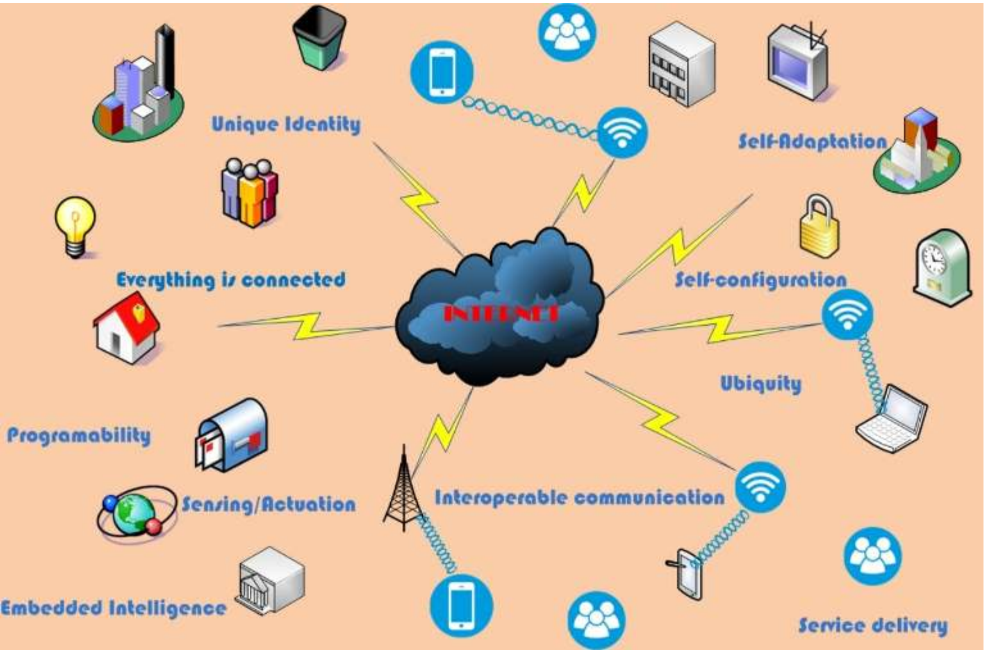
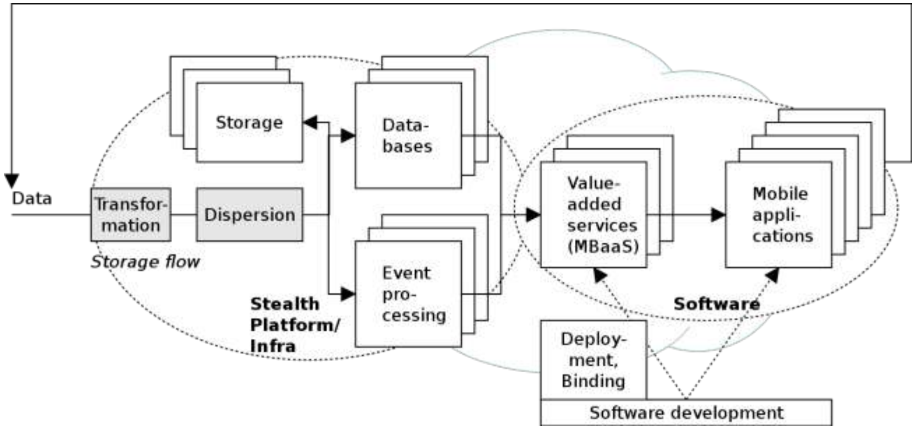
Keynote: Serverless Cyber-Physical Applications

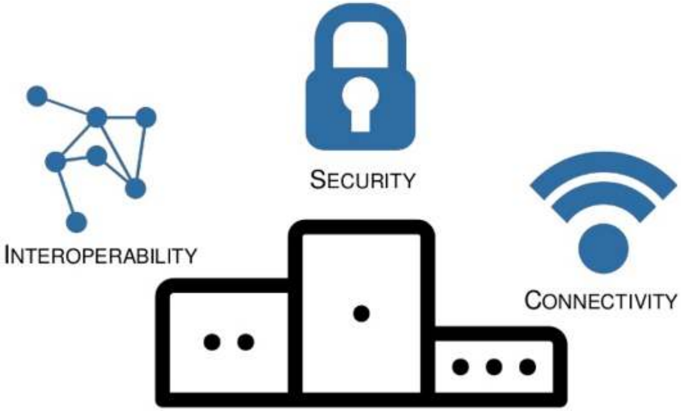
Josef Spillner <josef.spillner@zhaw.ch>
Service Prototyping Lab (blog.zhaw.ch/icclab)

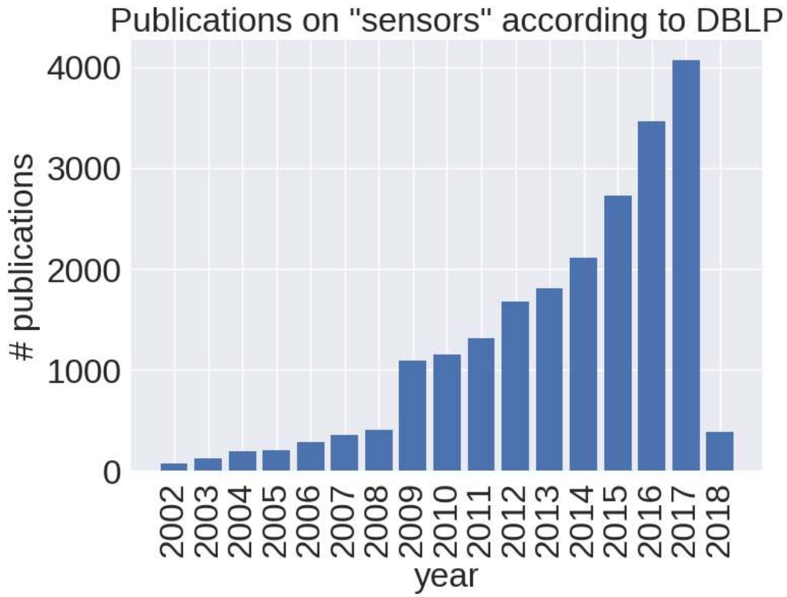
Mar 8, 2018 | Science Meets Industry, Dresden, DE

Sensing the World





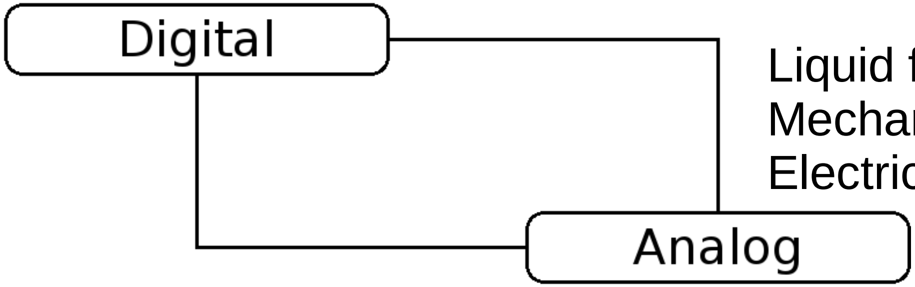




[Roberto Minerva et al., 2015; IoT Developer Survey 2017; own works 2015/2018]

*-Computing Paradigms 1968

“... physical devices, the state of which changes according to well-defined rules.“

