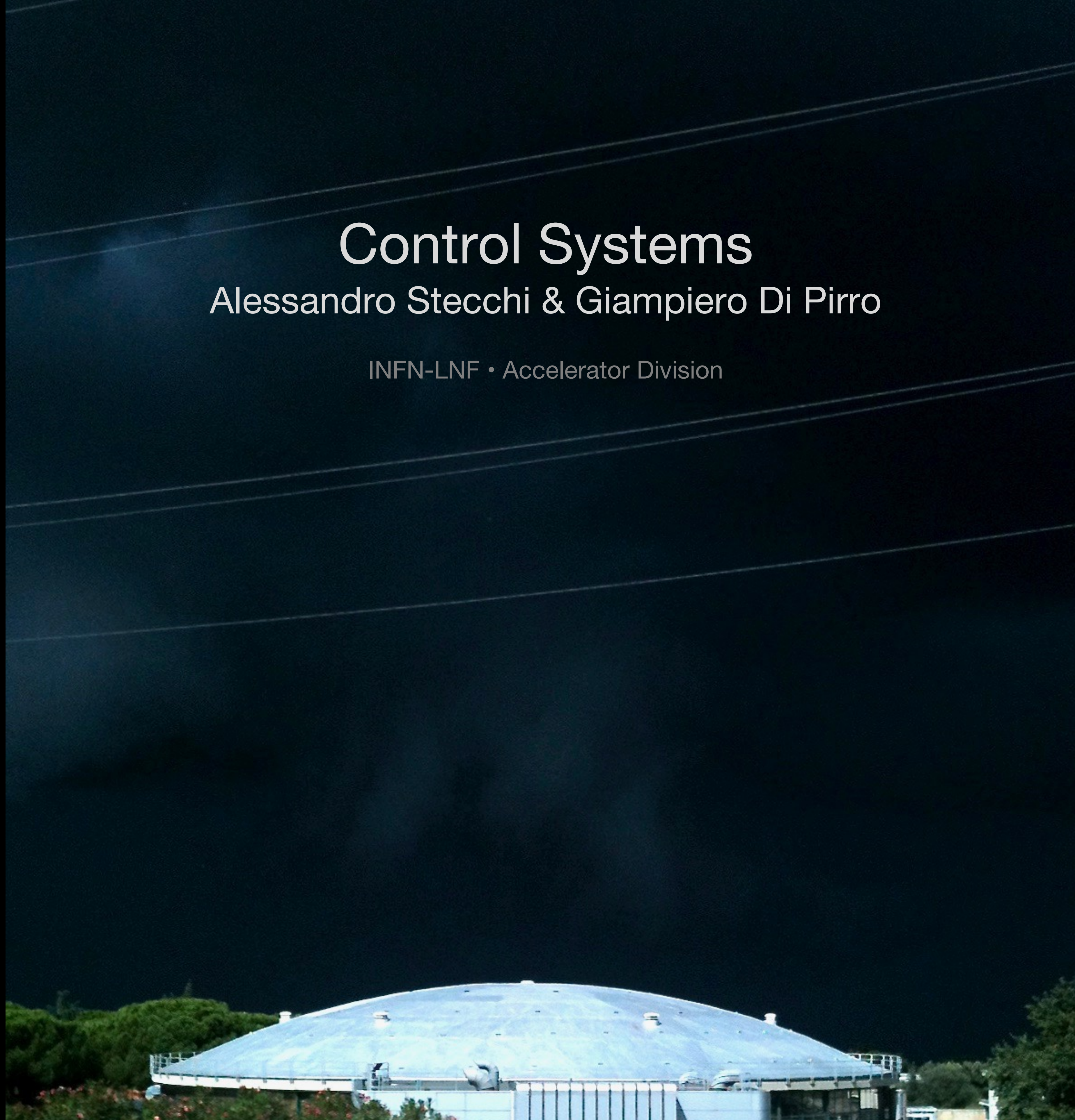
Control Systems

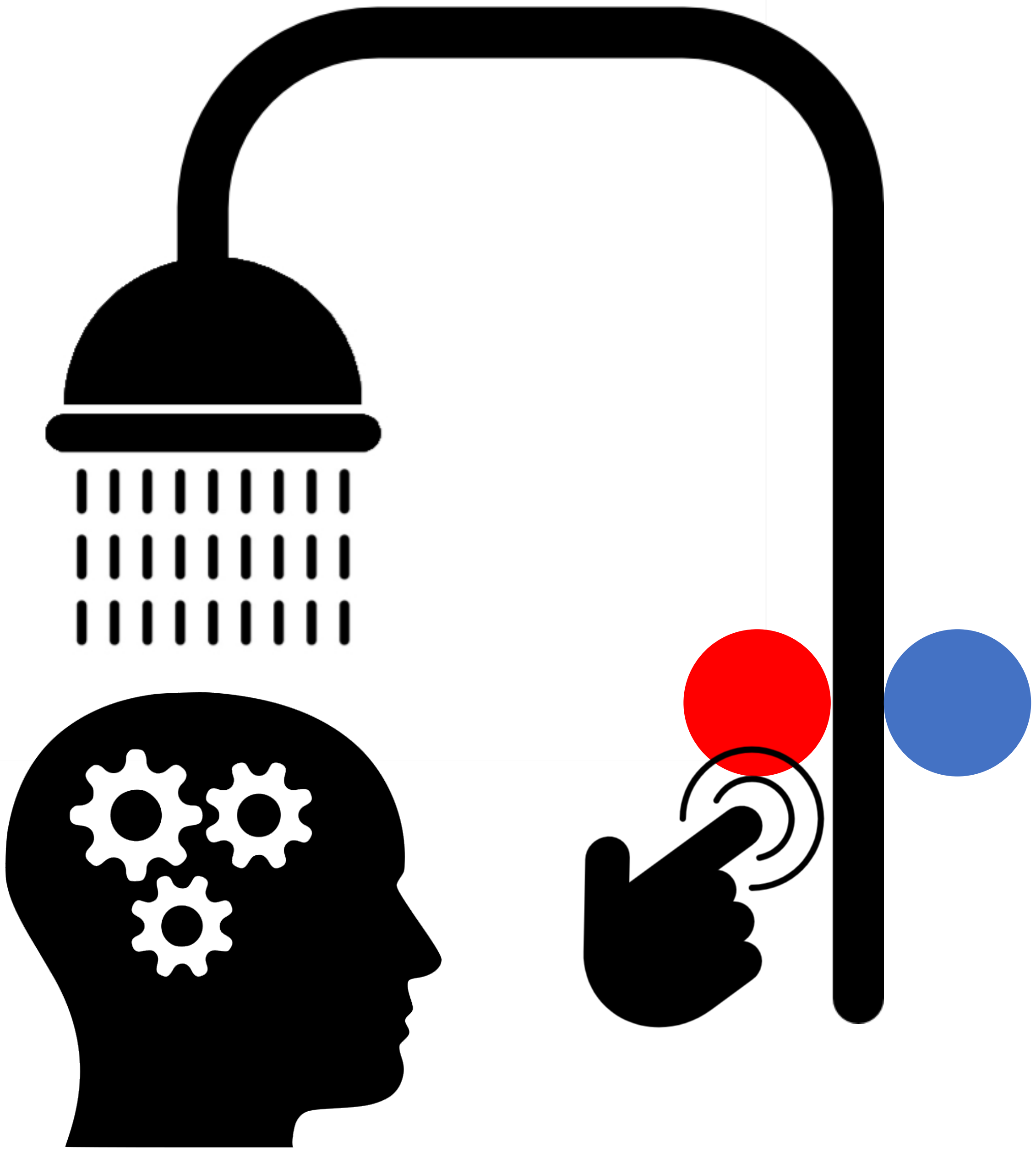
Alessandro Stecchi & Giampiero Di Pirro

INFN-LNF • Accelerator Division



Video courtesy of SpaceX

A Control System is a system that allows the **output variables** of a **process** to be varied (or kept constant) according to **predetermined laws**.

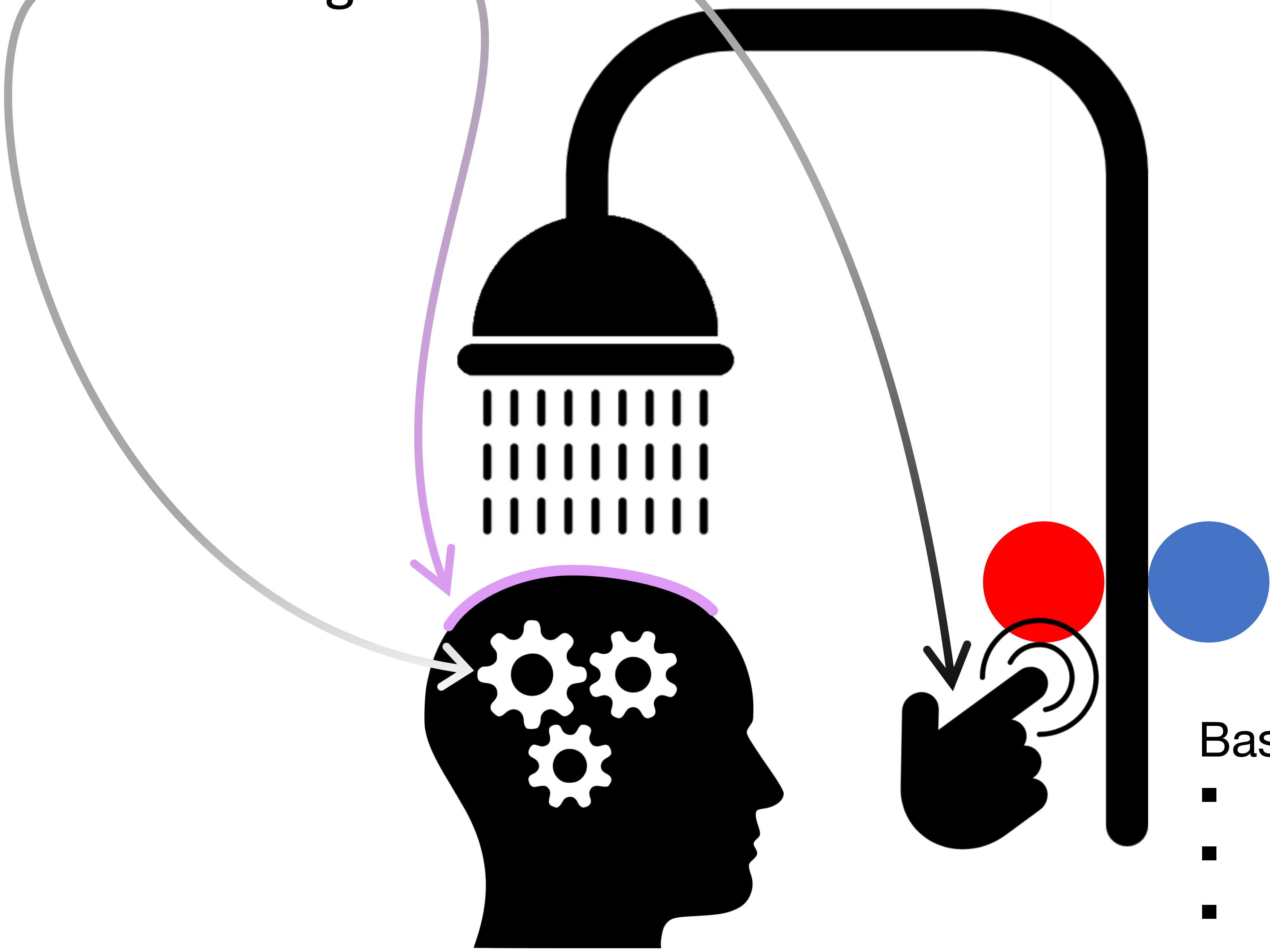


Open loop
Action is based on an algorithm and initial **static information**.

Closed loop
Action is based on an algorithm and **dynamic information** continuously read back from the controlled system.

Key elements of a Control System:

- actuators
- sensors
- control algorithms



Basic functions

- acquisition
- actuation
- automatic adjustment

Fields of application of control systems

Civil, Social

- home automation
- automotive, aeronautics
- building climate control (ESCO: Energy Service Company)
- agriculture
- medical equipment, radiotherapy, diagnostics
- telecommunications

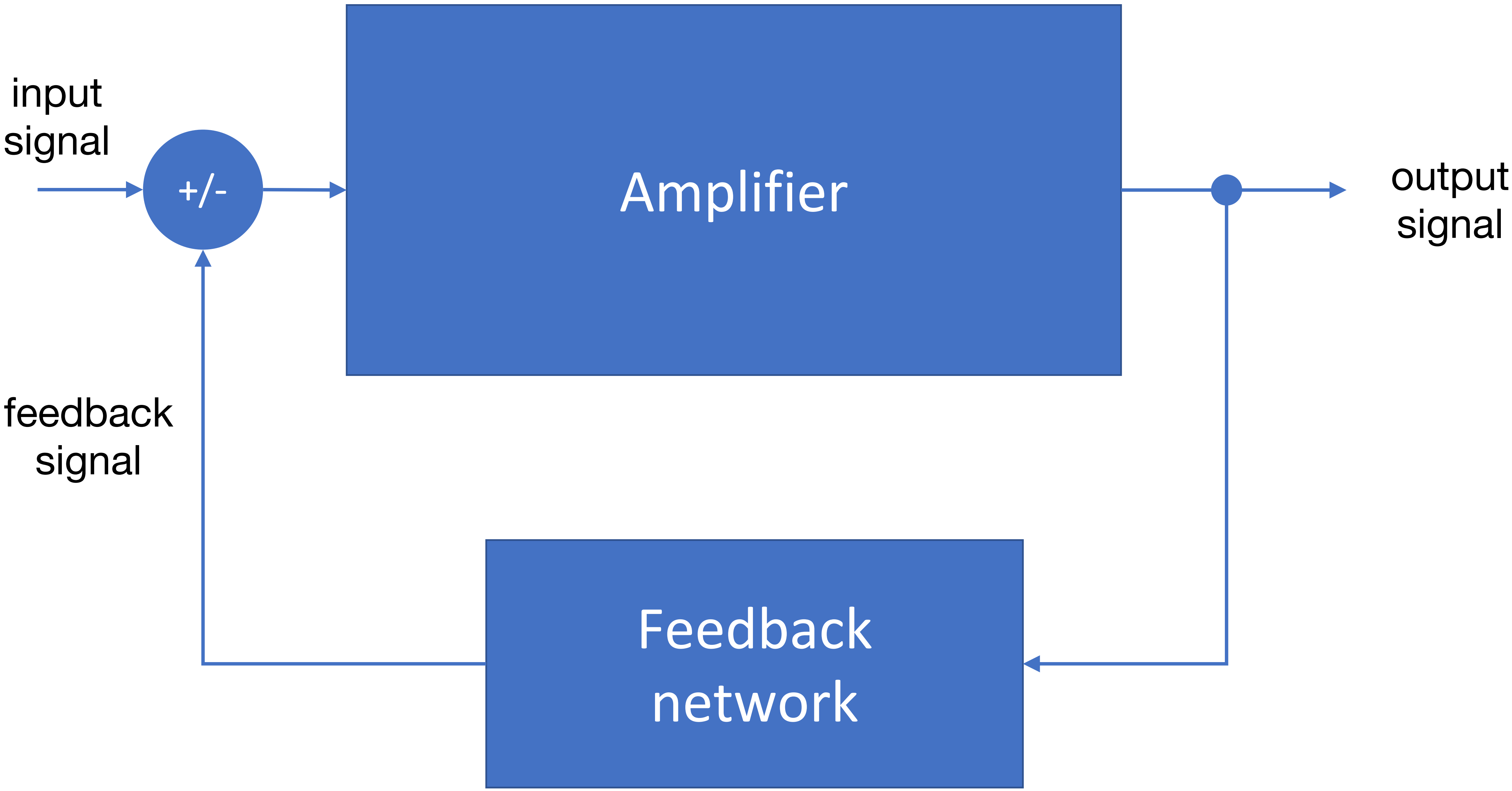
Industrial

- production lines (robots)
- prognostics
- power plants (thermal, electrical, nuclear)

Scientific

- unmanaged rovers, space probes
- astronomy
- experimental apparatus, detectors, particle accelerators

