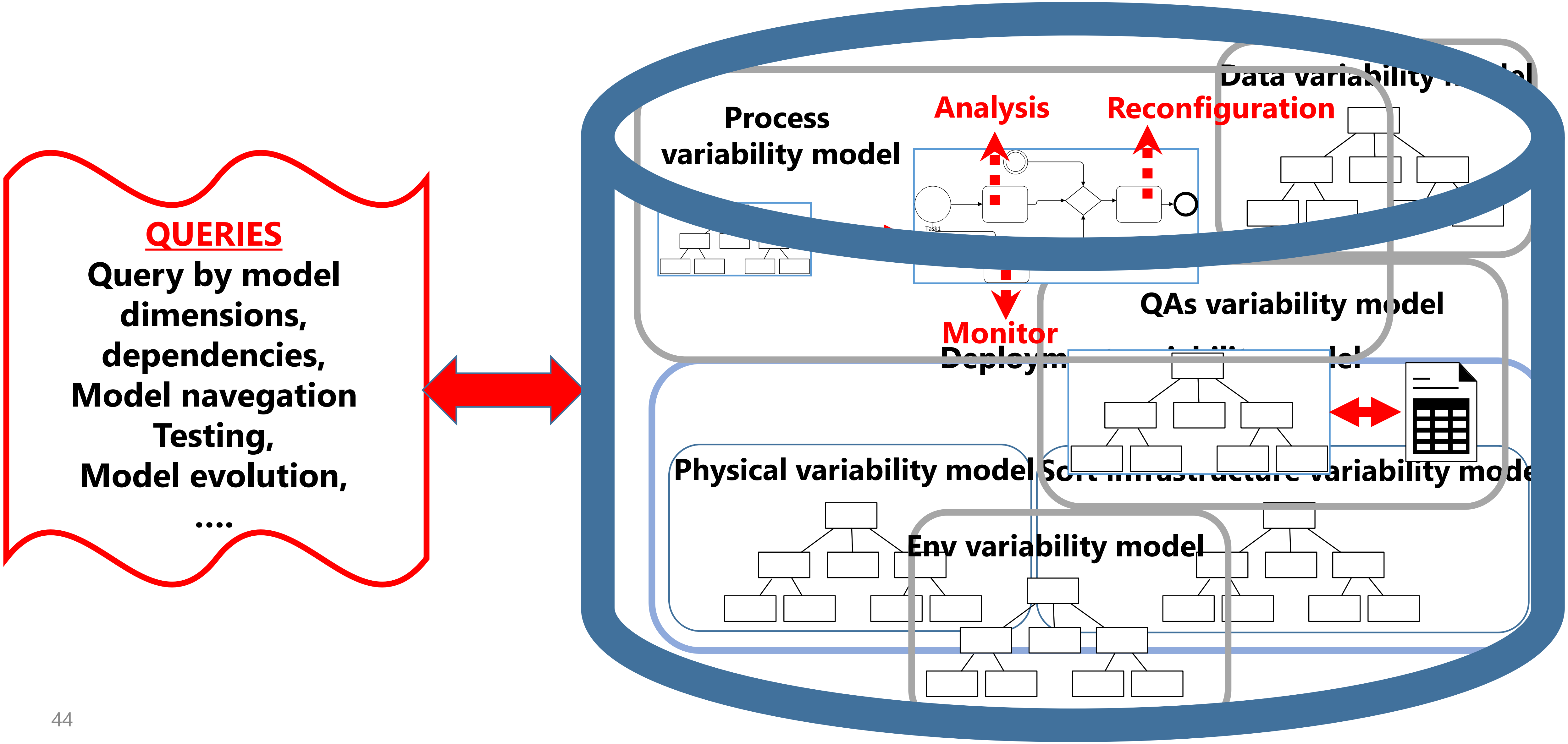
- Generate the optimum configuration of a distributed CPS
- Common resources
  - Physical configuration
  - Software infrastructure configuration