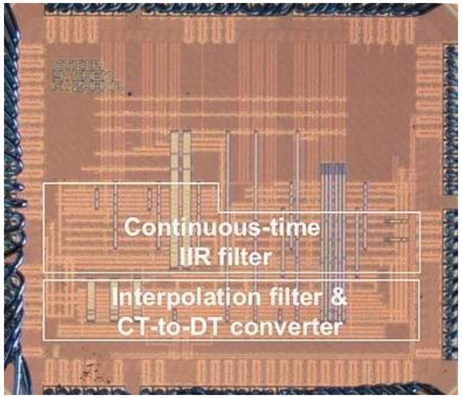


Liquid flows
Mechanical contraptions
Electric components

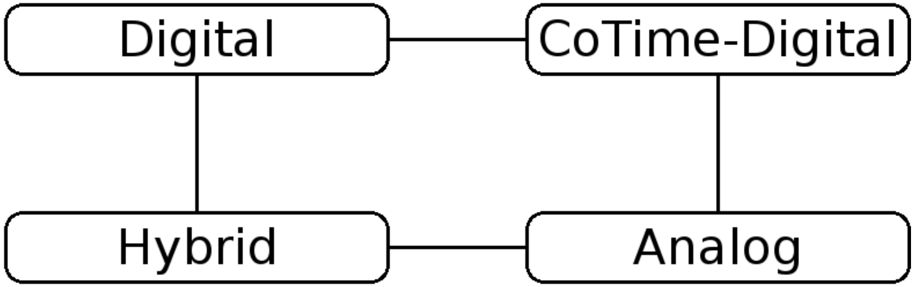
“... a physical system configured so that it is governed by equations identical to the ones you want to solve“

[Yannis Tsividis, IEEE Spectrum 02/2018; De Raedt, 2016]

*-Computing Paradigms 1968bis



Approximate Computing



[Yannis Tsividis, IEEE Spectrum 02/2018; Steven Fine]

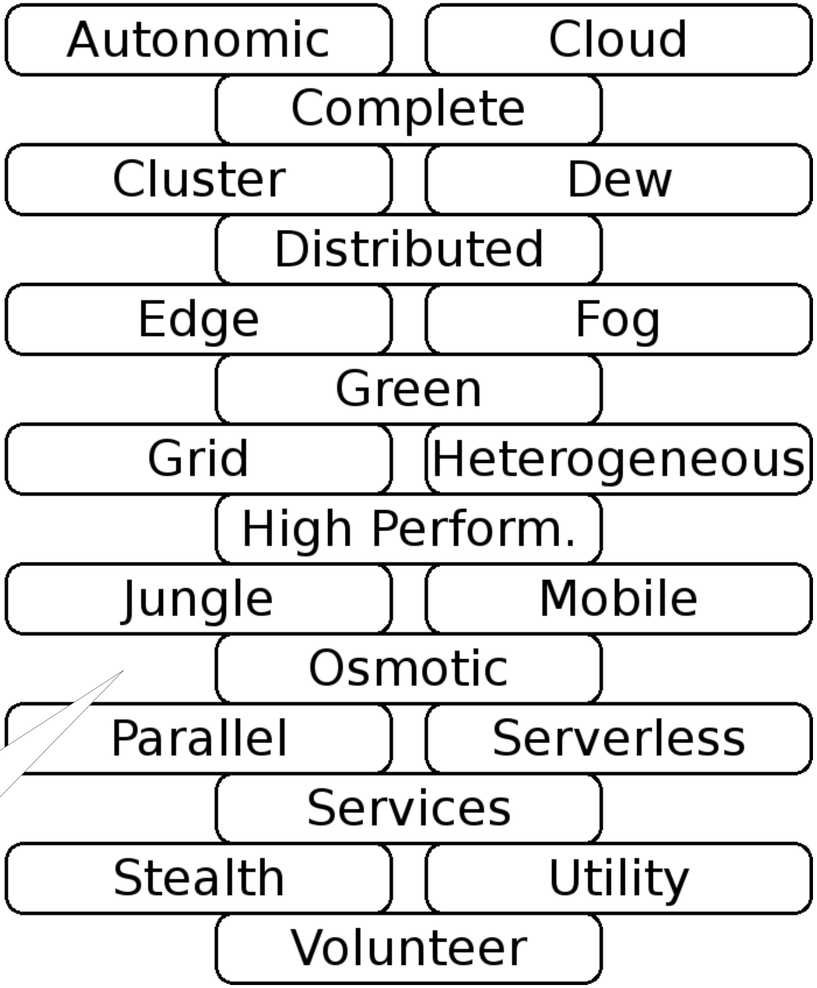
*-Computing Paradigms 2018

P2P? Trusted? Mobile Cloud? Scientific? CPS? Concentric?

"... simultaneous combination of heterogeneous, hierarchical, and distributed computing resources"

+ underspecified definitions ca. '12

"... decompose applications into microservices ... exploiting resources in edge and cloud infrastr."



Autonomic

Cloud

Complete

Cluster

Dew

Distributed

Edge

Fog

Green

Grid

Heterogeneous

High Perform.

Jungle

Mobile

Osmotic

Parallel

Serverless

Services

Stealth

Utility

Volunteer

"... to provide decentralized, cloud-friendly, and collaborative micro services to end-users"

+ competing definitions ca. '16

Required: dedicated research + teaching on *-paradigms

Paradigm → Platform → Applications

[Dimosthenis Kyriazis and Keith G. Jeffery, FGCS 78(1)/2018 & others]

Interdisciplinary Full-Stack Trends

Levels and granularities for computing paradigms
Technological & business innovations

Ecosystems
→ open hubs

Services
→ e.g. nanoservices, functions

Software
→ e.g. software, data, resources abstraction

Networks
→ e.g. travelers WiFi, 5G NR

Electronics
→ e.g. server-on-modules, RISC-V/OPENPULP

Material Sciences
→ e.g. self-adaptive materials

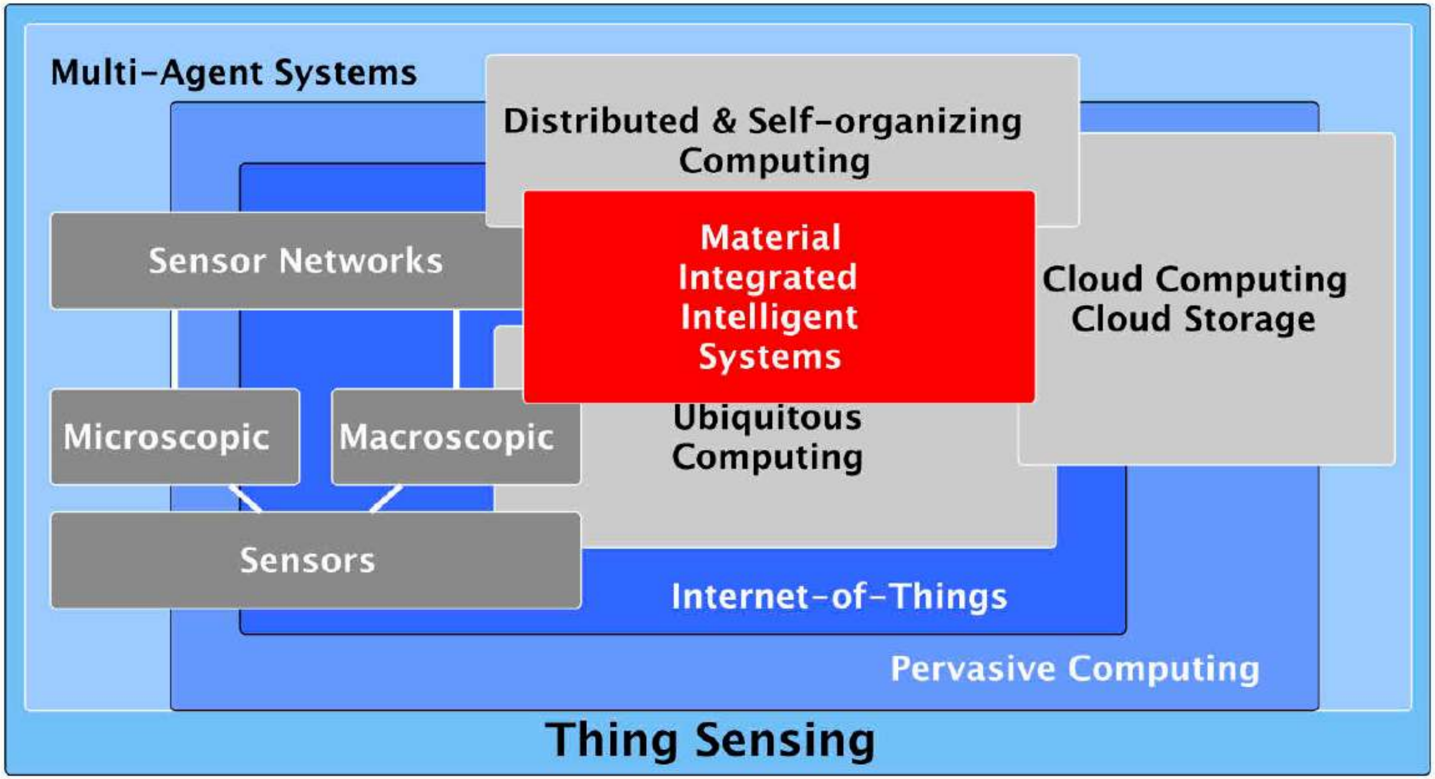
→ Clearly:
Serverless CPA only one possible cross-cut