What physicists and engineers usually mean by *Control System*

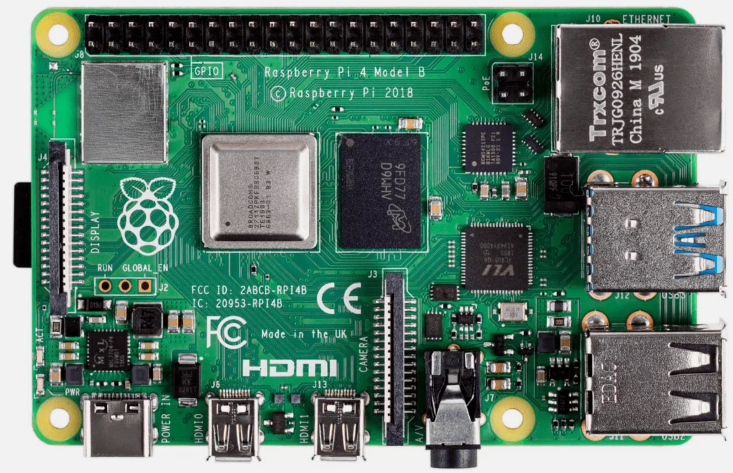


Controls adopting "CPUs"

Computers t ~ 1-10 ms

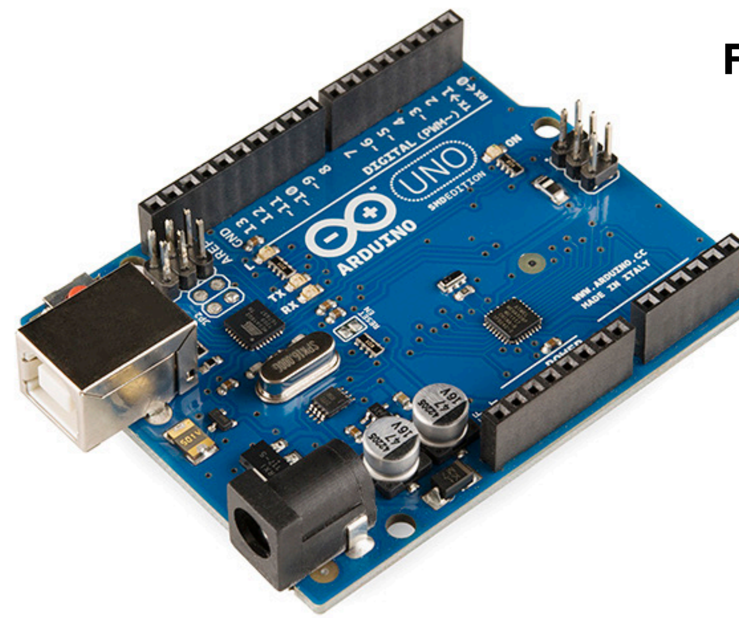
PC

SBC: Single Board Computer



Embedded processor

FPGA: Field Programmable Gate Array
t ~ ns



uP: microprocessor
t ~ 1 ms

ASIC: Application-Specific Integrated Circuit

Fields of application of **our** control systems

Civil, Social

- home automation
- automotive, aeronautics
- building climate control (ESCO: Energy Service Company)
- agriculture
- medical equipment, radiotherapy, diagnostics
- telecommunications

Industrial

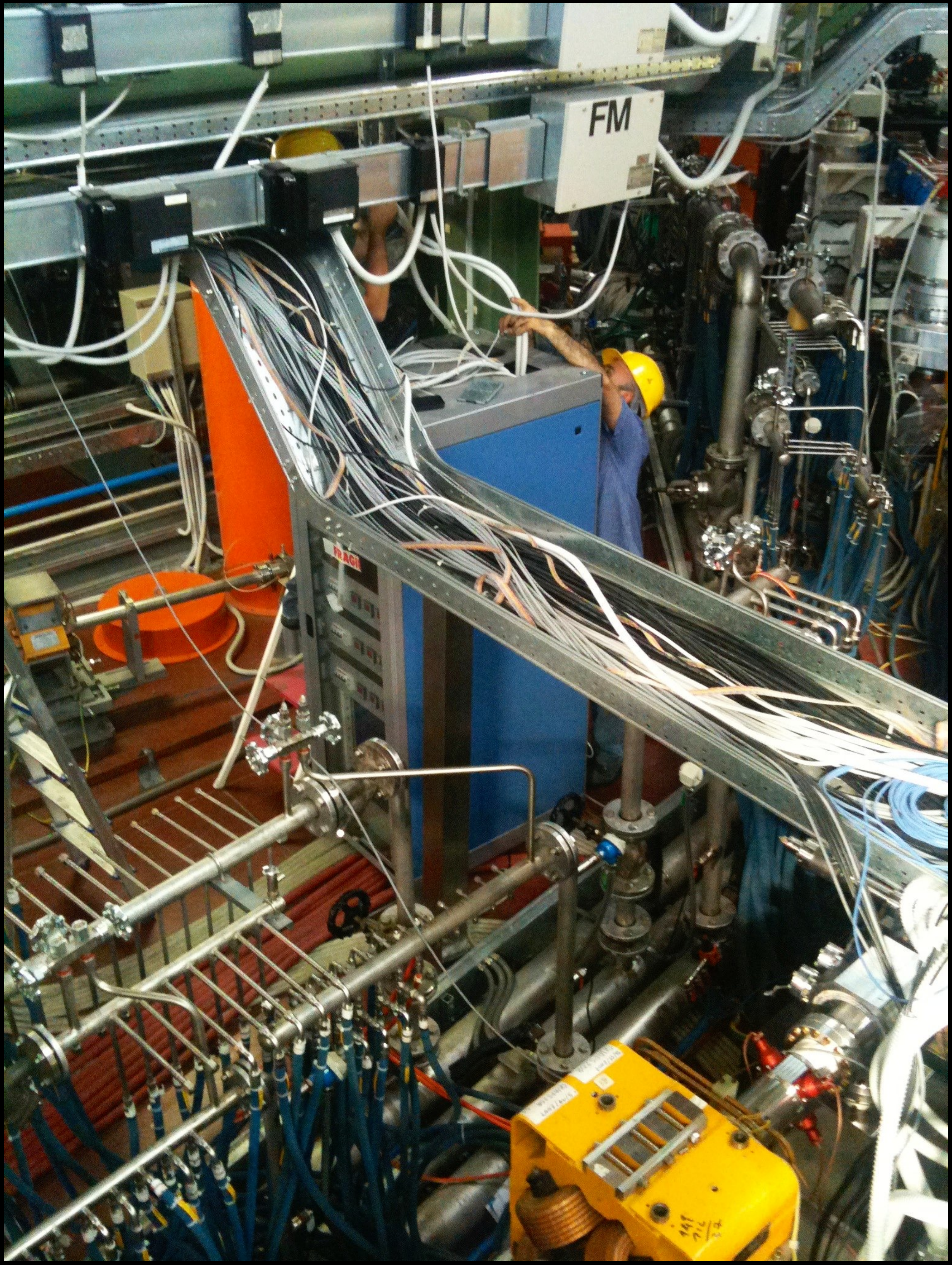
- production lines (robots)
- prognostics
- thermal, electrical, nuclear power plants

Scientific

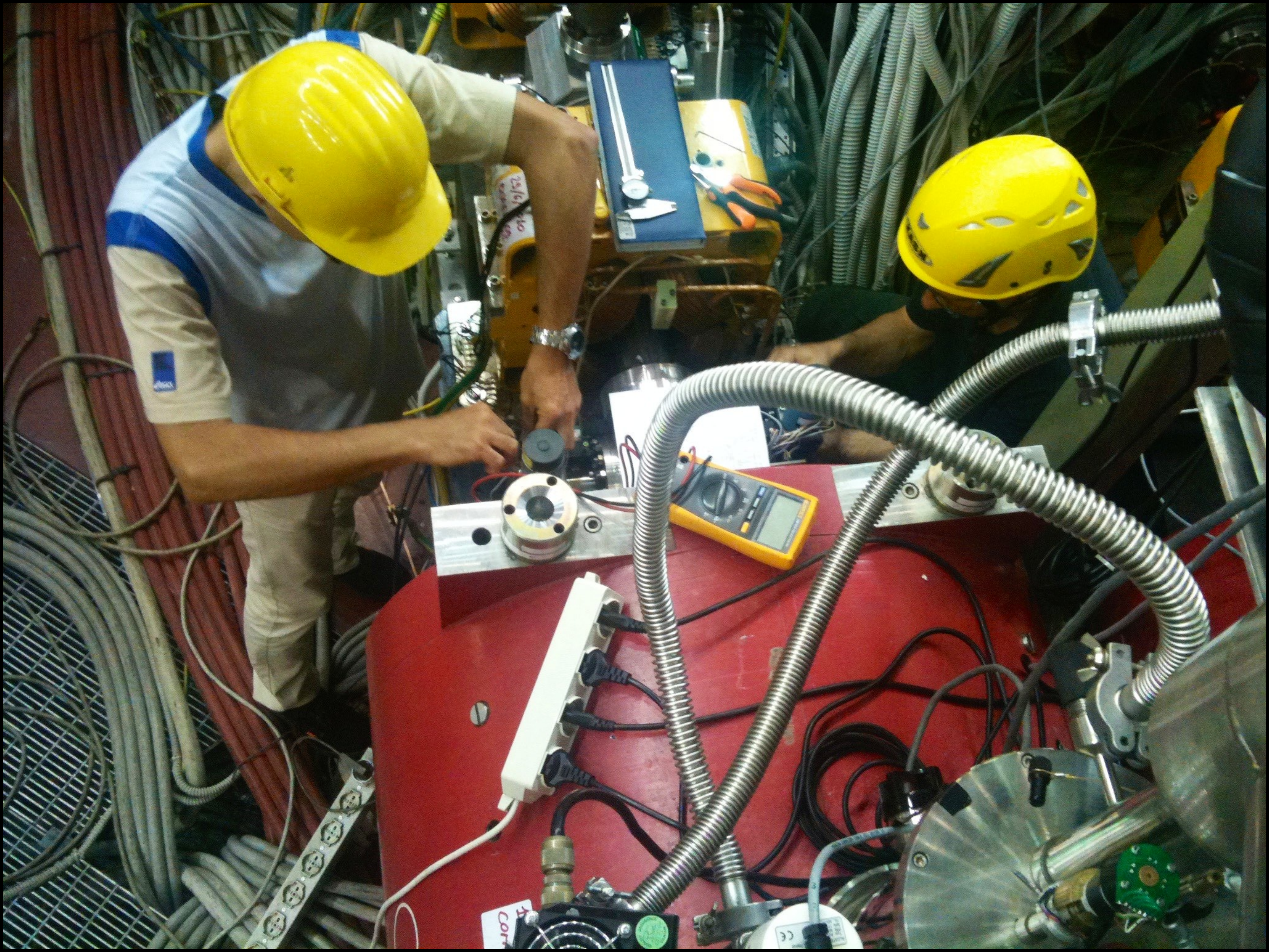
- automatic rovers, space probes
- astronomy
- **experimental apparatus, detectors, particle accelerators**

Control Systems – of complex plants – employing computers (*and IT in general*)

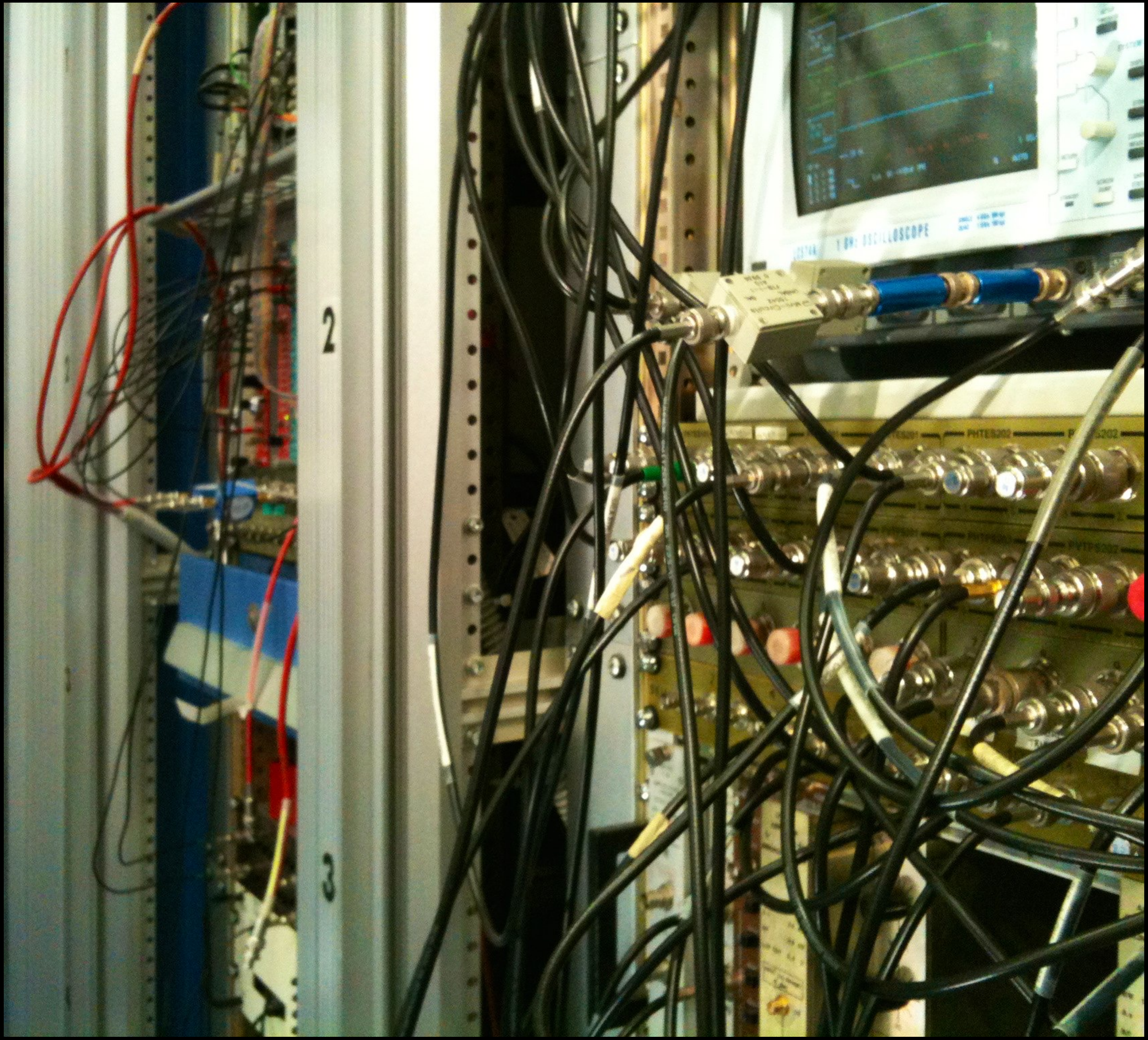
plant



plant → sensors/actuators



plant → sensors/actuators → front-end



plant → sensors/actuators → front-end → control room

Design and Implementation

Definition of specifications and objectives to achieve

- performance
- features
- reliability
- assessment of complexity and load estimation (processing power, throughput, latency, storage)

System modeling

- design of the control framework or adoption of an existing one (e.g. EPICS, TANGO,...)