Inter-variability dimensions interactions



Variability model repository



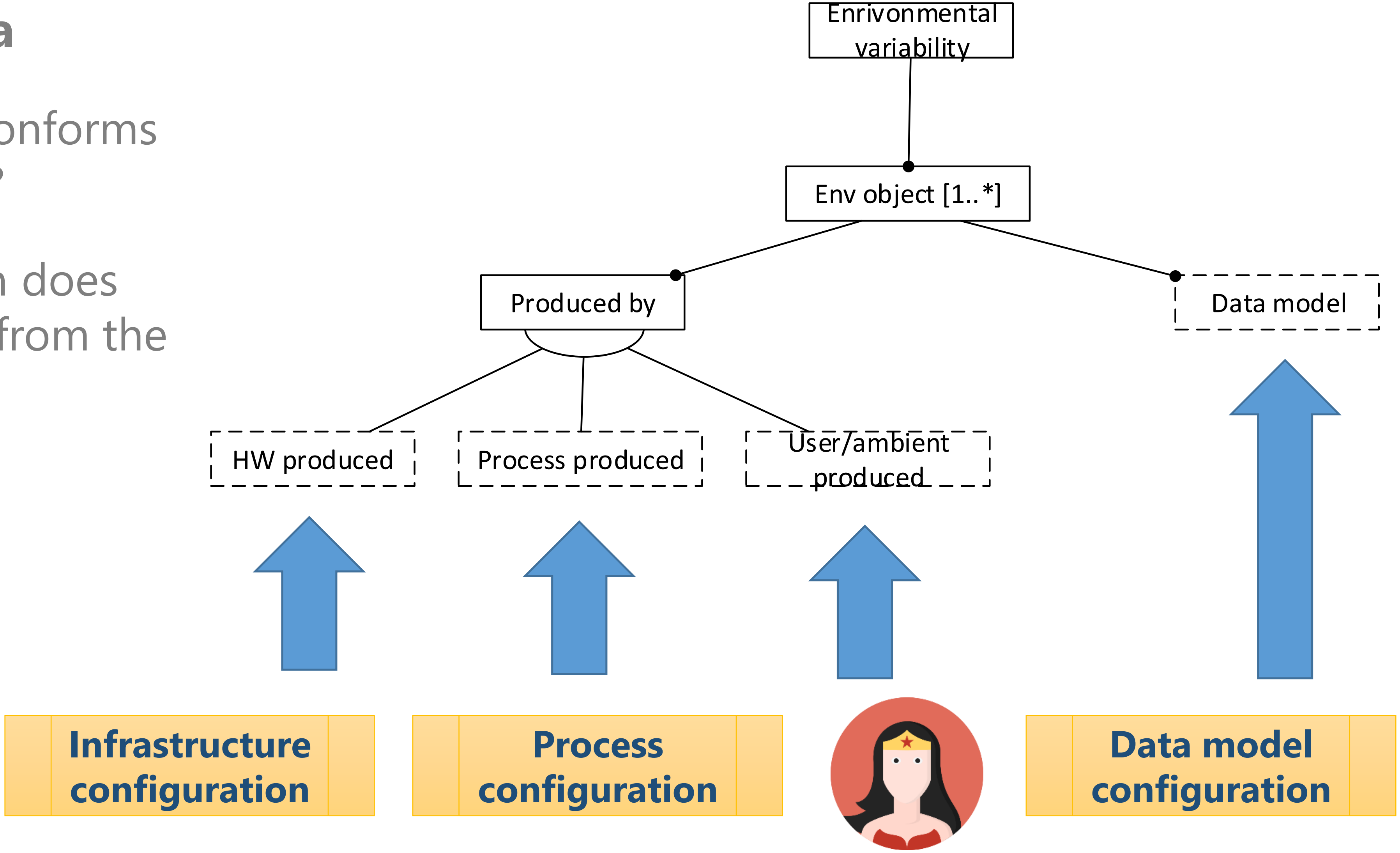
QUERIES

Query by model dimensions, dependencies,
Model navegation
Testing,
Model evolution,
....