[EECatalog 2016; Samsung]

Example: Hardware-Driven Features

Self-Adaptive Materials → Material-Integrated Computing

- sensors (smart dust) embedded into environment as sensor/actuator
- functionality: sensing, aggregation, application + strain vs. stiffness
- operations: processing, communication, storage, messaging, security...



[Bosse & Lehmhus DSS 2017]

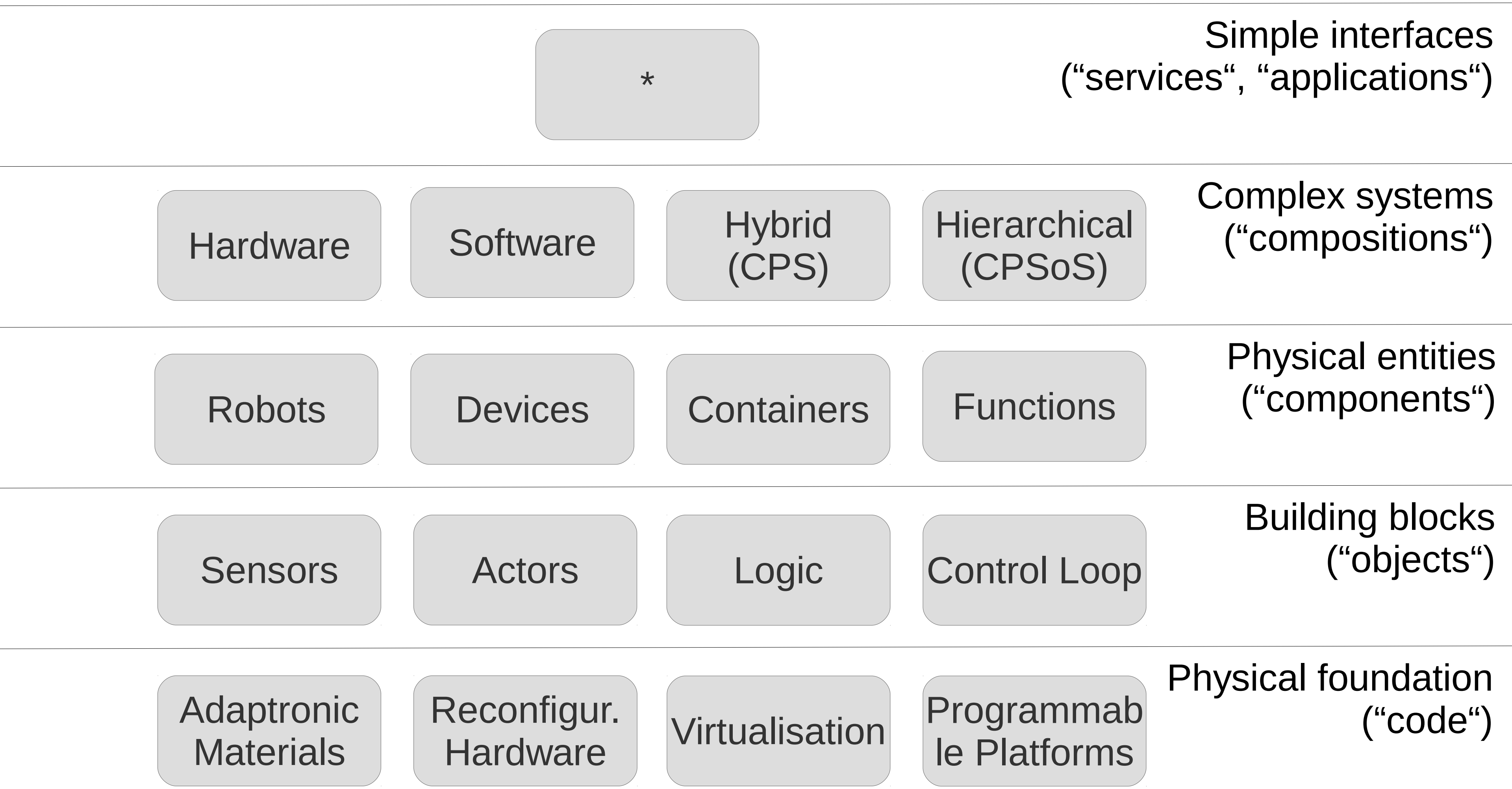
When Cyber (Digital) is not enough

[icc.mtu.edu]

Cyber-Physical Application (CPA): A composite application executed in parallel across physical and virtual spaces.

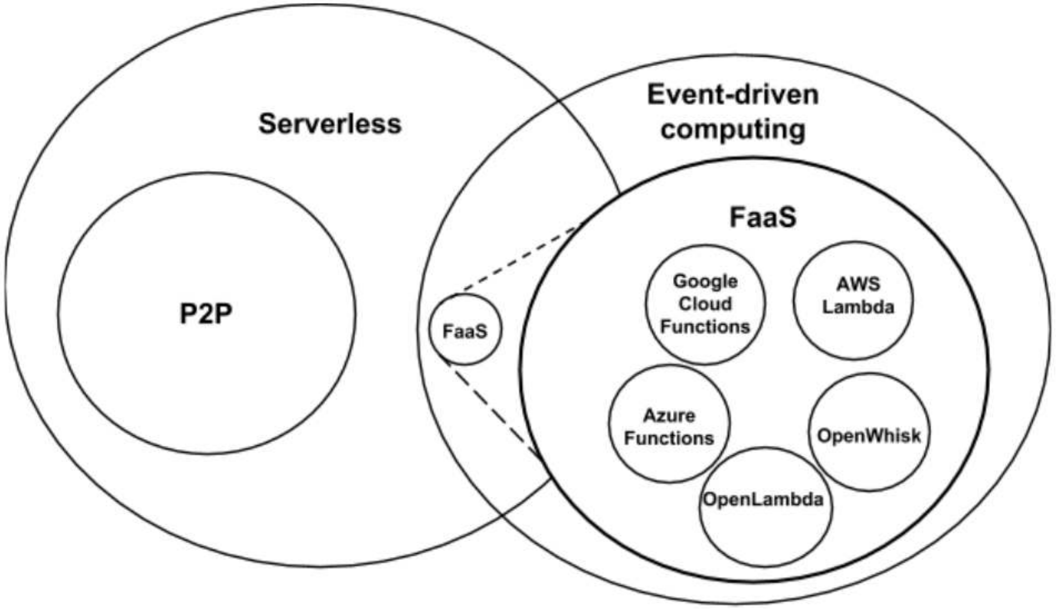
[adapted from own works at PTI 2017]

CPA-Related Terms and Trends



	*			Simple interfaces (“services“, “applications“)
Hardware	Software	Hybrid (CPS)	Hierarchical (CPSoS)	Complex systems (“compositions“)
Robots	Devices	Containers	Functions	Physical entities (“components“)
Sensors	Actors	Logic	Control Loop	Building blocks (“objects“)
Adaptronic Materials	Reconfigur. Hardware	Virtualisation	Programmable Platforms	Physical foundation (“code“)

“Serverless“ Computing



AWS Lambda

Google Cloud Platform Functions

IBM Cloud Functions

Azure

spotinst Functions

hook.io

webtask

OVH Functions PaaS

Serverless Application: A set of cloud-native stateless services with single-function granularity deployed at highest-level platforms.