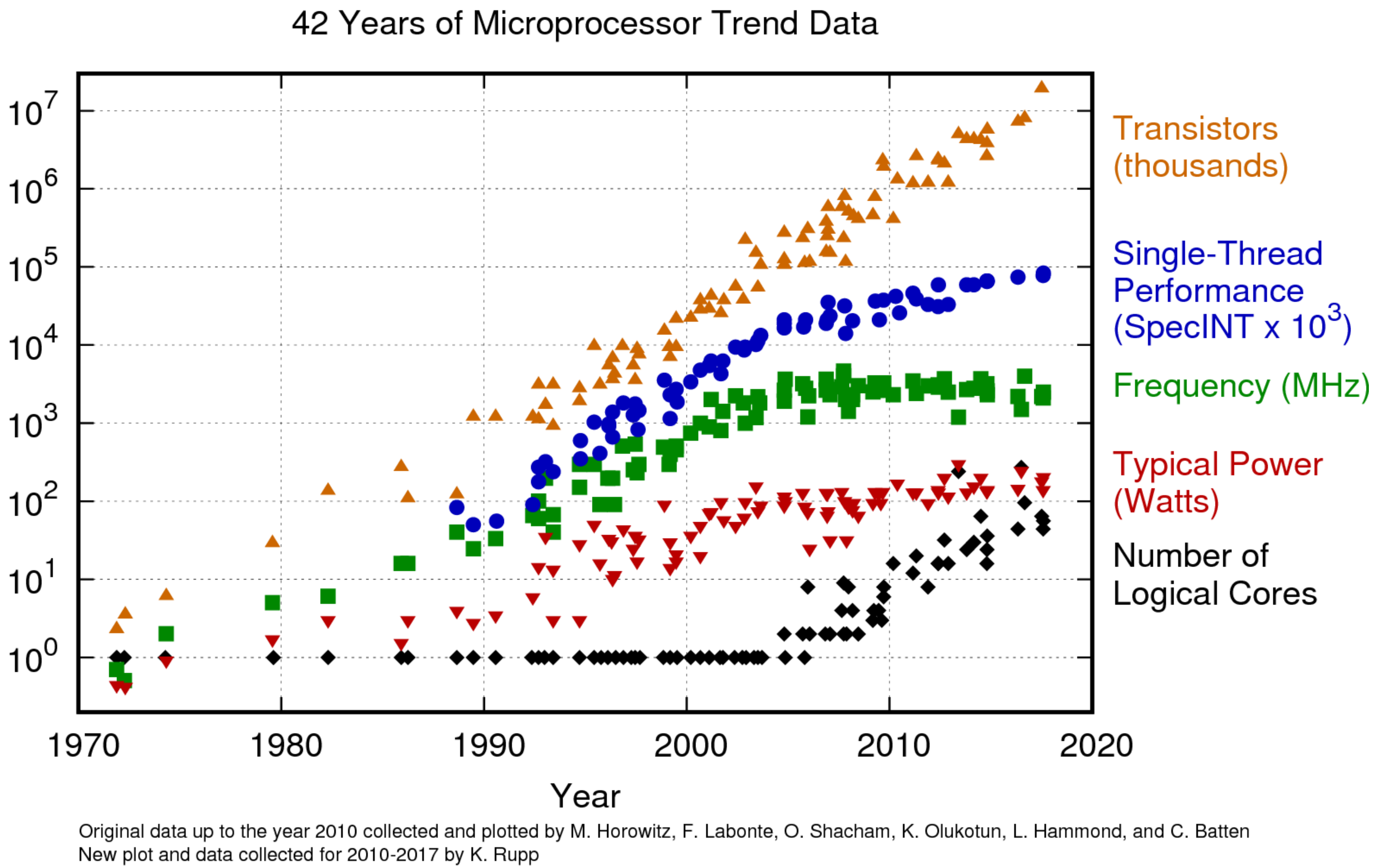
Development

- OOD, OOP (properties, methods)
- software development
- simulation
- engineering
- validation

General characteristics

- **open** to other systems (experiments, DAQ, machine protection, service plants);
- **easy** for the users (ergonomic UI, data import/export, custom software interfacing);
- **scalable** according to plant size and required performance;
- **reliable** (no SPF, redundancy);
- **flexible** with respect to technological updates.

Technological growth

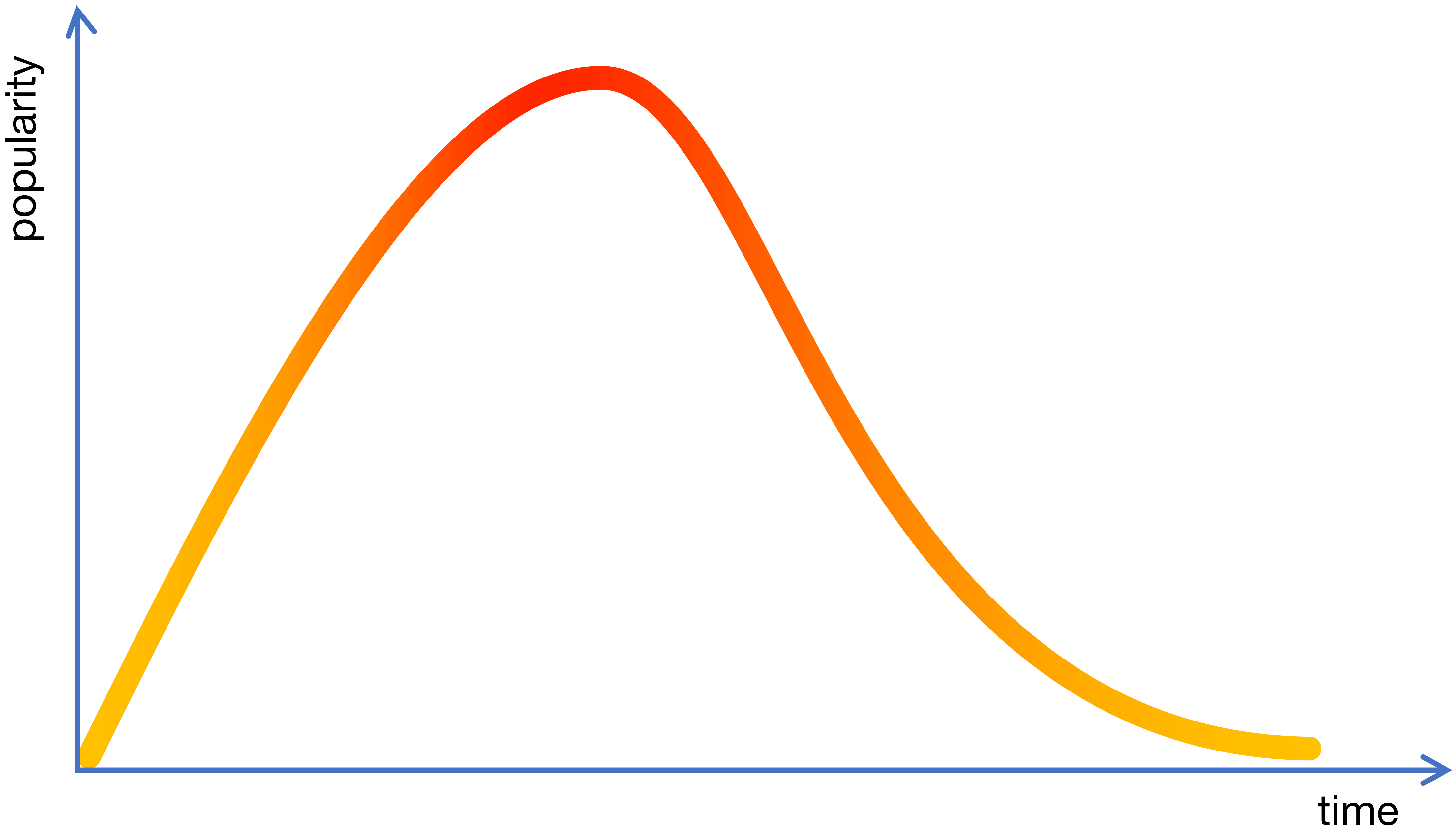


Moore's law

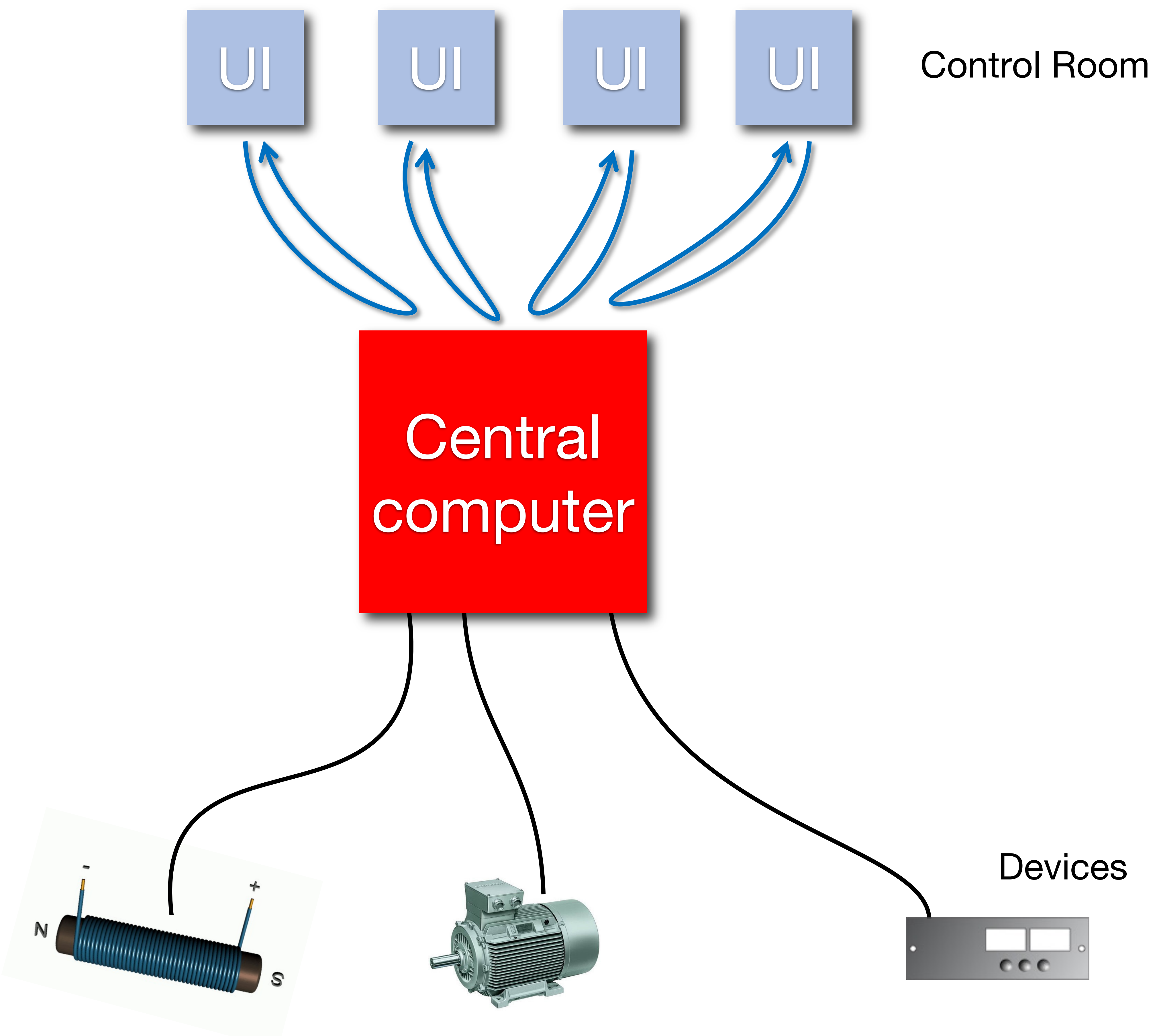
the number of transistors in processors doubles every 18 months

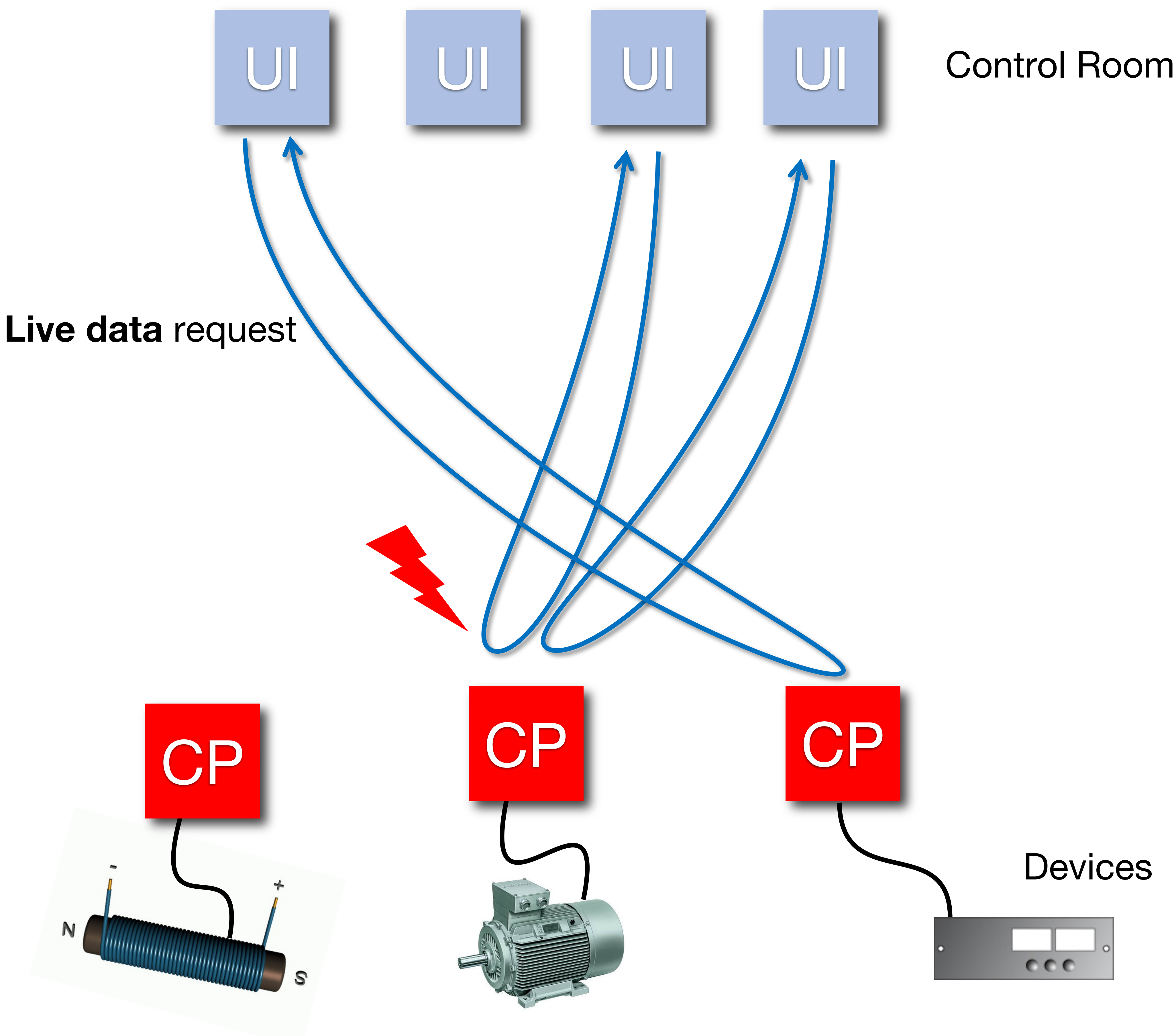
-> limits on reducing the size of transistors below which unwanted 'parasitic' quantum effects would occur in electronic circuits.