Environment variability

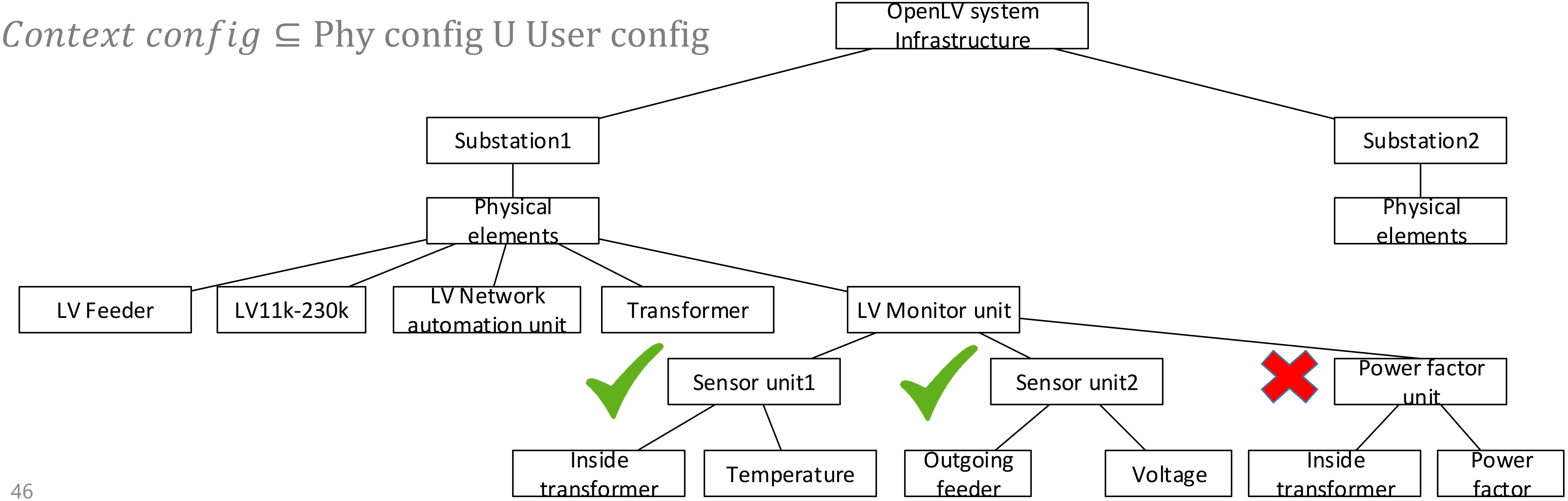
Environment variability

- **Environment data**
  - What elements conforms the environment?
  - What information does the CPS receives from the environment?