Ecosystem: runtimes, composers, deployers, debuggers/tracers, transformers, converters, marketplaces, ...

[Geoffrey C. Fox et al., arXiv:1708.08028; own works at Swiss Python Summit 2018]

Serverless Taxonomy

Serverless application features

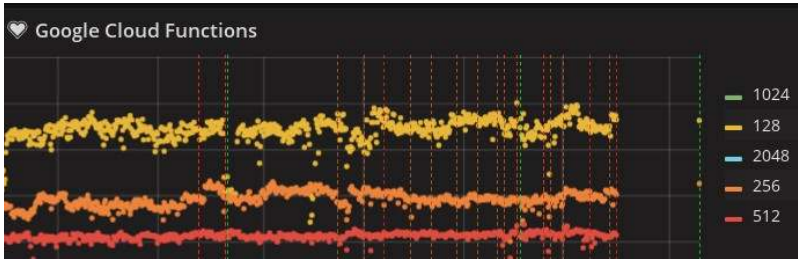
- Function-as-a-Service (FaaS) delivery
- triggered by events such as clock
- isolated function execution
- billing per invocation & load

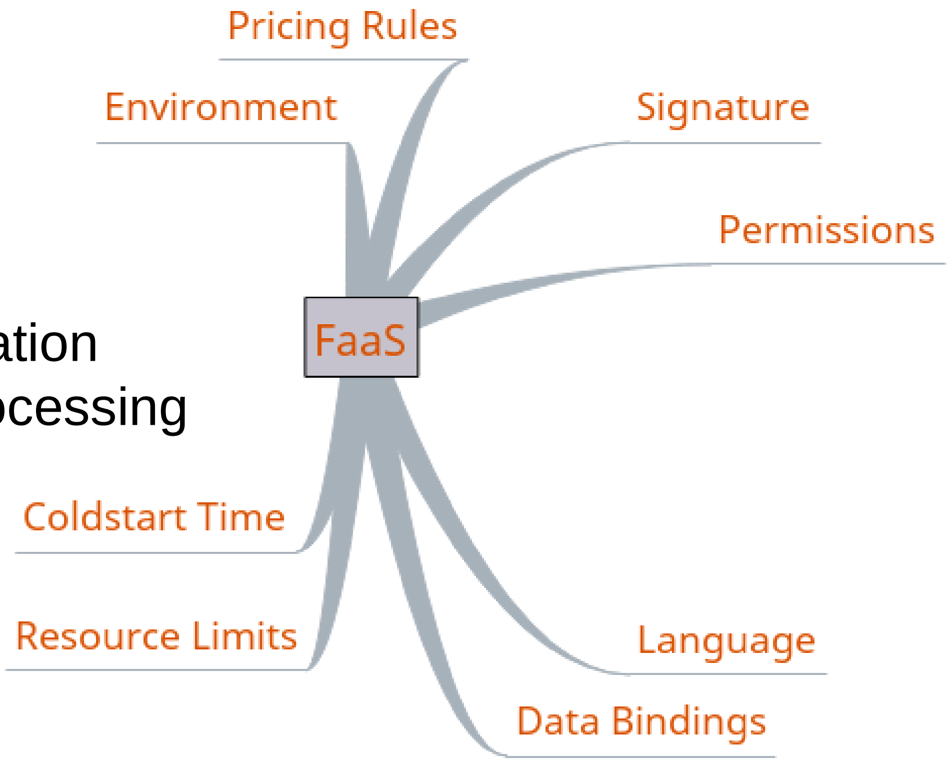
Combination with CPS/CPA

- Serverless Cyber-Physical Application
- pay-per-use for physical event processing