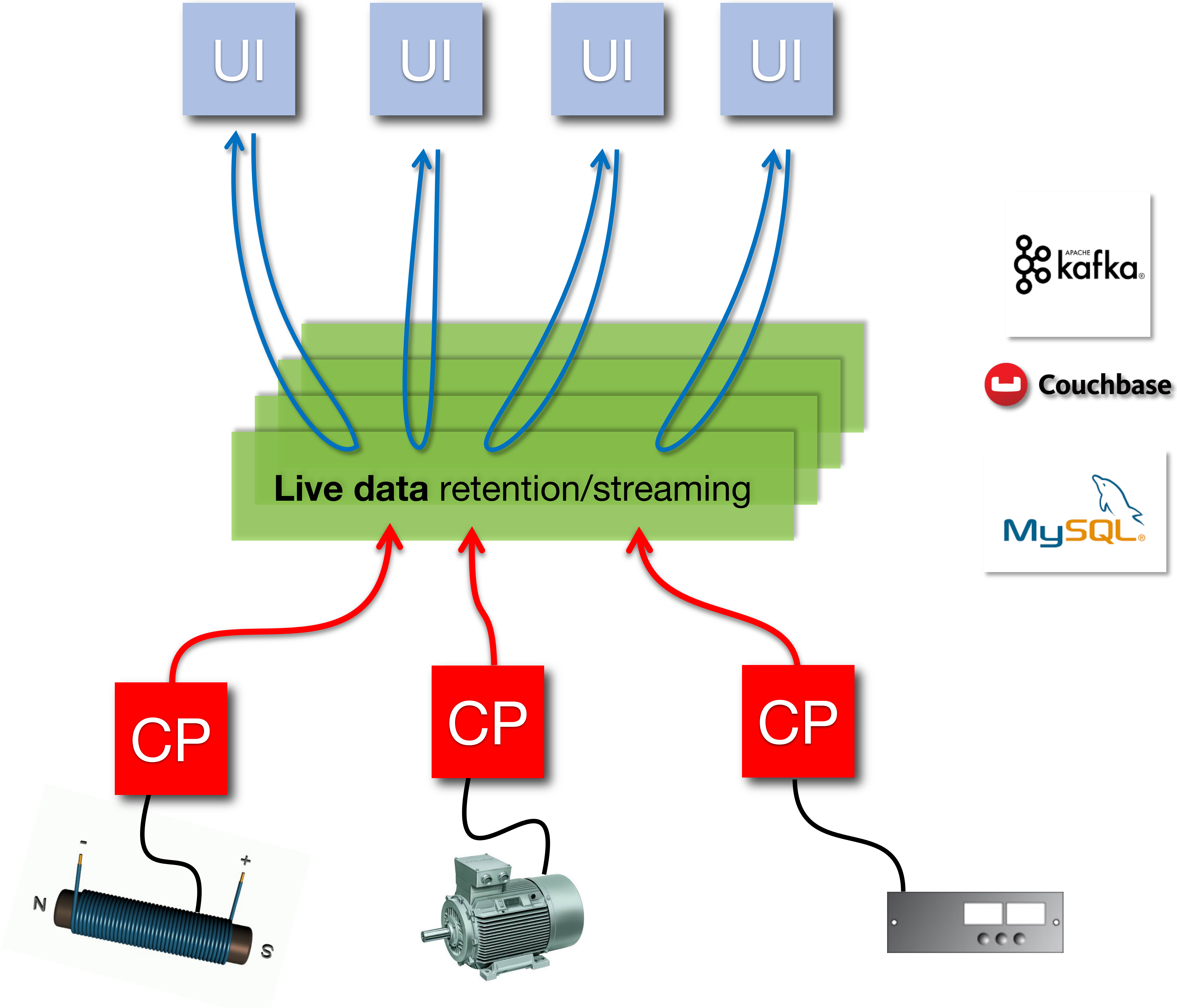
Obsolescence of standards



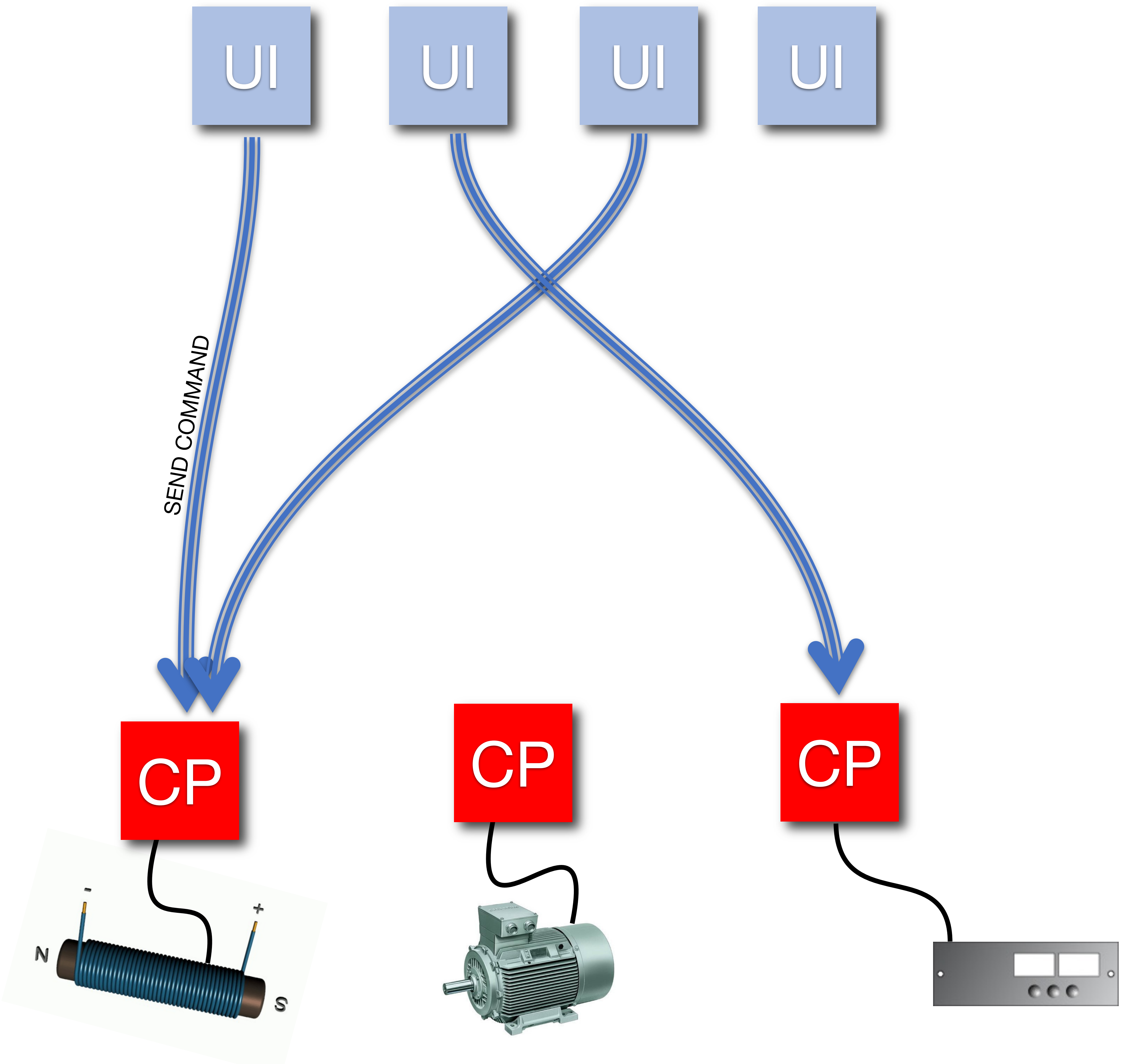
The lesson is: a system has to be as **independent** as possible from hardware standards and specific software products.

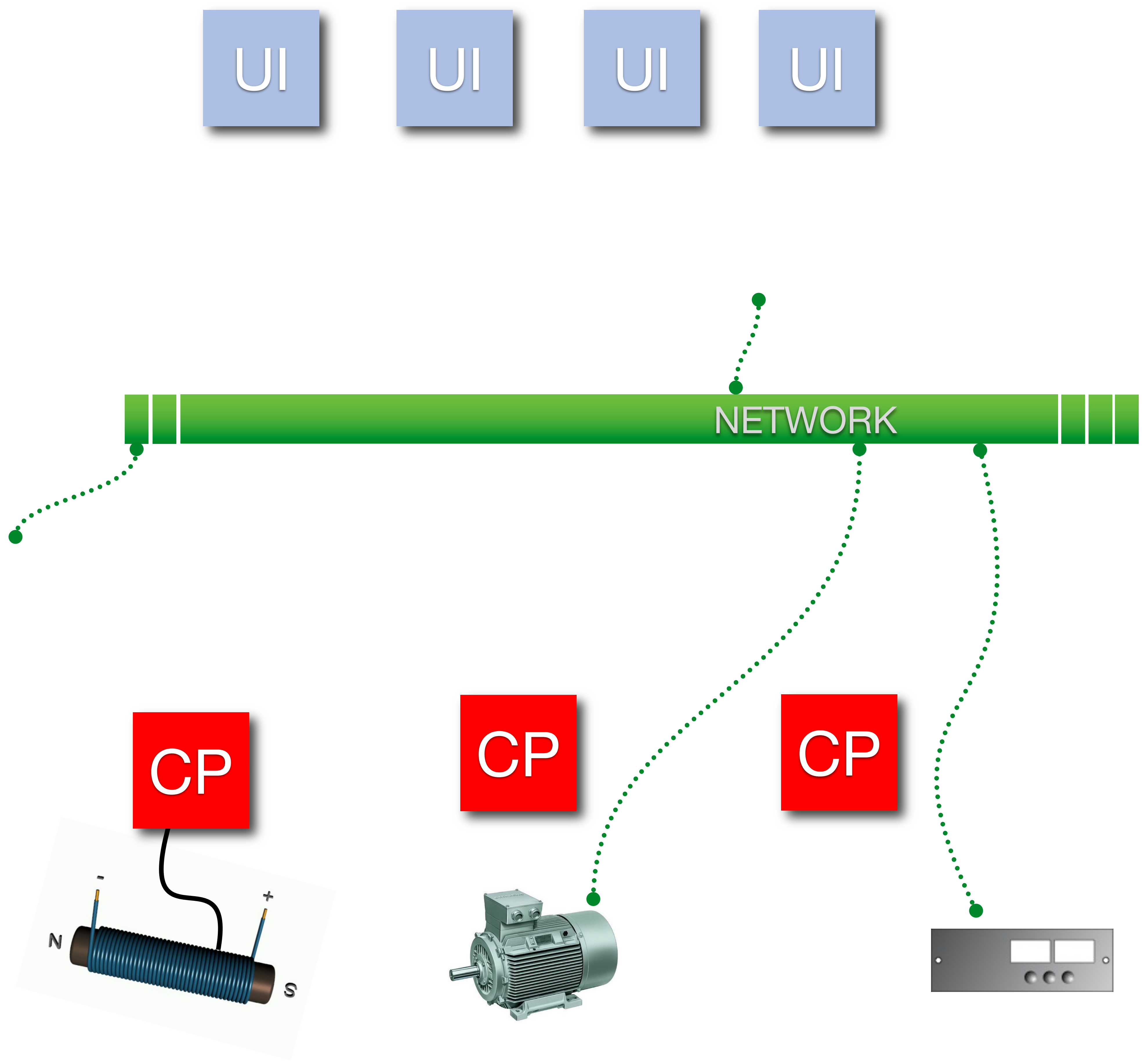

UI
UI
UI
UI
Control Room
Central computer
N
S
-
+
Devices


UI
UI
UI
UI
Control Room
Live data request
CP
CP
CP
N
S
-
+
Devices



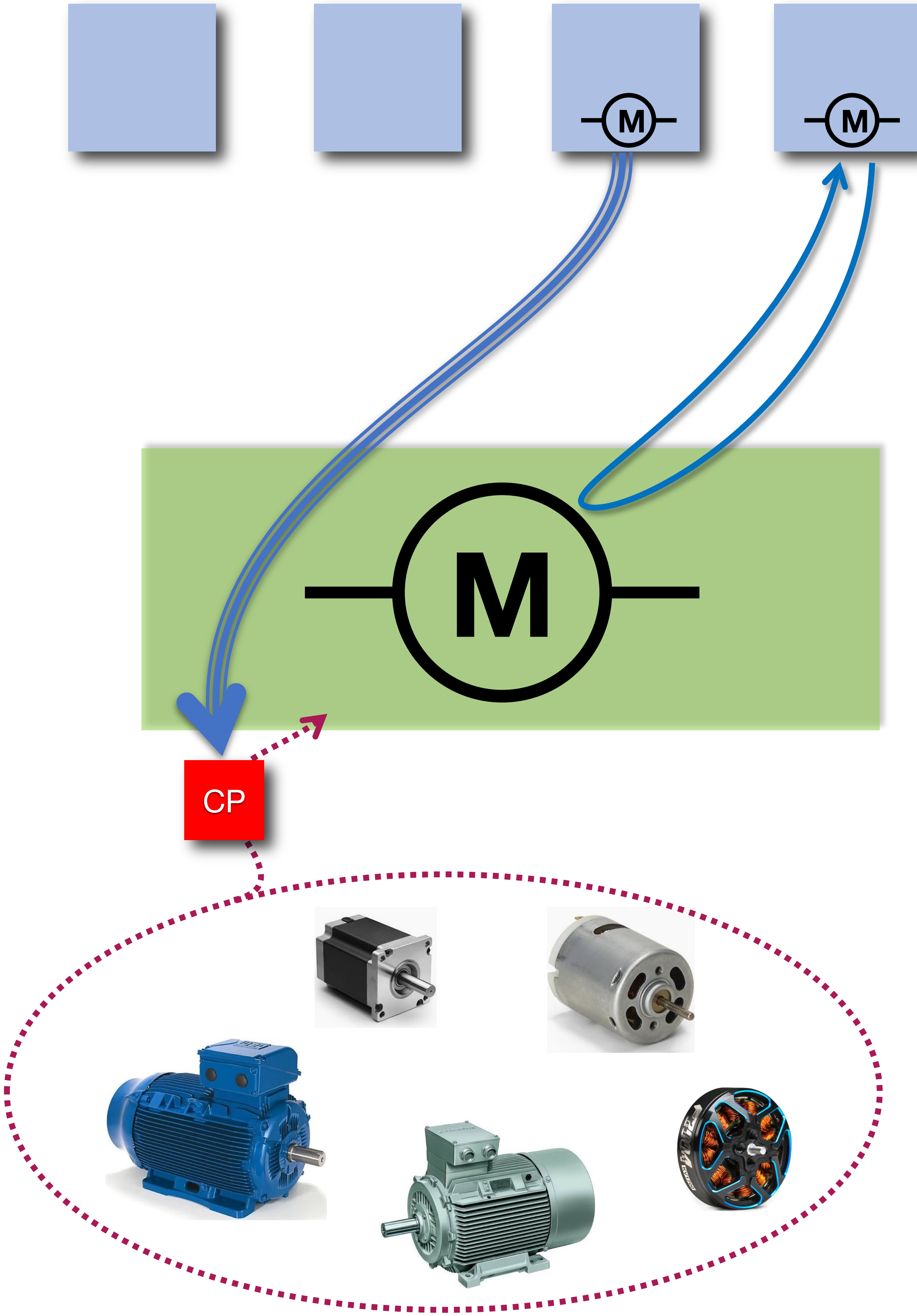
UI
UI
UI
UI
Live data retention/streaming
kafka
Couchbase
MySQL
CP
CP
CP