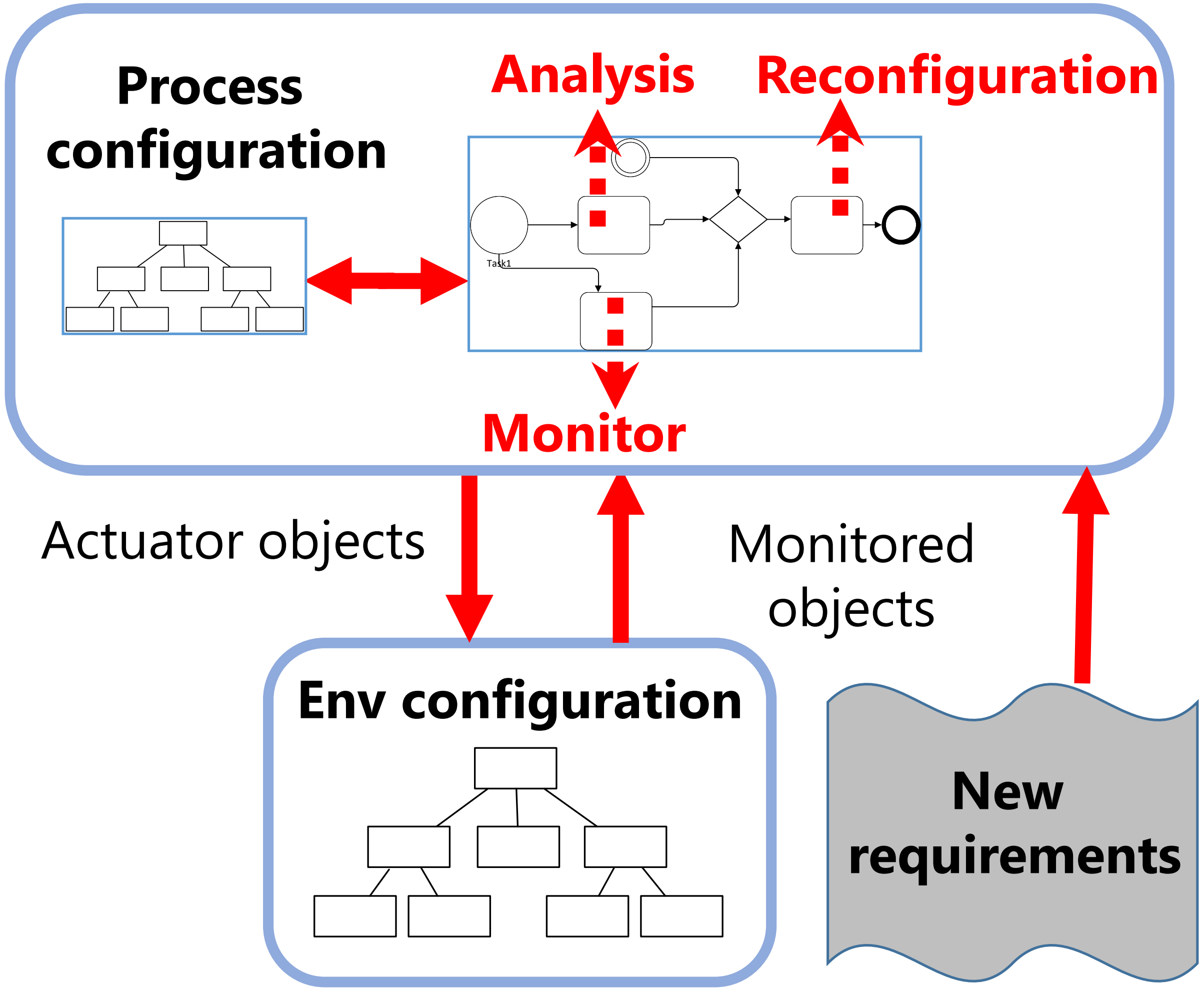
Environment variability

- Internal elements of the physical system are not part of the environment
- **Environment vs Adaptation Context**
- $Context\ config \subseteq \text{Phy config} \cup \text{User config}$