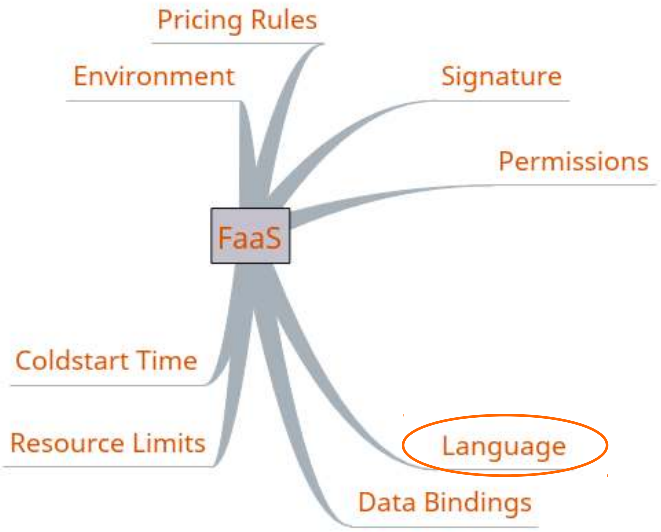
Hidden runtime characteristics

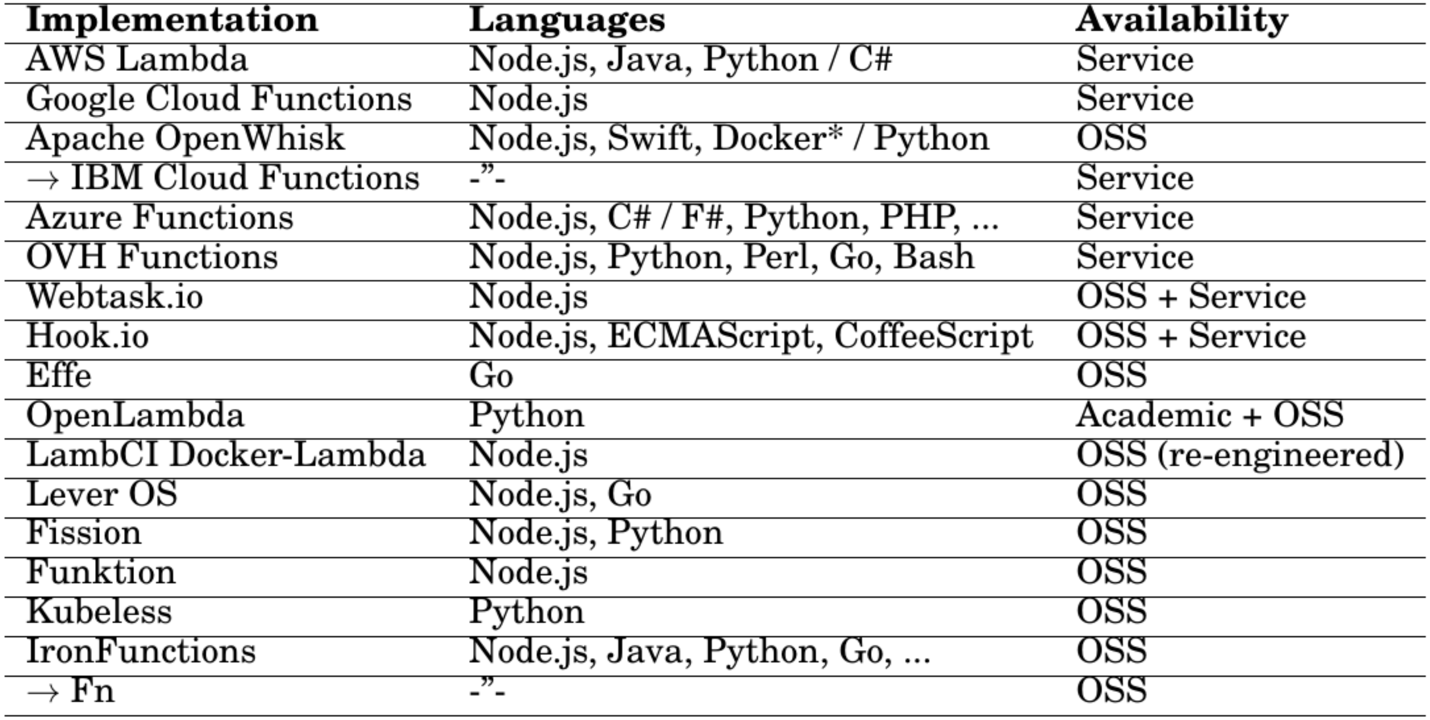
- known via long-time experiments





Serverless Taxonomy - Language





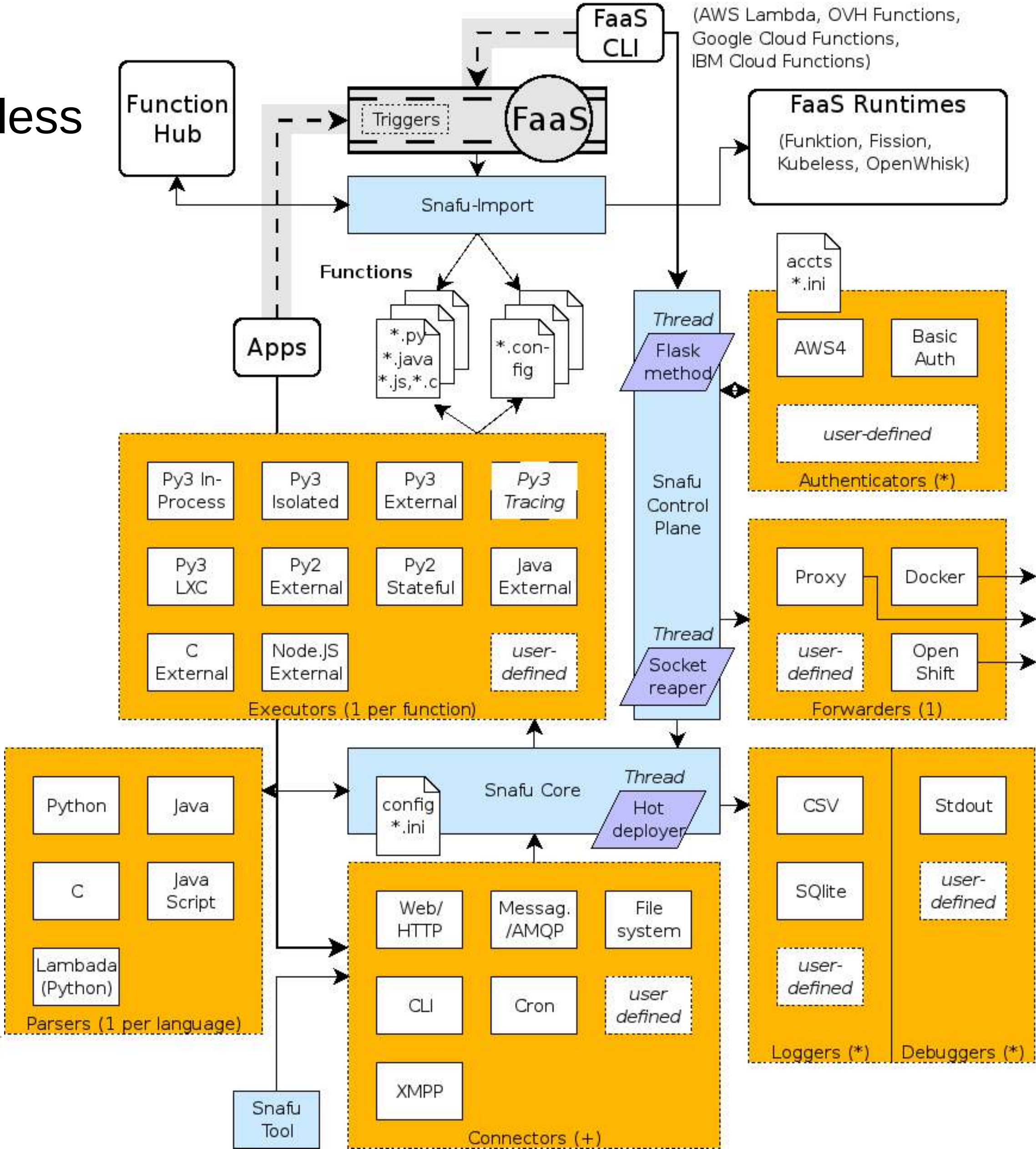
Implementation	Languages	Availability
AWS Lambda	Node.js, Java, Python / C#	Service
Google Cloud Functions	Node.js	Service
Apache OpenWhisk	Node.js, Swift, Docker* / Python	OSS
→ IBM Cloud Functions	-”-	Service
Azure Functions	Node.js, C# / F#, Python, PHP, ...	Service
OVH Functions	Node.js, Python, Perl, Go, Bash	Service
Webtask.io	Node.js	OSS + Service
Hook.io	Node.js, ECMAScript, CoffeeScript	OSS + Service
Effe	Go	OSS
OpenLambda	Python	Academic + OSS
LambCI Docker-Lambda	Node.js	OSS (re-engineered)
Lever OS	Node.js, Go	OSS
Fission	Node.js, Python	OSS
Funktion	Node.js	OSS
Kubeless	Python	OSS
IronFunctions	Node.js, Java, Python, Go, ...	OSS
→ Fn	-”-	OSS