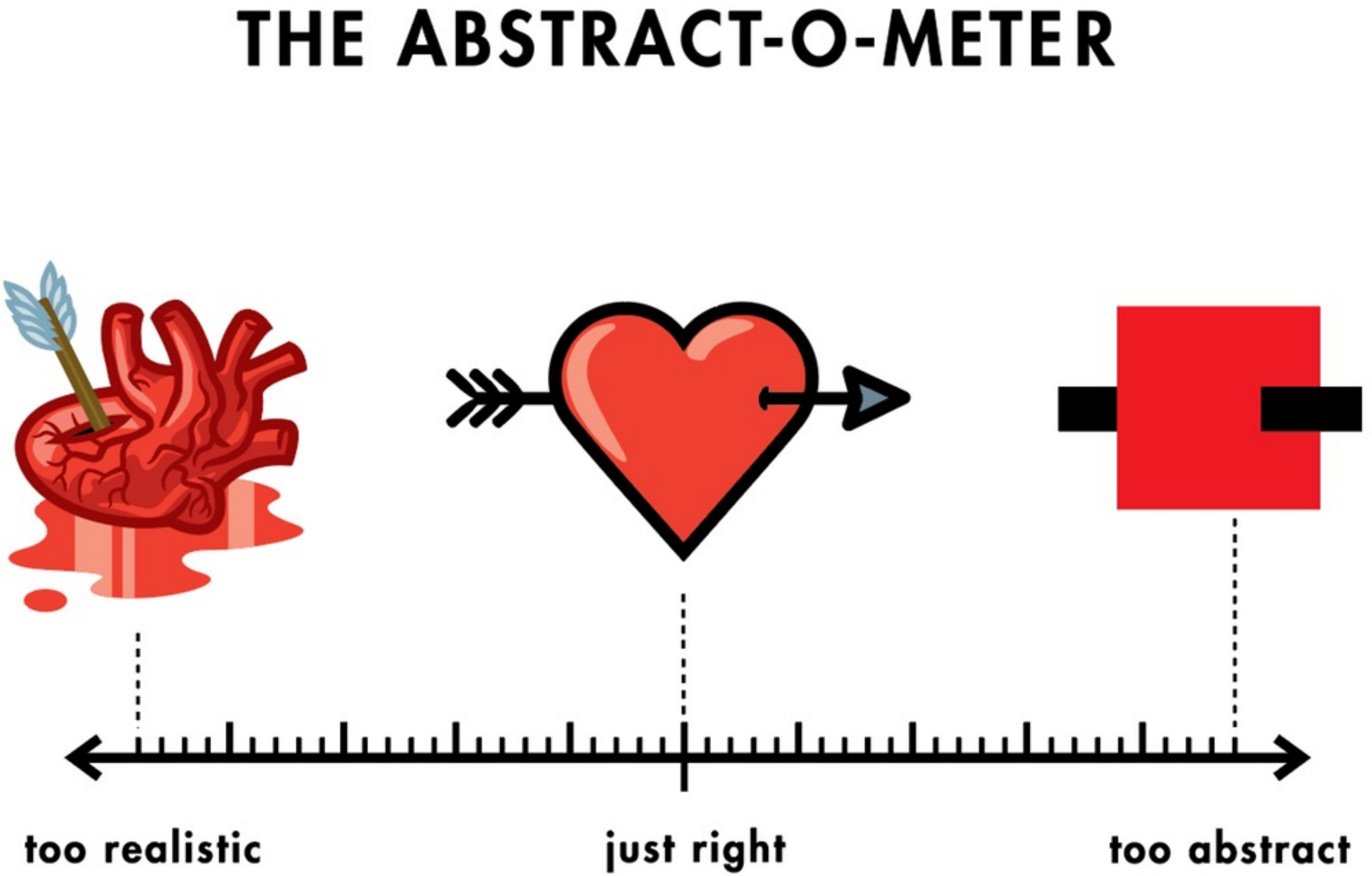

UI
UI
UI
UI
SEND COMMAND
CP
CP
CP
N
S
-
+


UI
UI
UI
UI
NETWORK
CP
CP
CP
N
S
-
+

Abstraction

Saloua Raouda Choucair, *Fractional Module*

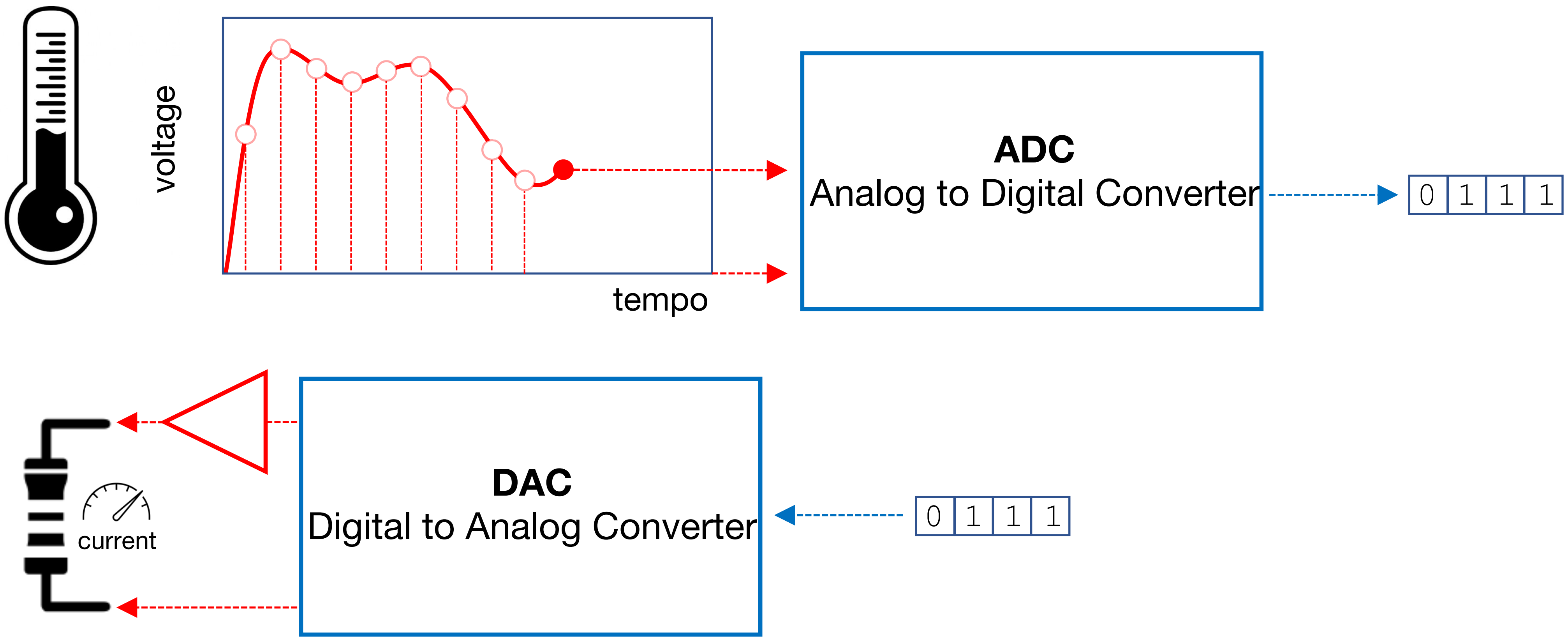

M
M
M
CP


THE ABSTRACT-O-METER
too realistic
just right
too abstract

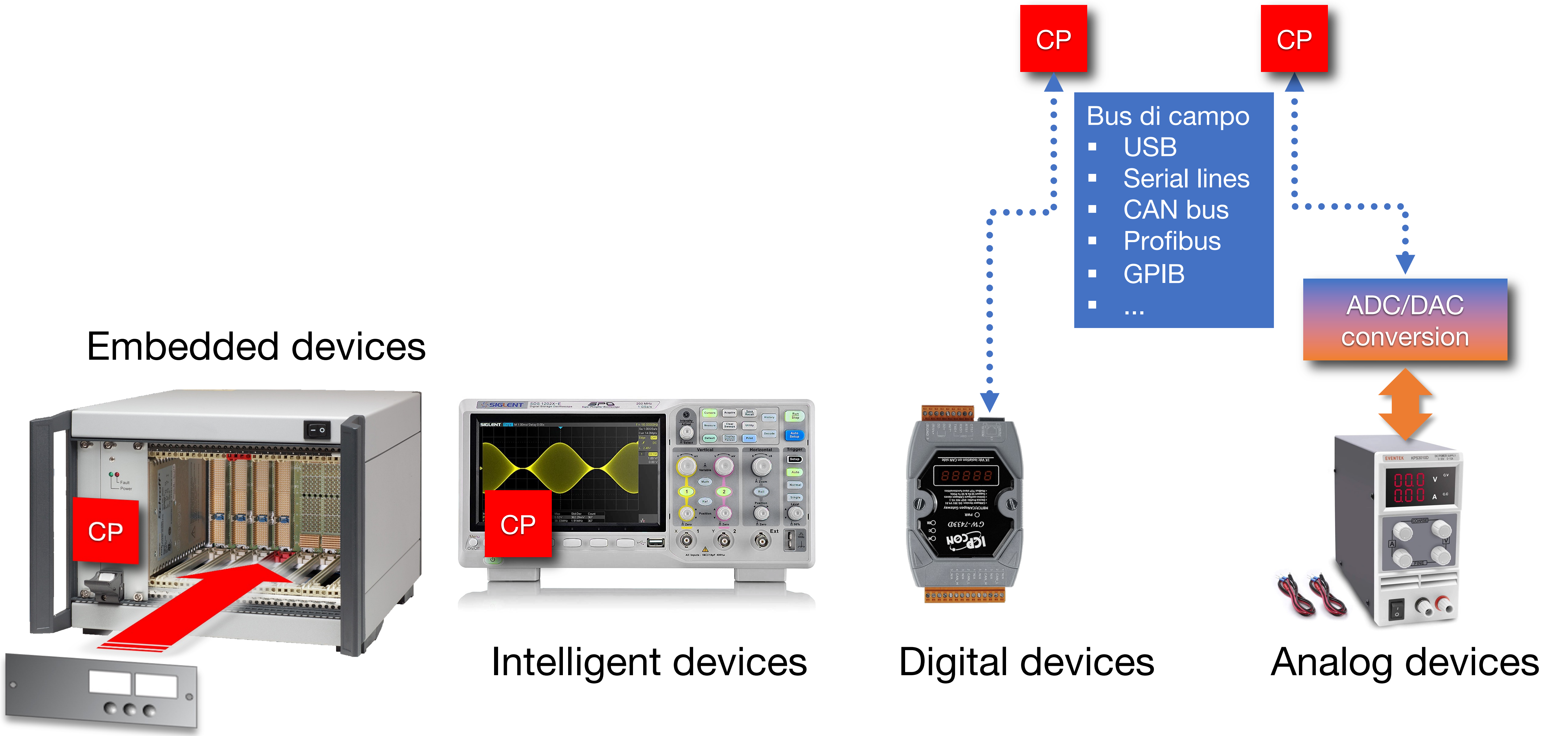
Features of a Control System

- acquisition/actuation
- communication
- storage
- presentation
- analysis / correlation
- historization

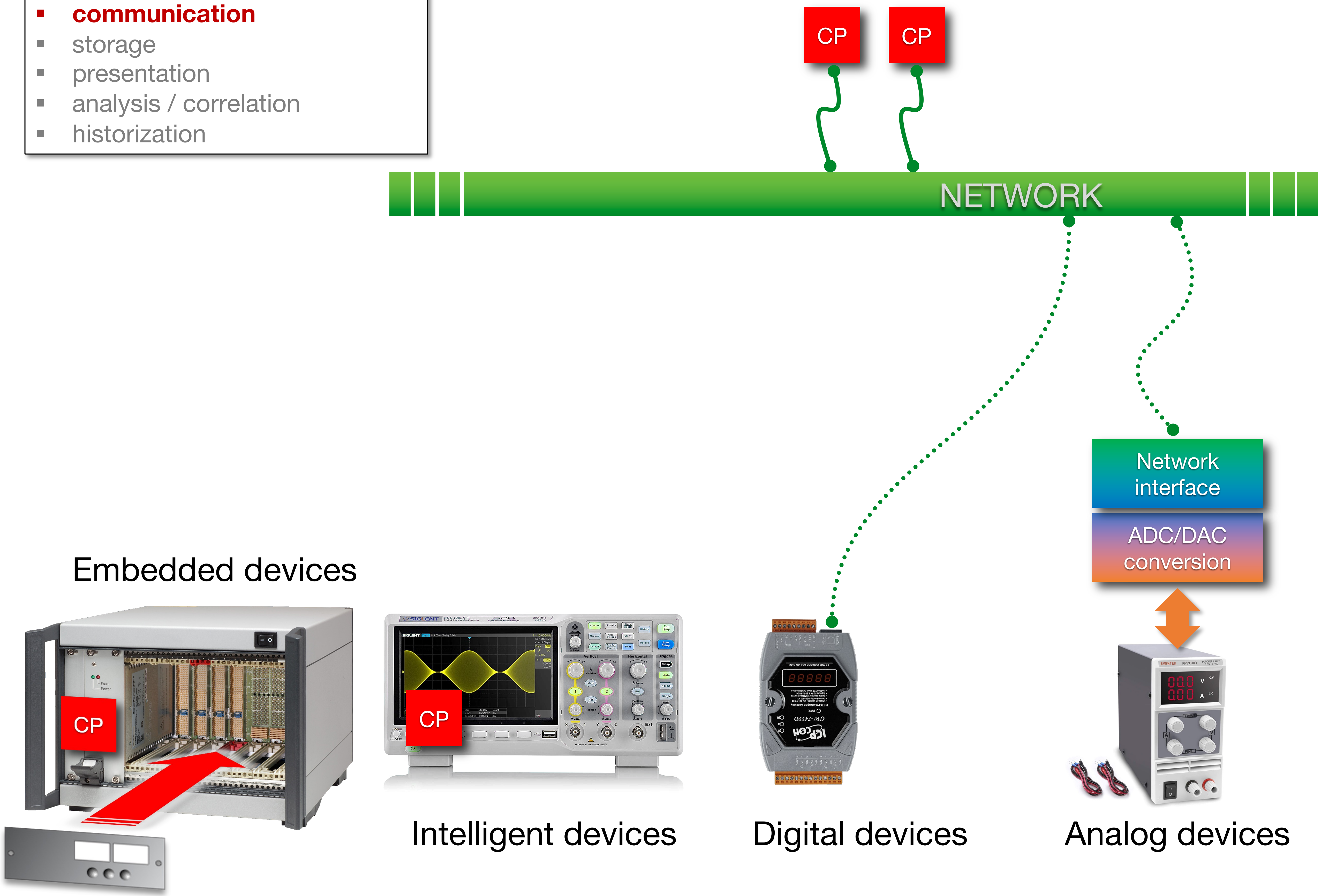
- **acquisition/actuation**
- communication
- storage
- presentation
- analysis / correlation
- historization



- acquisition/actuation
- **communication**
- storage
- presentation
- analysis / correlation
- historization



- acquisition/actuation
- **communication**
- storage
- presentation
- analysis / correlation
- historization



- acquisition/actuation
- communication
- **storage**
- presentation
- analysis / correlation
- historization

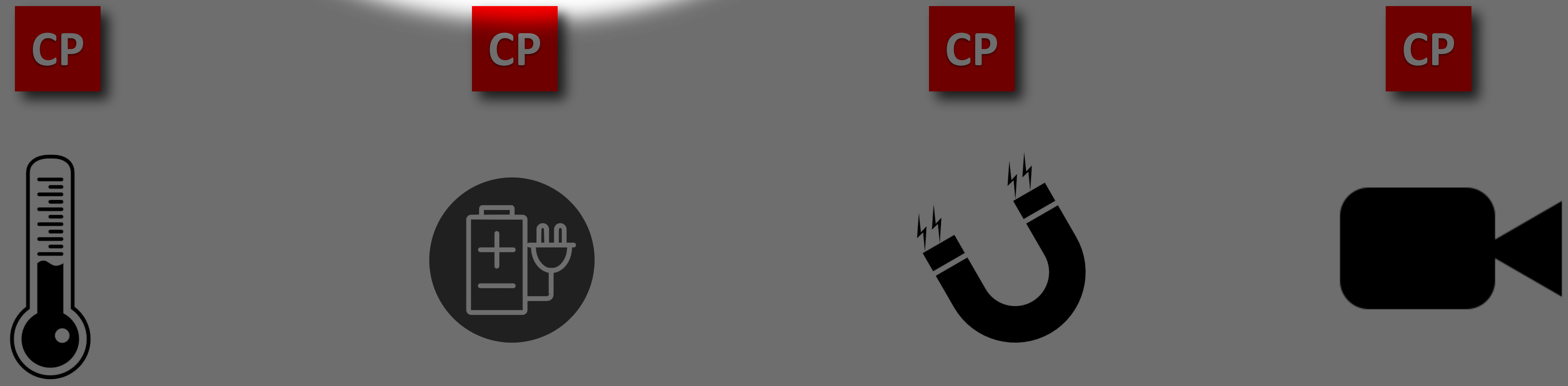
average size (@DAFNE)
~ 650 byte

```
{
float64 temperature_read;
char    units;
boolean status;
....
}
```

```
{
float64 current_read;
float64 current_set;
char    units;
boolean error;
....
}
```

```
{
float64 mag_field_read;
float64 mag_field_set;
int     polarity;
....
}
```

```
{
int     aperture;
int     time;
int     n_pixels_h;
int     n_pixels_v;
boolean status;
....
}
```



650 byte

Dante Alighieri (1265–1321)

Nel mezzo del cammin di nostra vita
mi ritrovai per una selva oscura,
ché la diritta via era smarrita.

Ahi quanto a dir qual era è cosa dura
esta selva selvaggia e aspra e forte
che nel pensier rinova la paura!

Tant’è amara che poco è più morte;
ma per trattar del ben ch’i’ vi trovai,
dirò de l’altre cose ch’i’ v’ho scorte.

Io non so ben ridir com’i’ v’intrai,
tant’era pien di sonno a quel punto
che la verace via abbandonai.

Ma poi ch’i’ fui al piè d’un colle giunto,
là dove terminava quella valle
che m’avea di paura il cor compunto

guardai in alto e vidi le sue spalle
vestite già de’ raggi del pianeta
che mena dritto altrui per ogne calle.