Runtime adaptation

Self-adaptation



Process configuration

Analysis

Reconfiguration

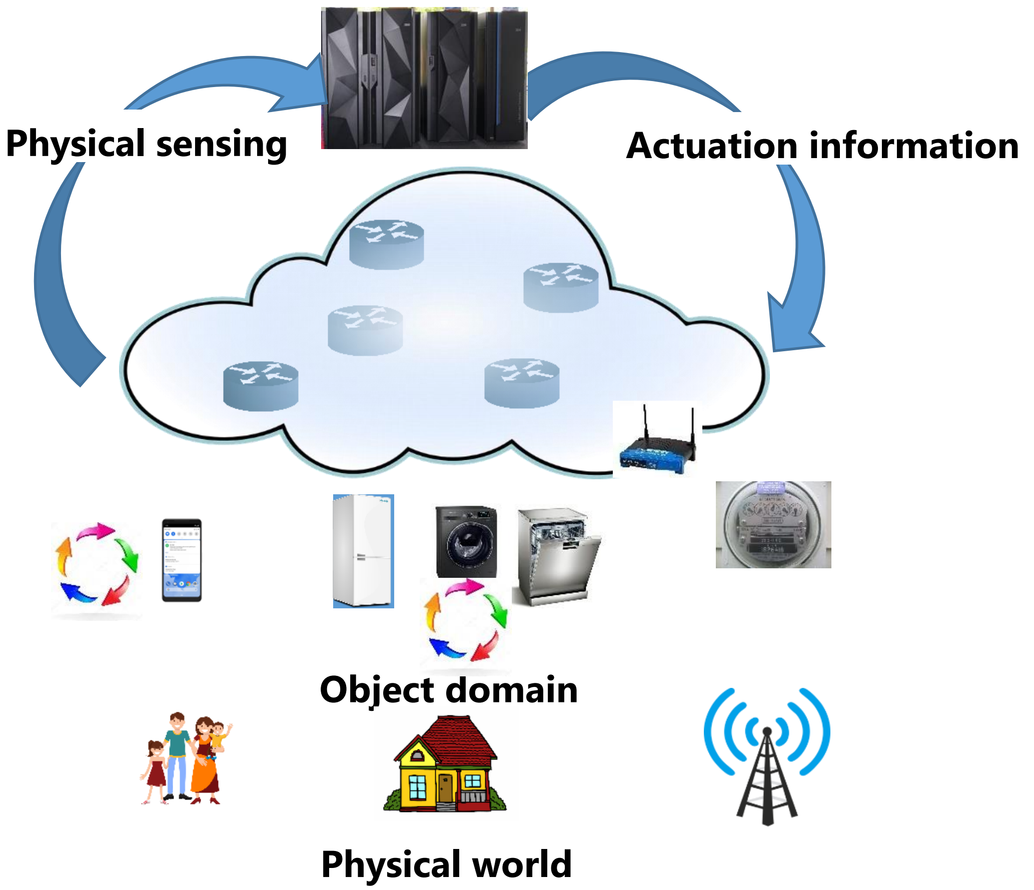
Monitor

Actuator objects

Monitored objects

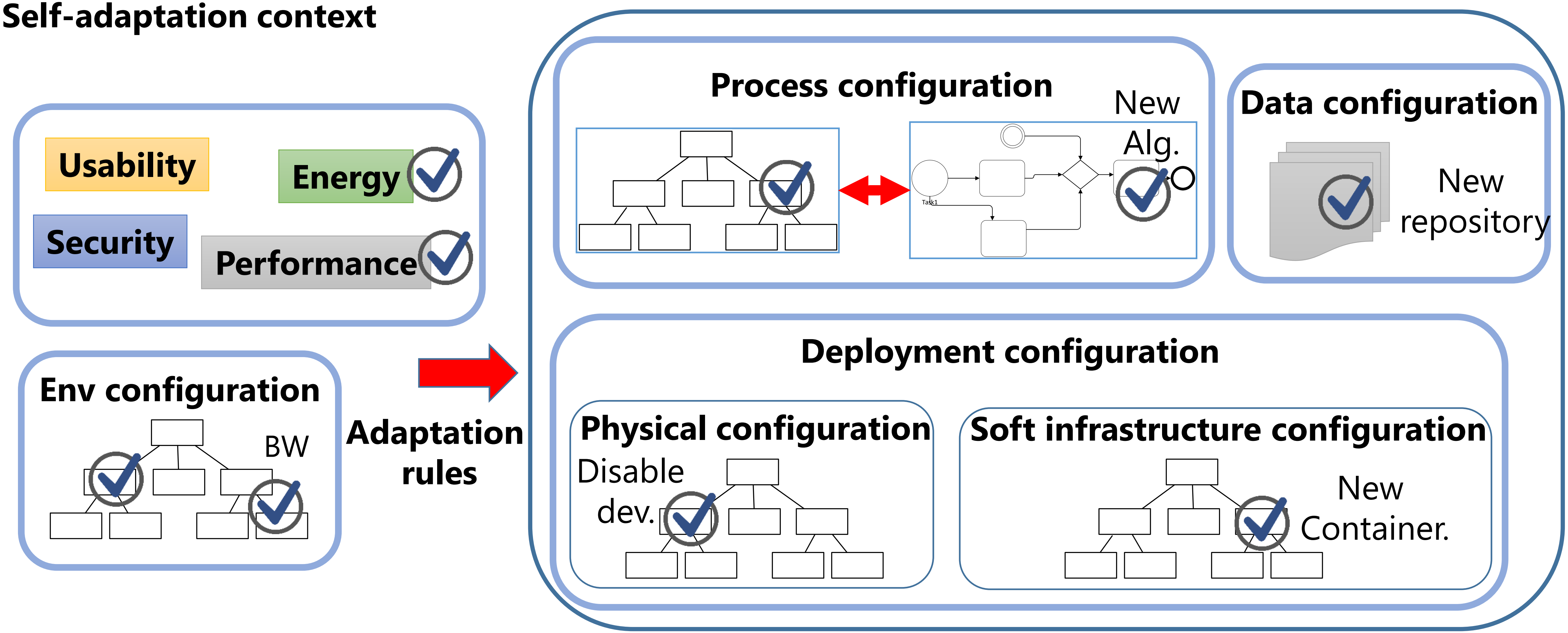
Env configuration

New requirements



Runtime model interactions

Self-adaptation context



Usability

Energy ✓

Security

Performance ✓

Env configuration

BW

Adaptation rules

Process configuration

New Alg.