Serverless Computing Tools

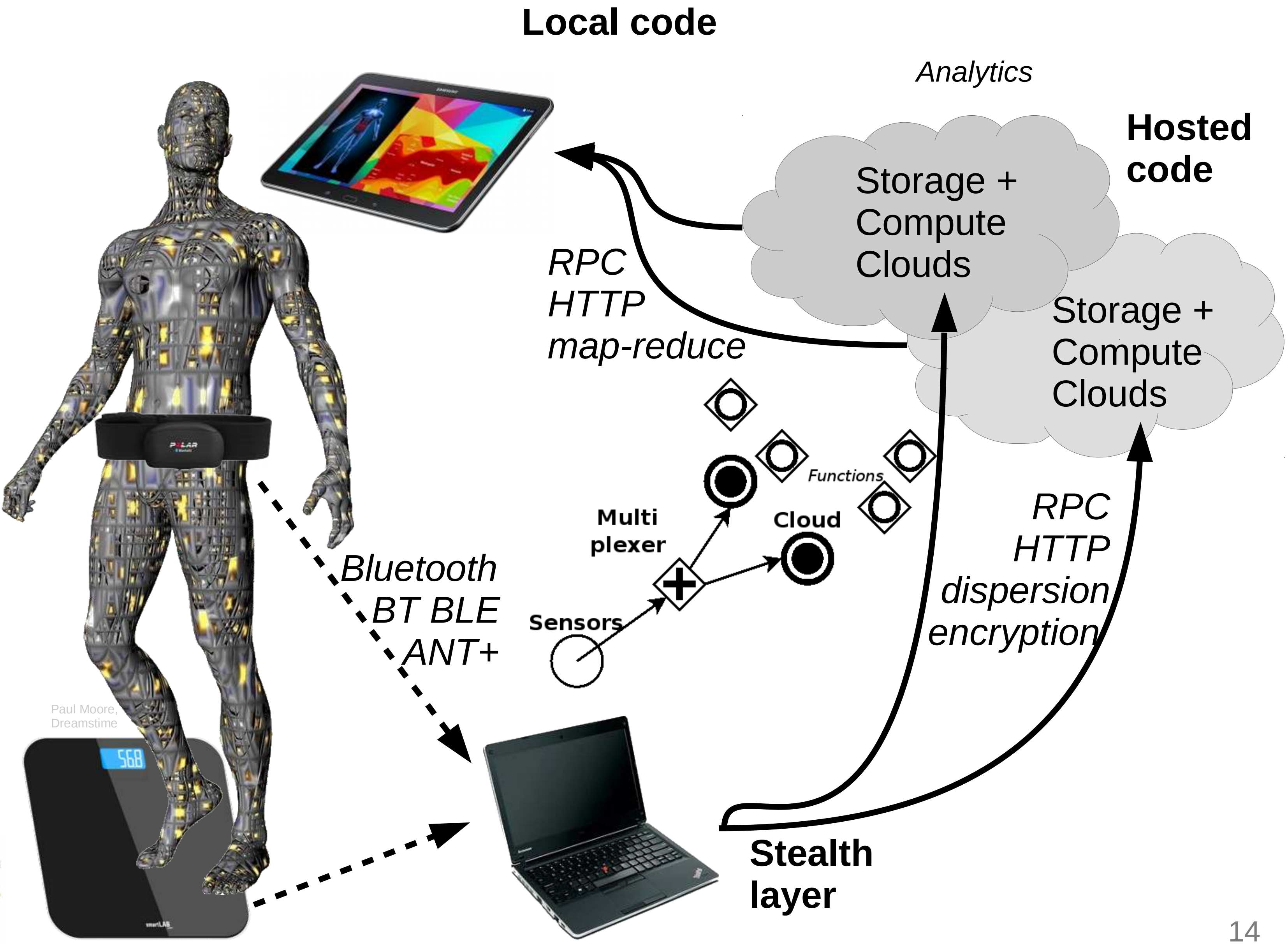
Execution

- “Swiss Army Knife of Serverless Computing“ - Snafu

- Legacy code transformers - Lambada, Termite, FaaS converter



Case: Body Informatics



Local code

Analytics

Hosted code

Storage + Compute Clouds

Storage + Compute Clouds

RPC
HTTP
map-reduce

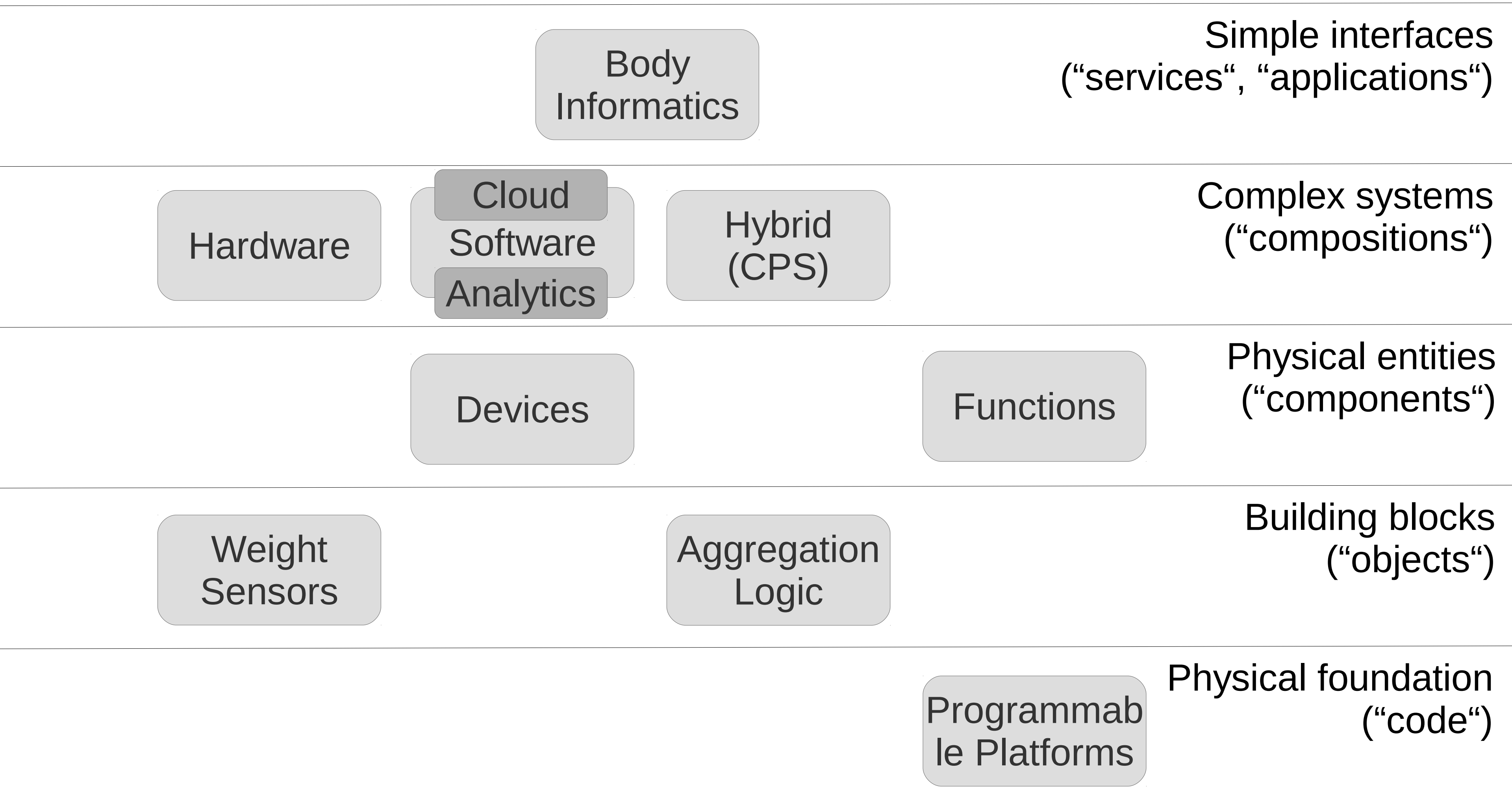
Bluetooth
BT BLE
ANT+

RPC
HTTP
dispersion
encryption

Stealth layer

Paul Moore, Dreamstime

Case: Body Informatics



Body Informatics

Simple interfaces (“services“, “applications“)

Hardware

Cloud

Software

Analytics

Hybrid (CPS)

Complex systems (“compositions“)

Devices

Functions

Physical entities (“components“)

Weight Sensors

Aggregation Logic

Building blocks (“objects“)