DAFNE ~1.4 Mbyte

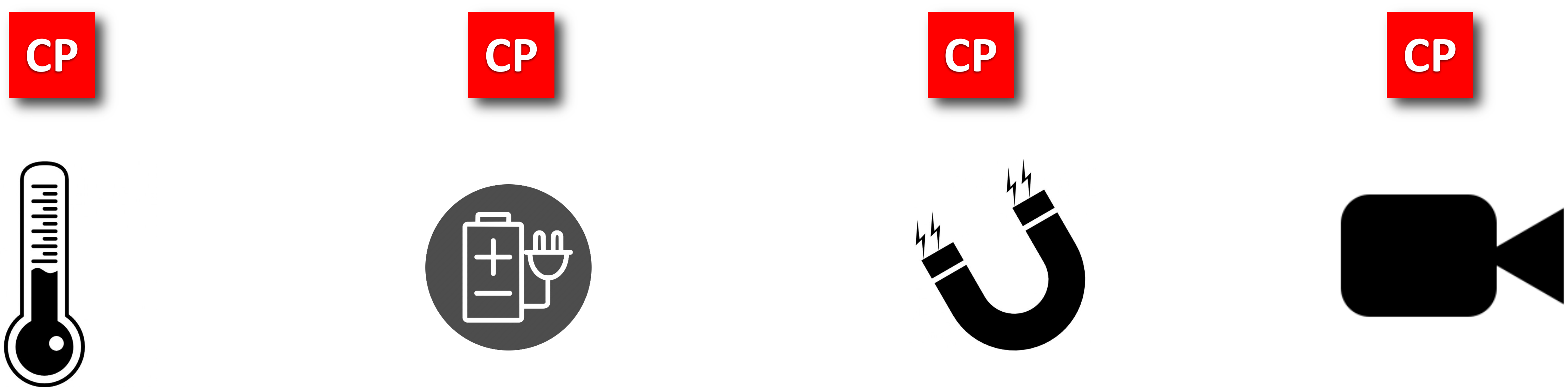
Relational database →

```
{
float64 temperature_read;
char    units;
boolean status;
....
}
```

```
{
float64 current_read;
float64 current_set;
char    units;
boolean error;
....
}
```

```
{
float64 mag_field_read;
float64 mag_field_set;
int     polarity;
....
}
```

```
{
int     aperture;
int     time;
int     n_pixels_h;
int     n_pixels_v;
boolean status;
....
}
```



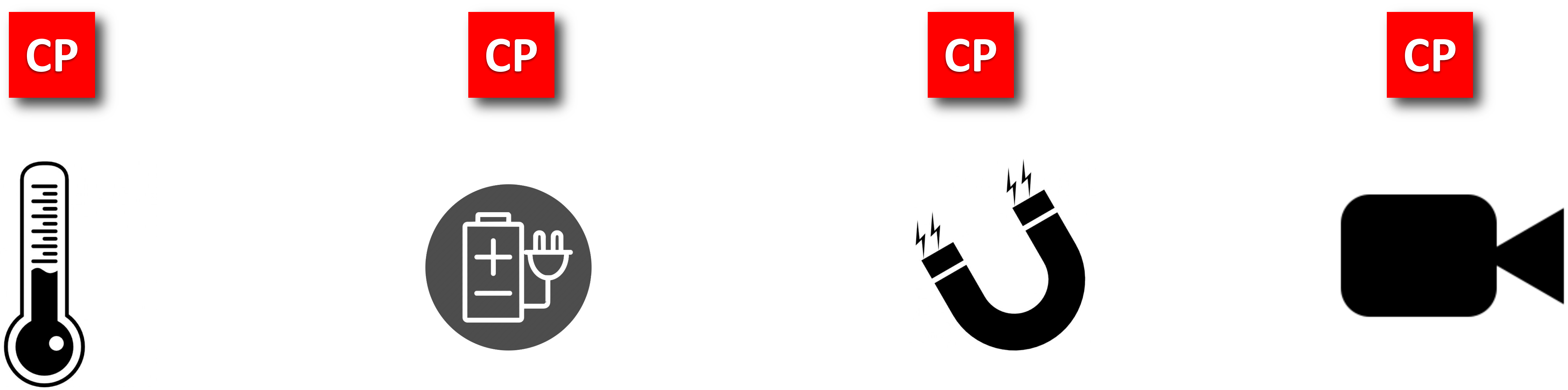
Relational database →

```
{
float64 temperature_read;
char    units;
boolean status;
....
}
```

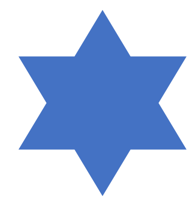
```
{
float64 current_read;
float64 current_set;
char    units;
boolean error;
....
}
```

```
{
float64 mag_field_read;
float64 mag_field_set;
int     polarity;
....
}
```

```
{
int     aperture;
int     time;
int     n_pixels_h;
int     n_pixels_v;
boolean status;
....
}
```



Relational database →



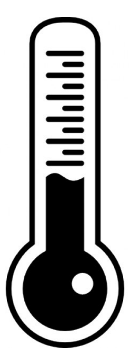
```
{
float64 temperature_read;
char    units;
boolean status;
....
}
```

```
{
float64 current_read;
float64 current_set;
char    units;
boolean error;
....
}
```

```
{
float64 mag_field_read;
float64 mag_field_set;
float64 orientation;
int     polarity;
....
}
```

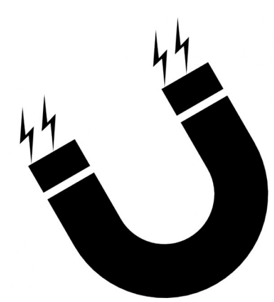
```
{
int     aperture;
int     time;
int     n_pixels_h;
int     n_pixels_v;
boolean status;
....
}
```

CP

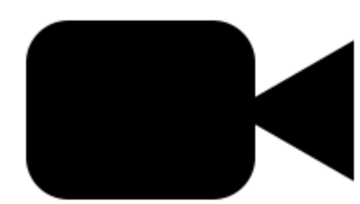


CP

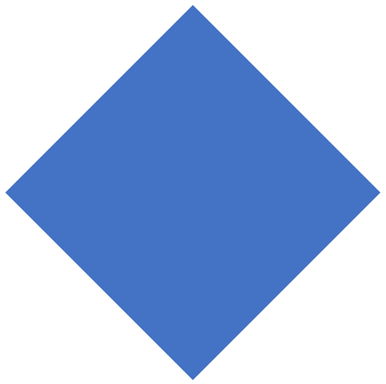
CP



CP



Serialization: *self-describing* data vs *formatted* data



```
BEGIN OF DATA
  here it is described a rhombus;
  each side measures 100 mm;
  the orientation of the diagonal is 0 deg;
  blah blah...;
END OF DATA
```