Data configuration

New repository

Deployment configuration

Physical configuration

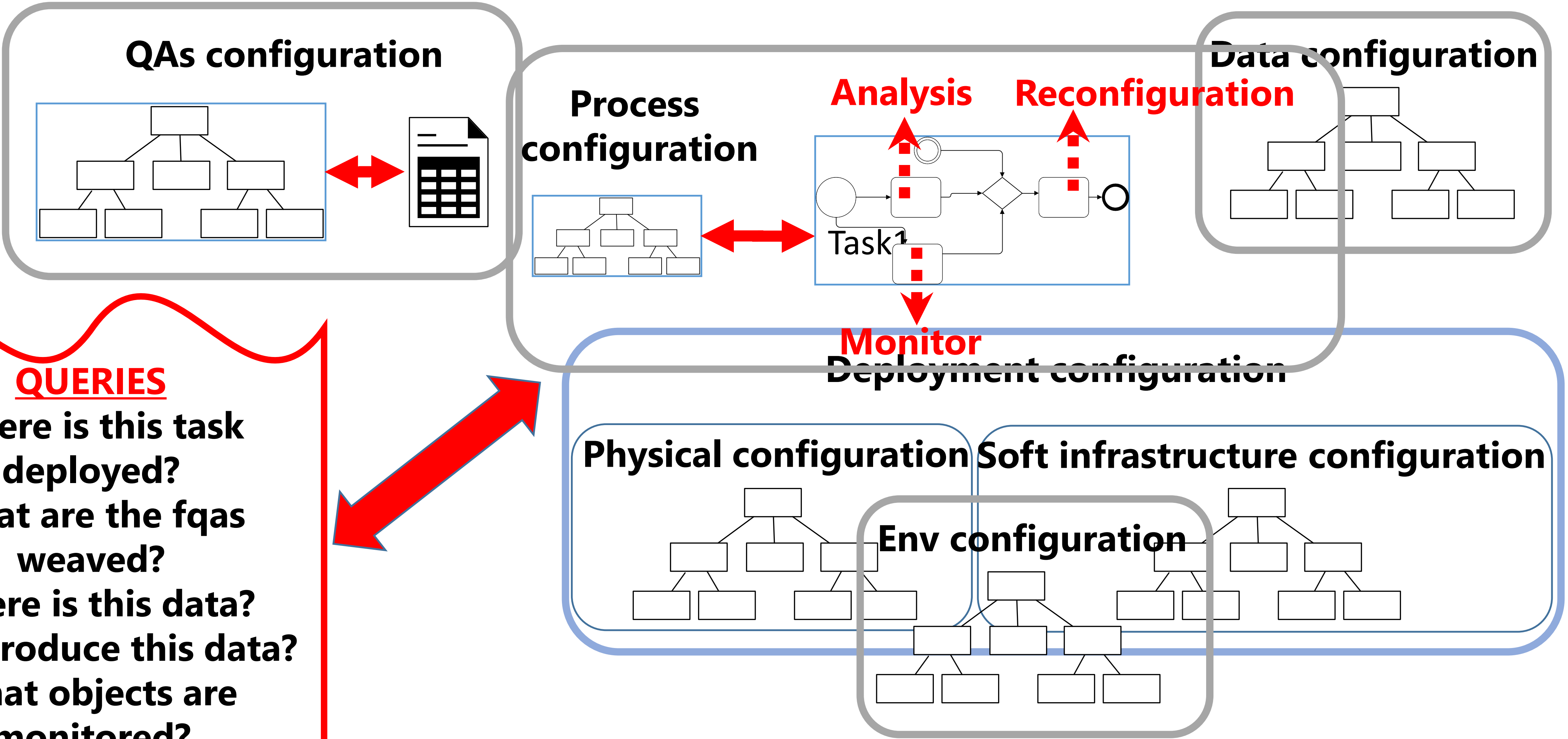
Disable dev.

Soft infrastructure configuration

New Container.