Programmab le Platforms

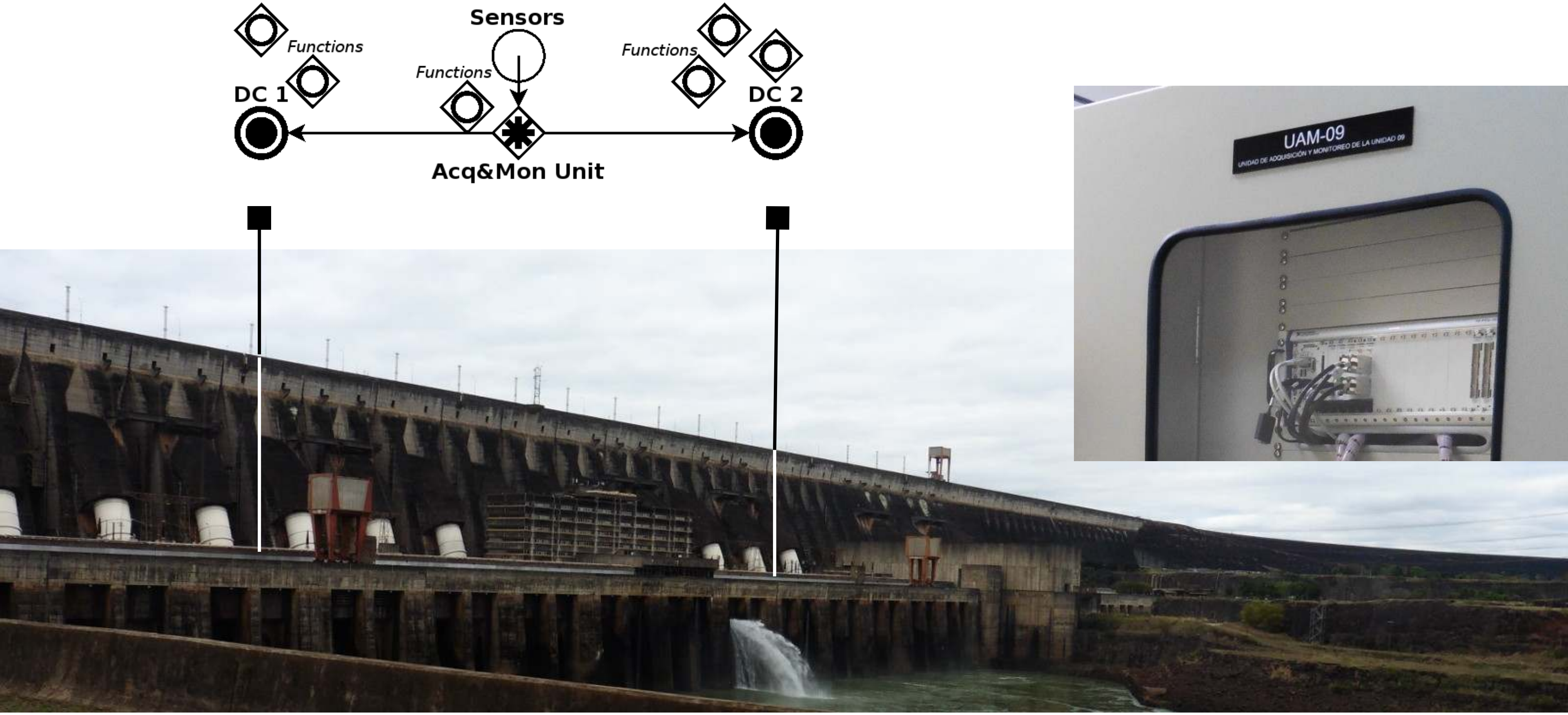
Physical foundation (“code“)