Most popular *self-describing* data formats

XML

```xml
<note>
  <to>Andrew</to>
  <from>Jane</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

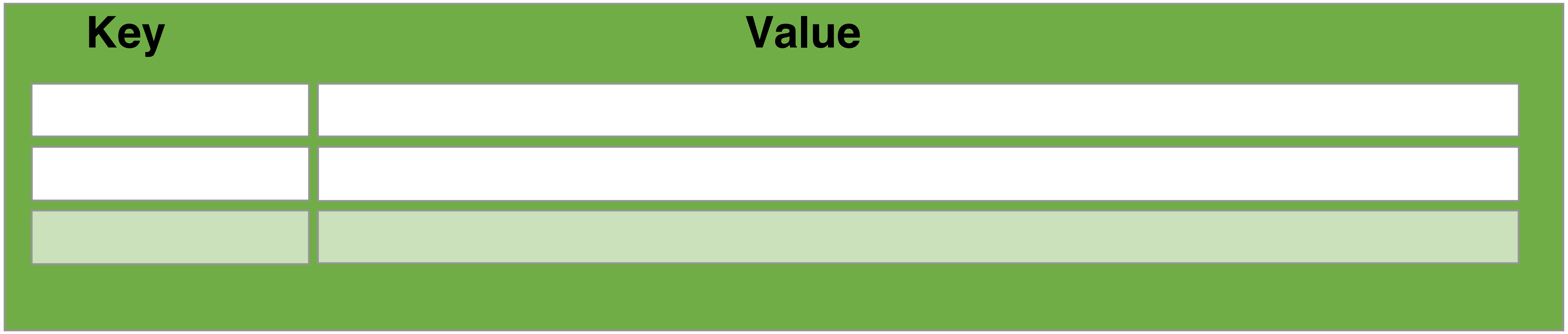
YAML

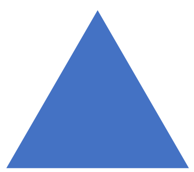
```yaml
note:
  to: Andrew
  from: Jane
  heading: Reminder
  body: "Don't forget me this weekend!"
```

JSON

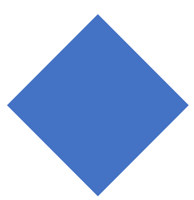
```json
{
  "note":{
    "to":"Andrew",
    "from":"Jane",
    "heading":"Reminder",
    "body":"Don't forget me this weekend!"
  }
}
```

Non relational database





`my_data` `0x16\0x00\0x00\0x00\0x02`**`hello`**`\0x00\0x06\0x00\0x00\0x00`**`world`**`\0x00\0x00`



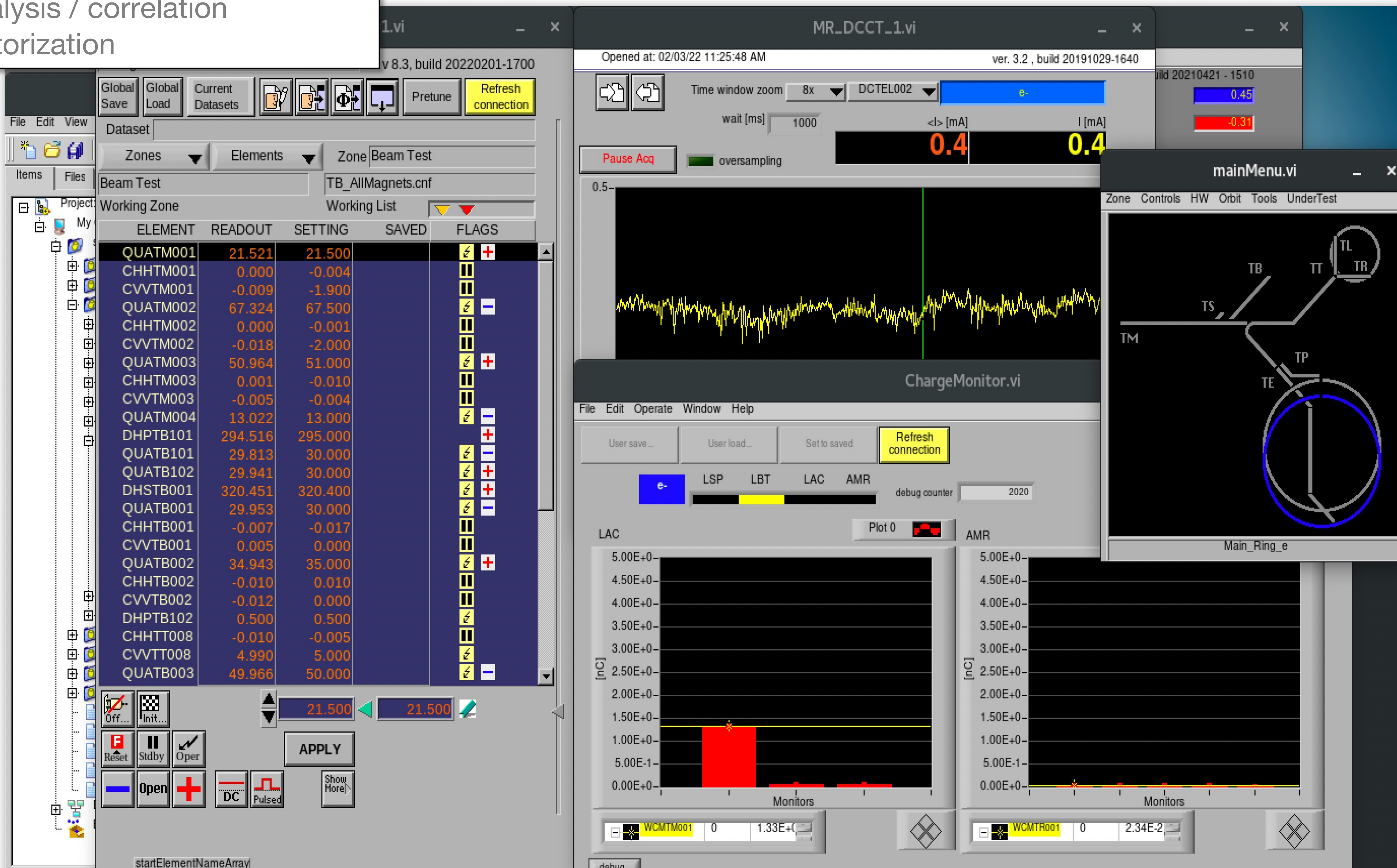
`my_other_data` `.... another descriptive sequence .......`

e.g. BSON serialization

\0x16\0x00\0x00\0x00	total document size
\0x02	0x02 = field data type is string
hello\0x00	field name
\0x06\0x00\0x00\0x00world\0x00	field value (size of value, value, terminator)
\0x00	type EOO ('end of object')

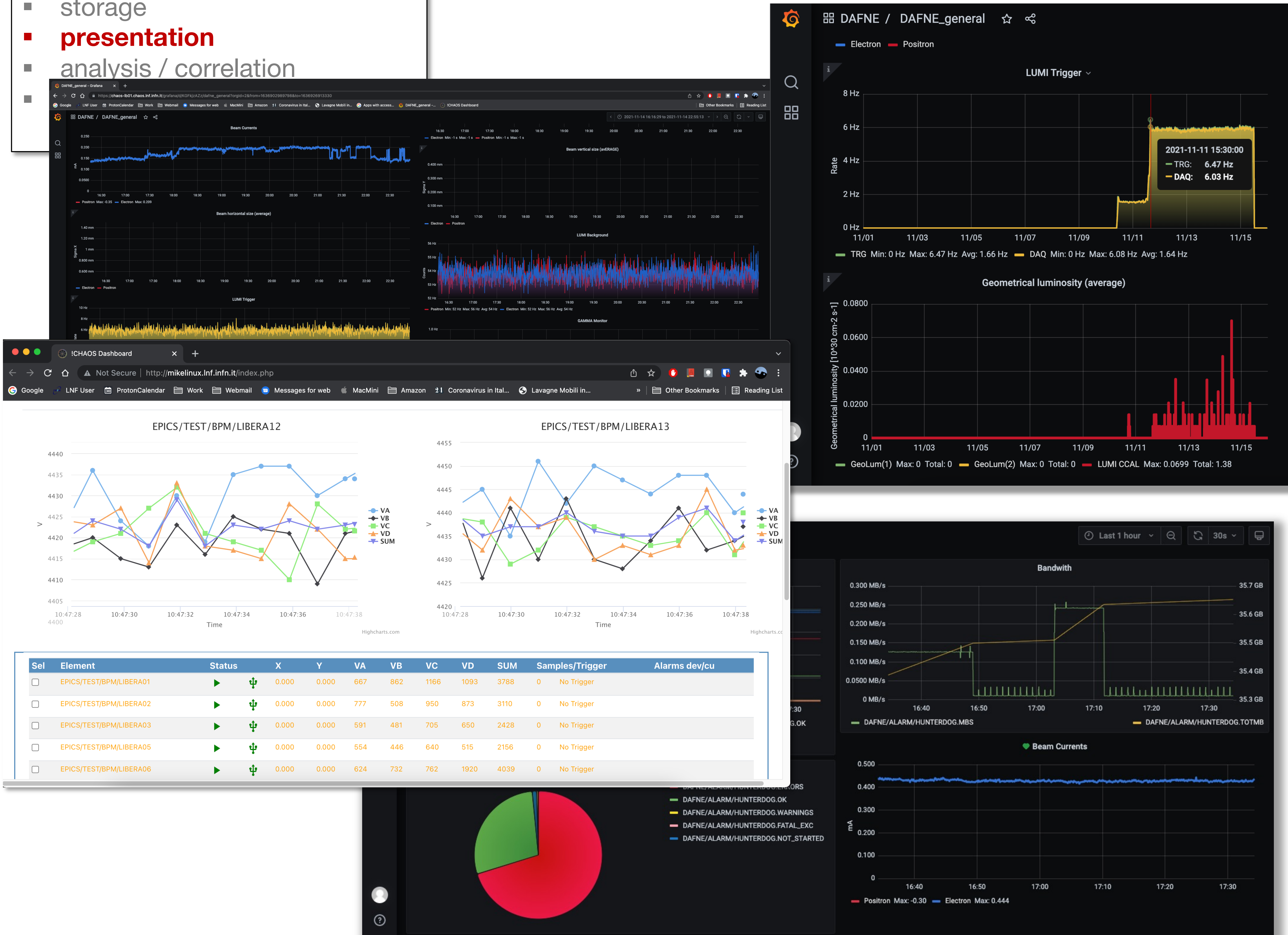
- acquisition/actuation
- communication
- storage
- **presentation**
- analysis / correlation
- historization

User Interface

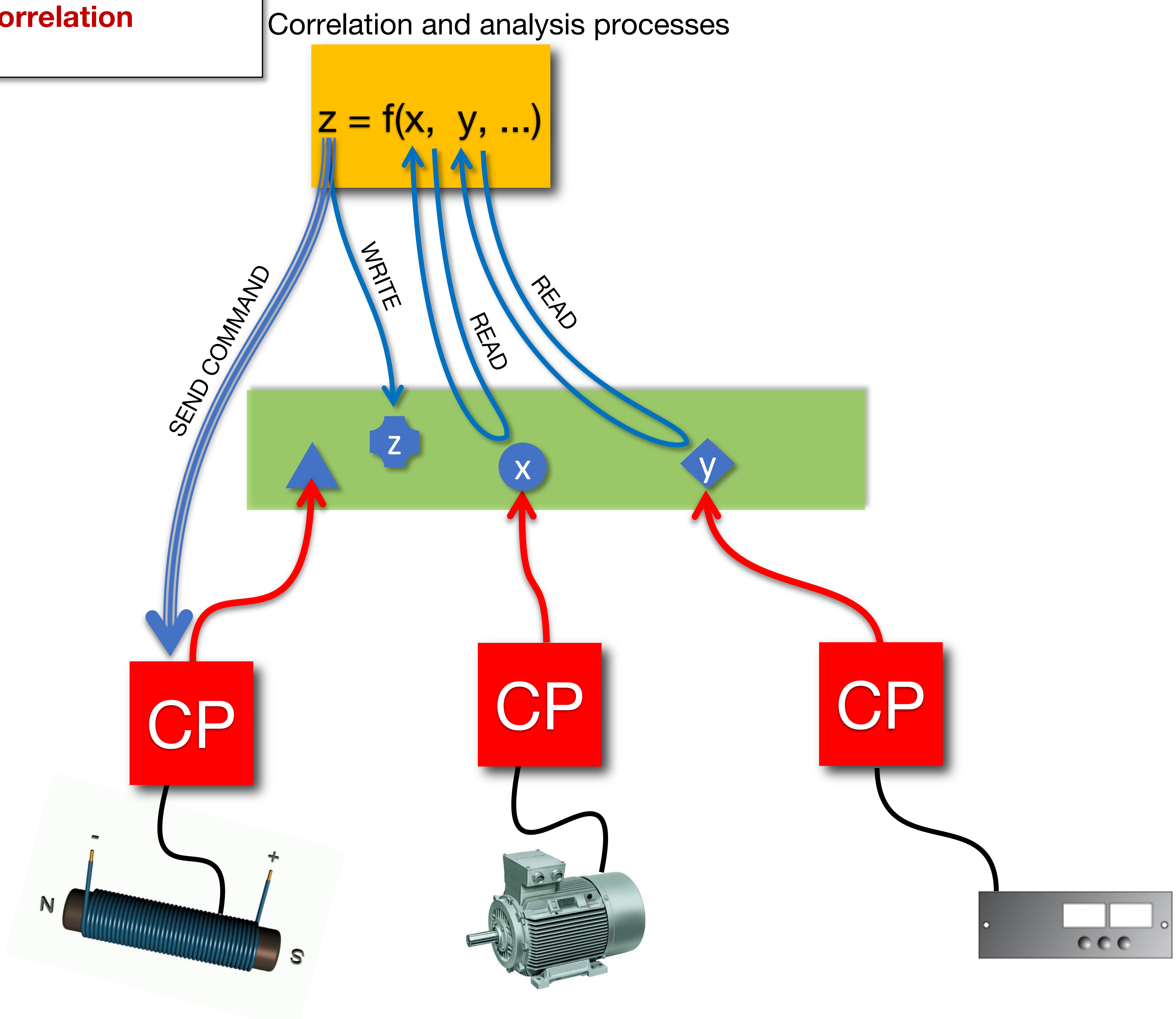


User Interface

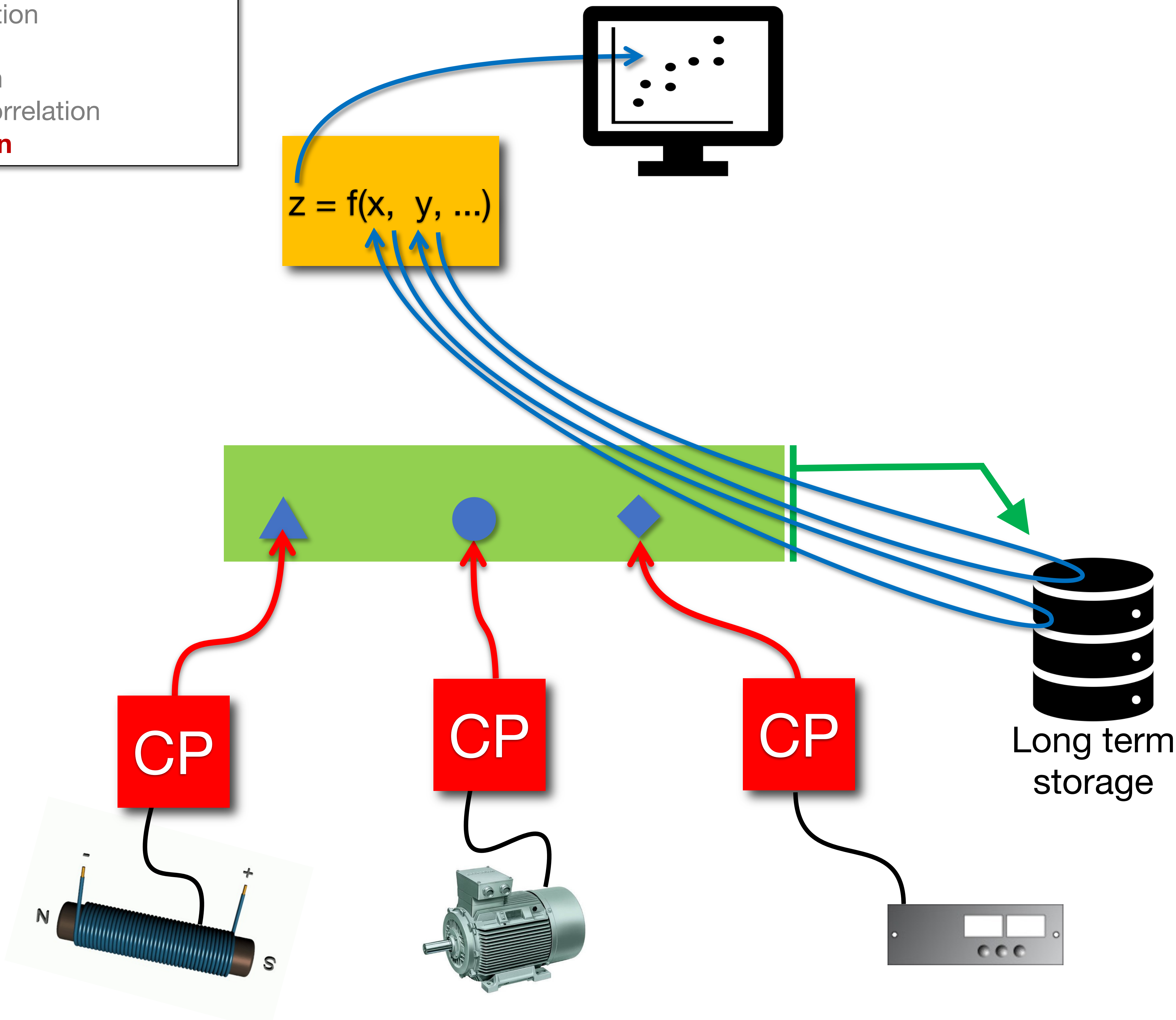
- acquisition/actuation
- communication
- storage
- **presentation**
- analysis / correlation



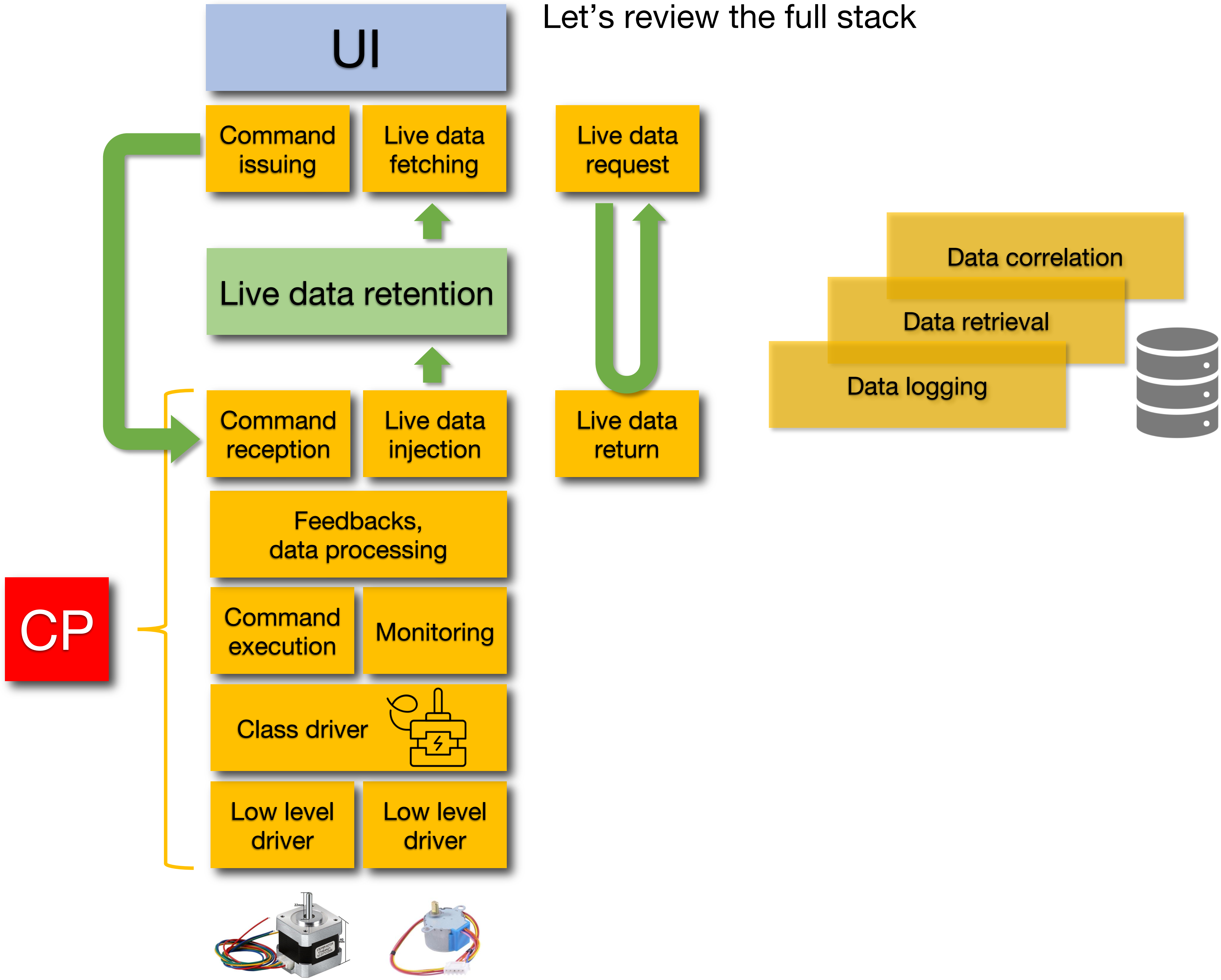
- acquisition/actuation
- communication
- storage
- presentation
- **analysis / correlation**