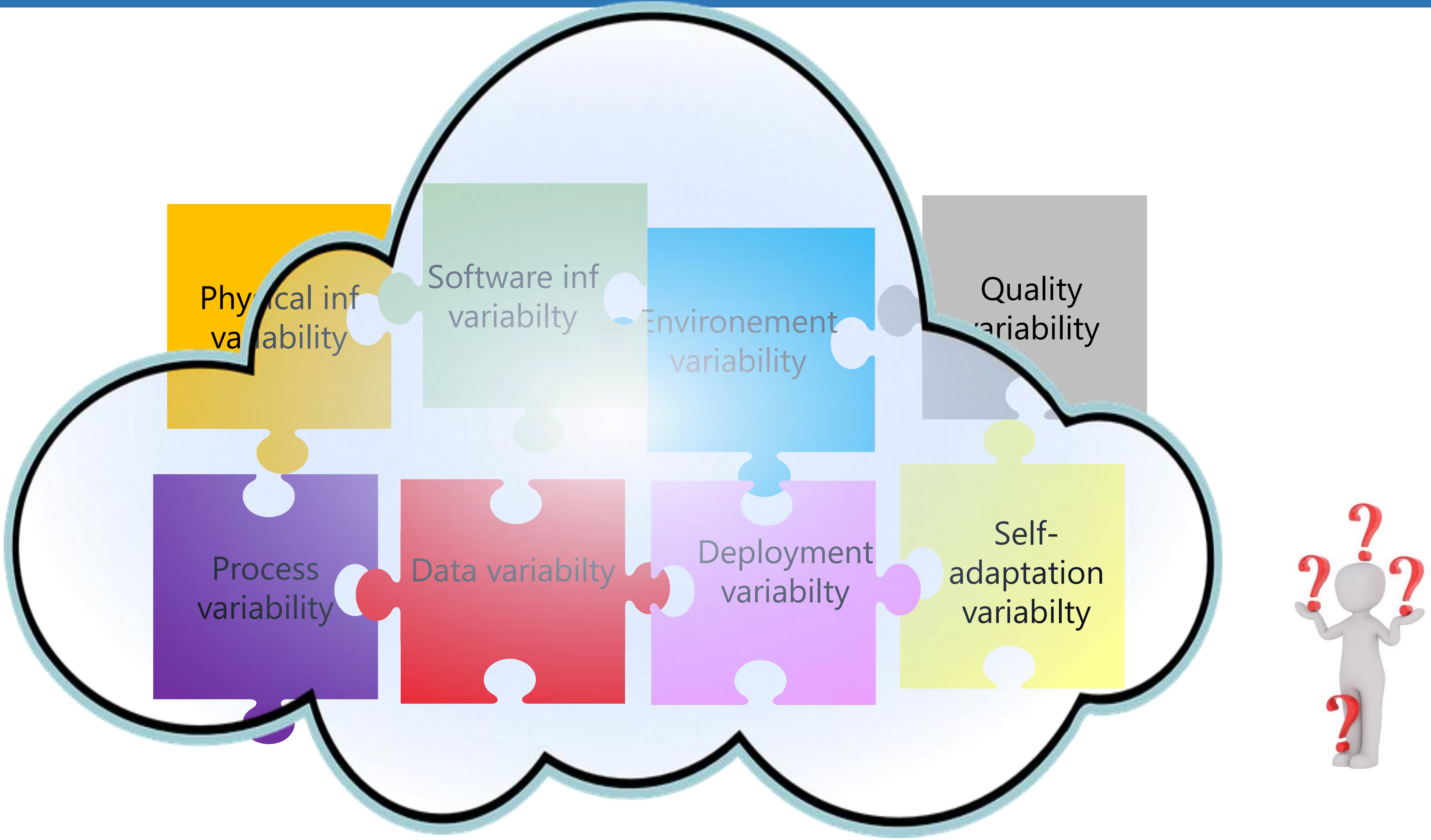
Dynamic Software Product Lines

Runtime model queries



QUERIES
Where is this task deployed?
What are the fqas weaved?
Where is this data?
Who produce this data?
What objects are monitored?

Solve the puzzle



Future challenges

- Define all kinds of variabilities of CPSs
- Separate if posible in different variability models
- Add semantic to those models
- Define formal relationships inter-models, and inter-configurations
- Store models in a repository and define evolution, navigation, ….
- Define repositories containing QAs data
- Reduce the number of configurations to measure QAs with numerical features
- Define advanced query operators

Main references

- An automatic process for weaving functional quality attributes using a software product line approach. Journal of Systems and Software, 112: 78-95 (2016)
- Variability models for generating efficient configurations of functional quality attributes. Information & Software Technology 95: 147-164 (2018)
- Energy-aware environments for the development of green applications for cyber-physical systems. Future Generation Comp. Syst. 91: 536-554 (2019)
- Context-aware energy-efficient applications for cyber-physical systems. Ad Hoc Networks 82: 15-30 (2019)
- Uniform random sampling product configurations of feature models that have numerical features. SPLC (A) 2019: 39:1-39:13

THANKS