- historization



- acquisition / actuation
- communication
- storage
- presentation
- analysis / correlation
- **historization**



Let’s review the full stack

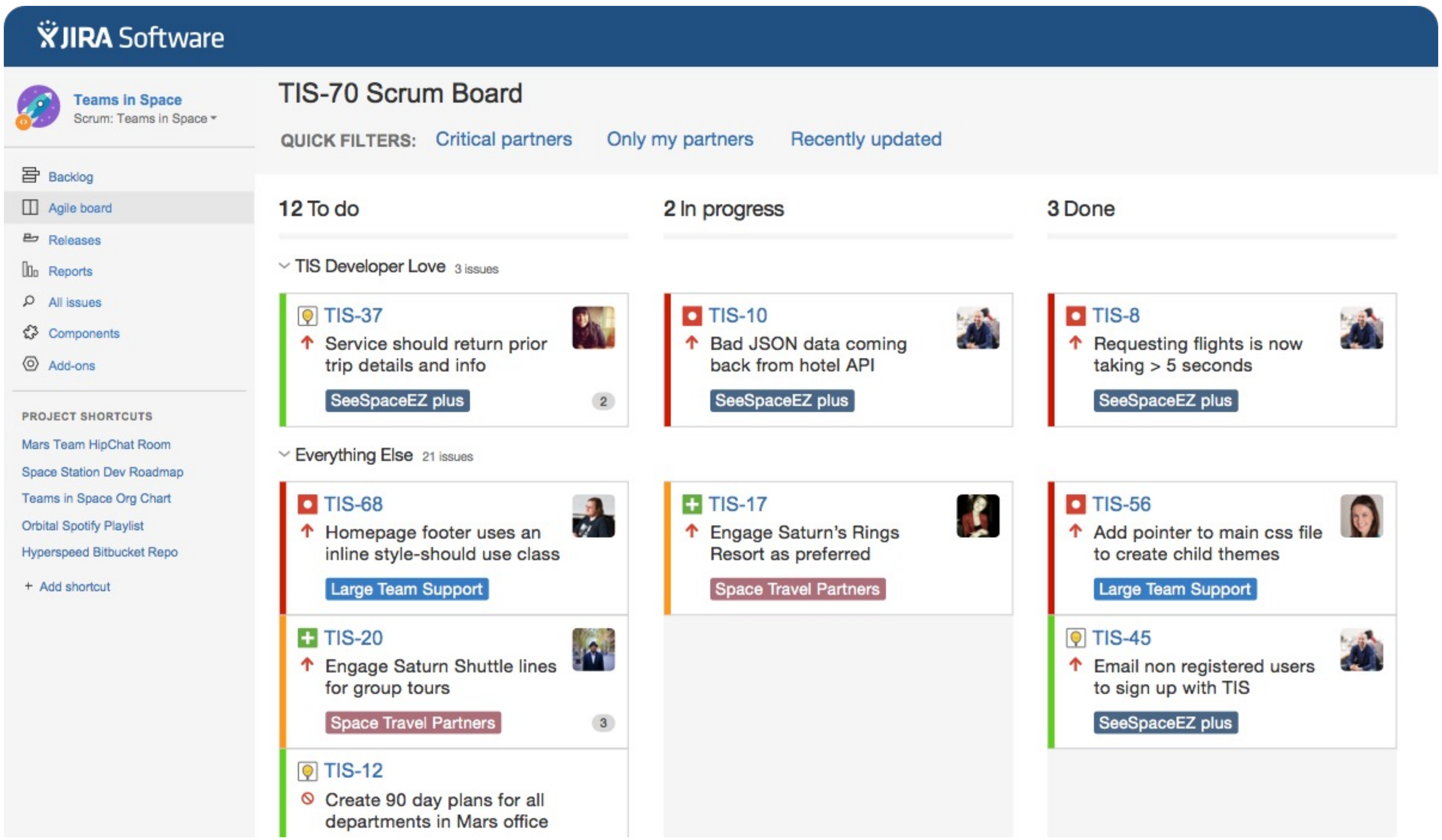


Golden rules

- Analysis → Design → Development → Qualification
- <u>Do</u> abstract physical devices and create classes
- **KISS** (Keep It Simple)

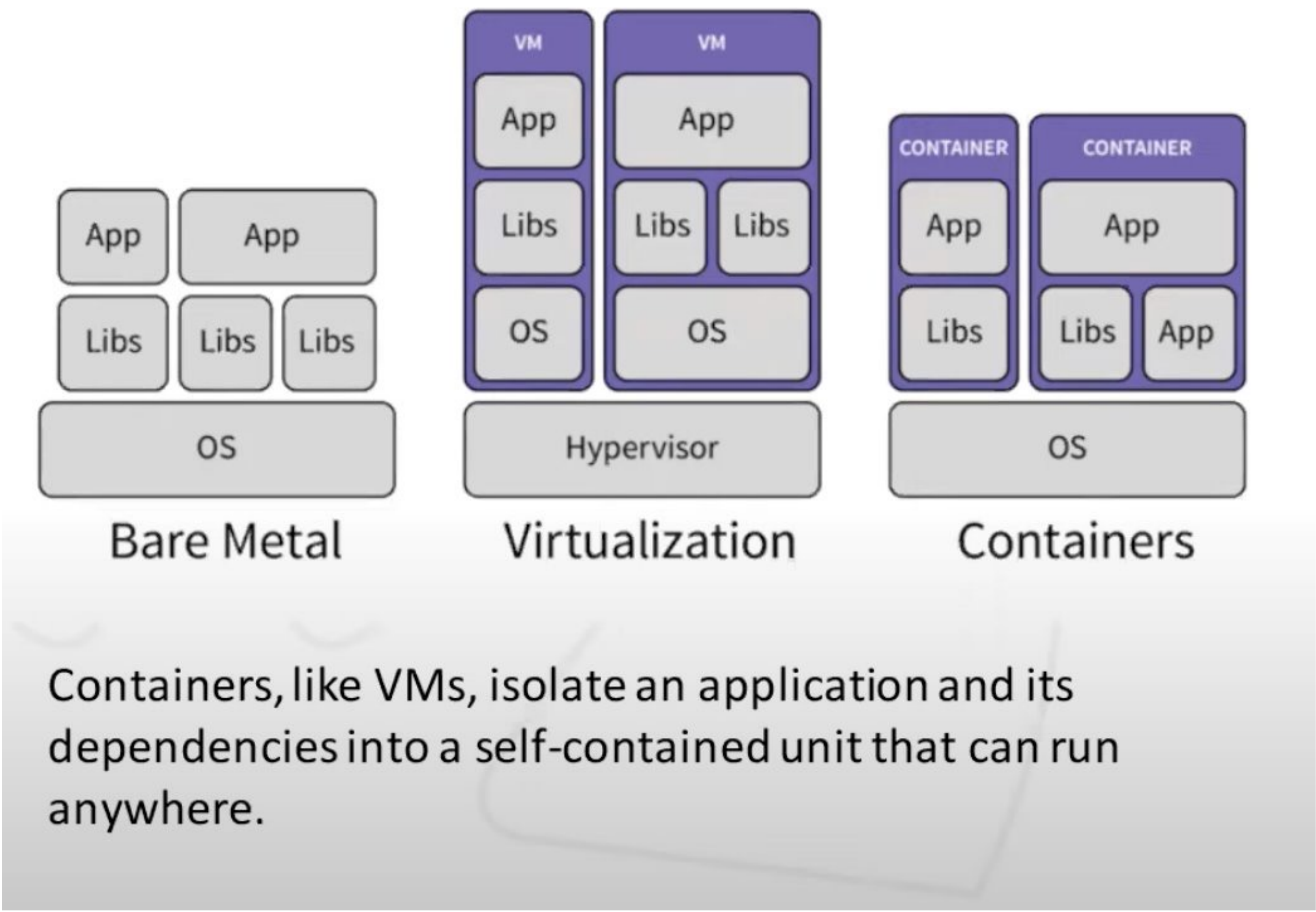
IT good methods & practices

- Development: Agile approach, CI/CD, DevOps
- Documentation, Asset Management
- Project tracking

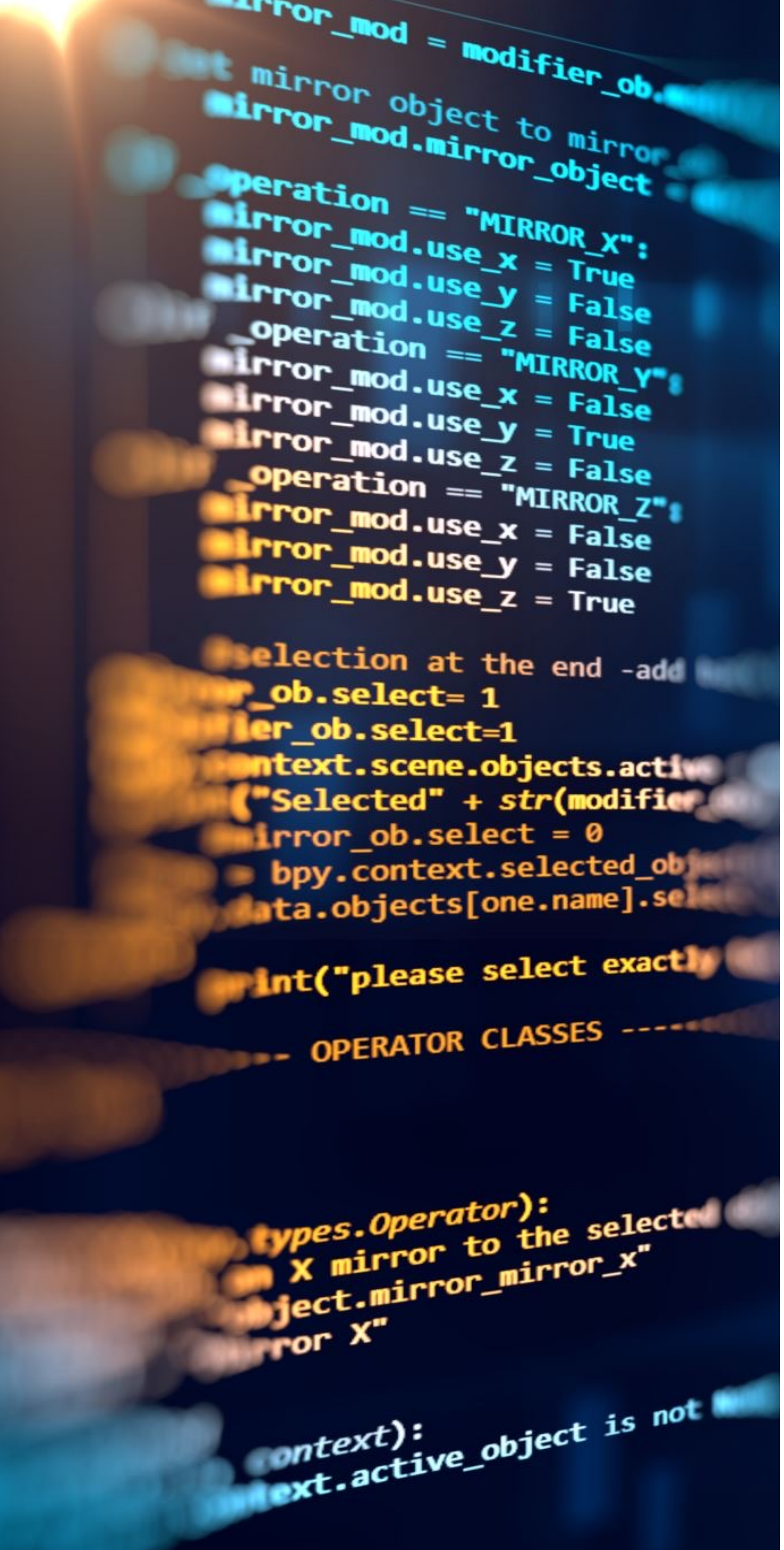


IT interesting topics

- Big data
- AI
- Virtualization, Dockers, Orchestrators



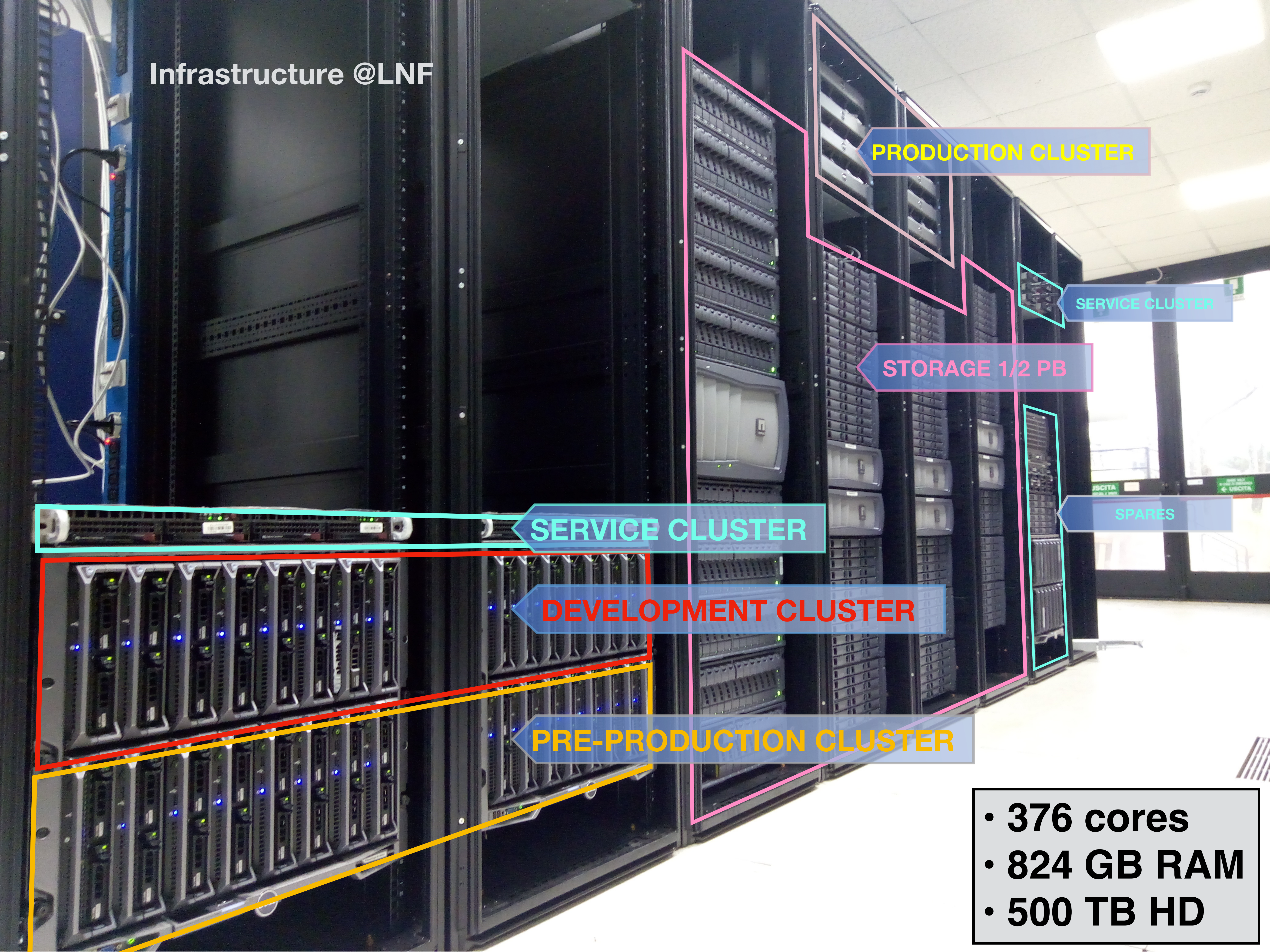
Courtesy of Giles Knap (Diamond)



Professionals

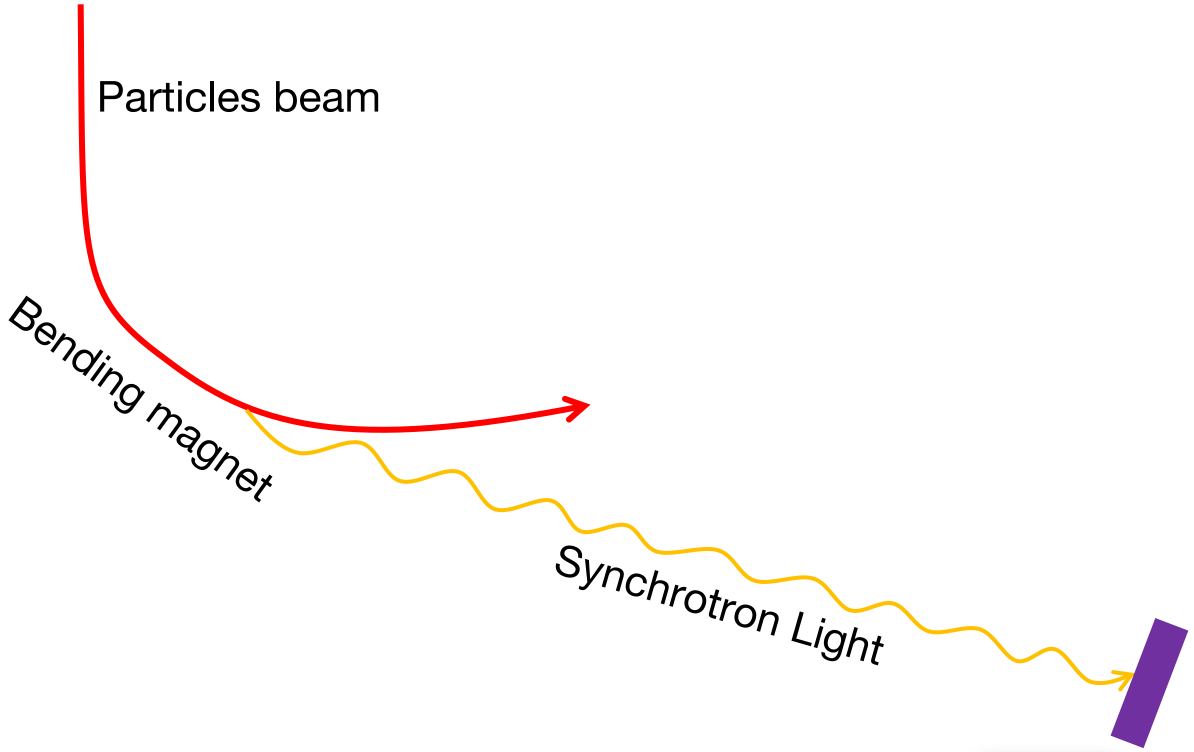
- Data scientist
- Software Developer, Software Engineer, DevOps
- System Manager
- Network Manager
- HW system Integrator

Infrastructure @LNF



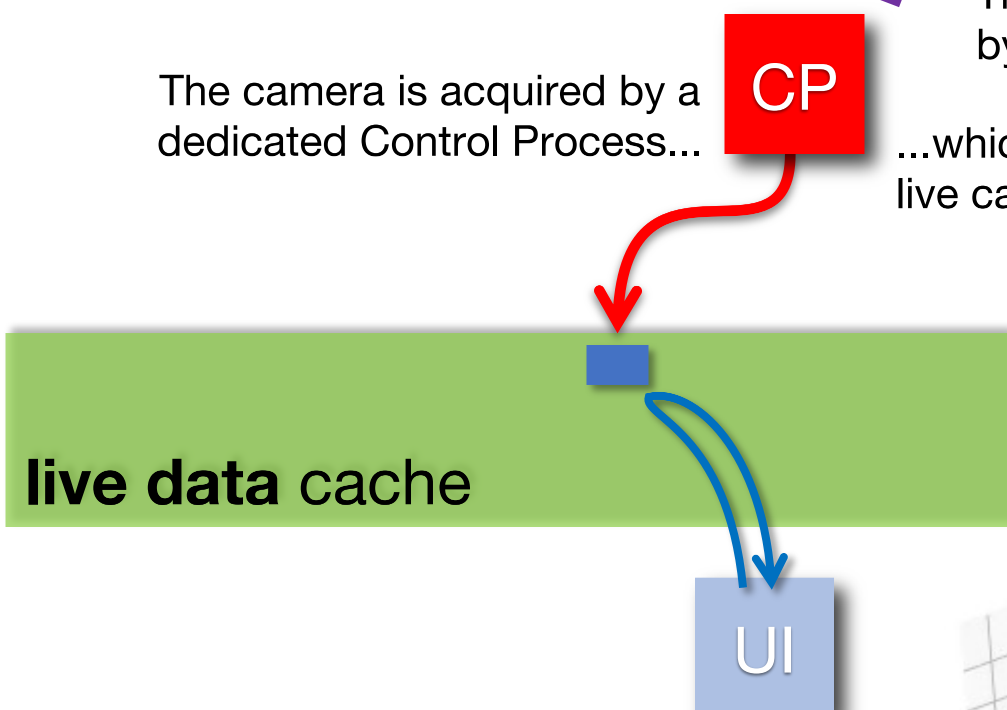
- 376 cores
- 824 GB RAM
- 500 TB HD

Today's exercise



SLM (Synchrotron Light Monitor)
It is basically a CCD camera
The CCD resolution is 1200 x 950 pixels
The numeric value of the luminosity on each pixel is returned by the camera as an integer value in the range 0-4095.

The camera is acquired by a dedicated Control Process...



...which writes the matrix in the system live cache (at ~ 2Hz)

The exercise is to continuously readback the matrix from the central cache and calculate *on-the-fly* the FWHM of the 3D profile of the spot.