Service Engineering Research Area

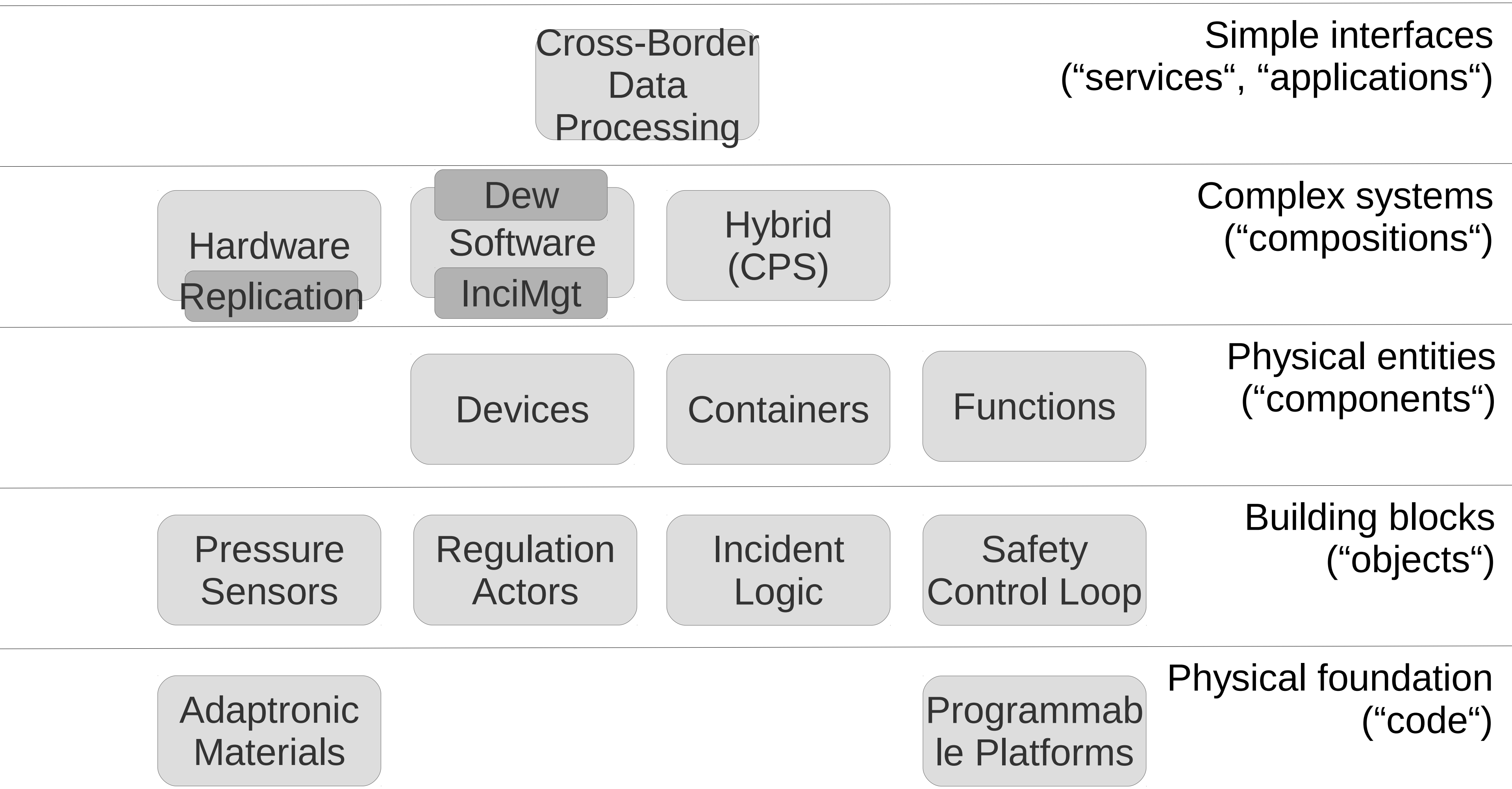
ICCLAB

SPLAB

Case: Cross-Border Data Processing

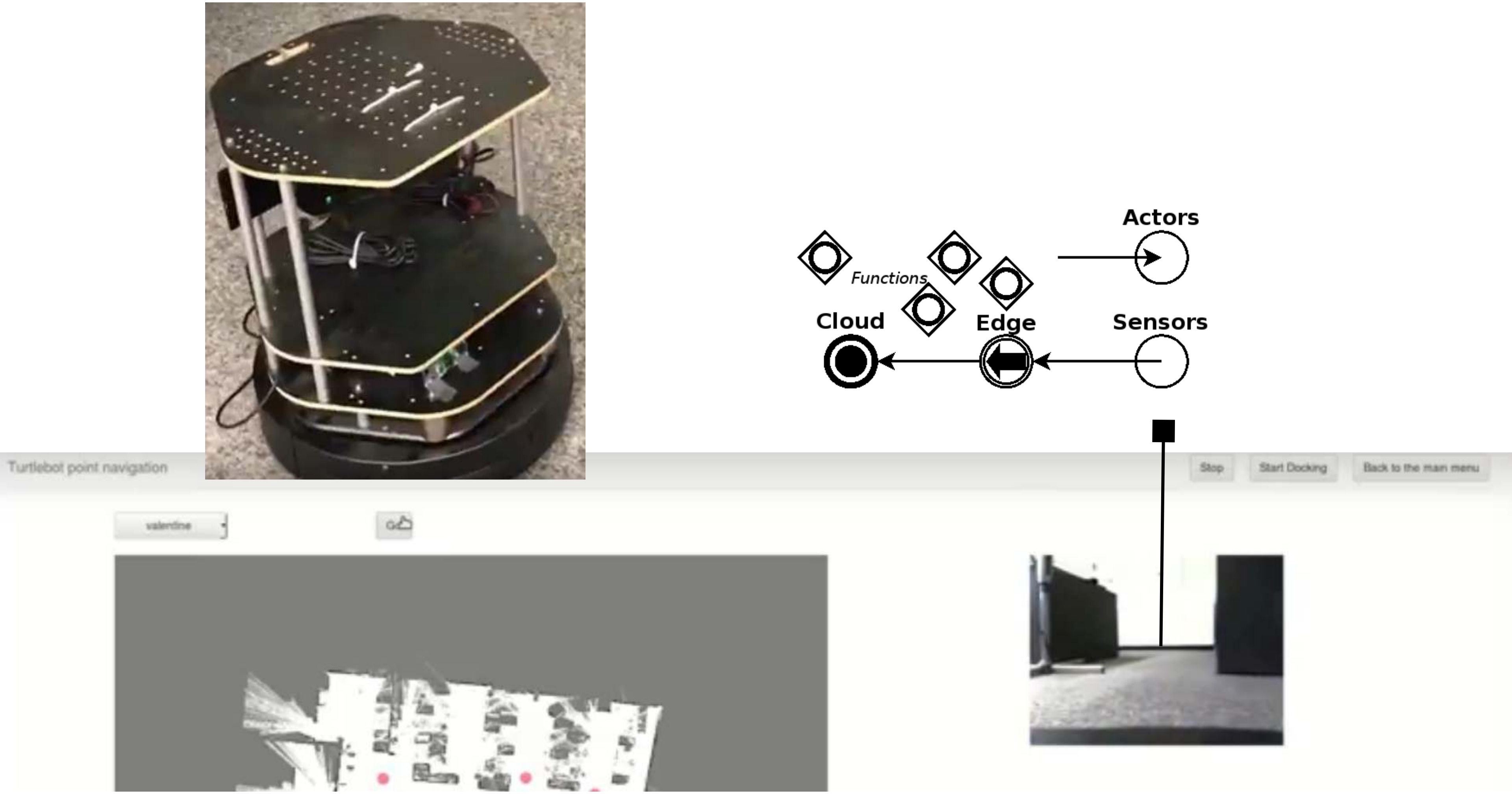


Case: Cross-Border Data Processing



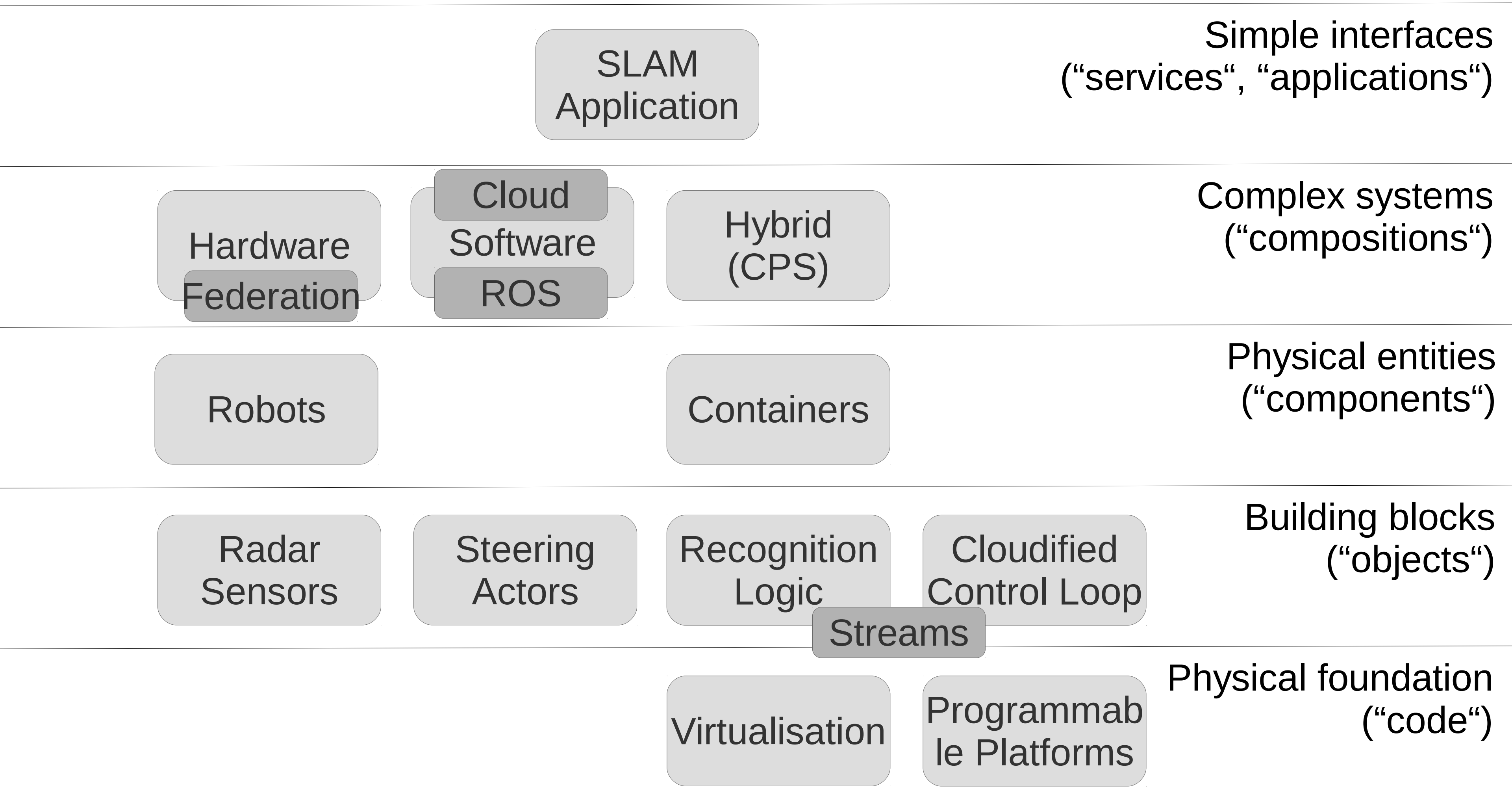
Cross-Border Data Processing
Simple interfaces (“services“, “applications“)
Hardware
Replication
Dew
Software
InciMgt
Hybrid (CPS)
Complex systems (“compositions“)
Devices
Containers
Functions
Physical entities (“components“)
Pressure Sensors
Regulation Actors
Incident Logic
Safety Control Loop
Building blocks (“objects“)
Adaptronic Materials
Programmab le Platforms
Physical foundation (“code“)

Case: Autonomic Robotic Application



[submitted to IROS 2018; based on: G. Toffetti et al., CloudAM 2017]

Case: Autonomic Robotic Application



Science Meets→Needs Industry

3 simple questions (survey-style) to you:

1) Which computing paradigms do you favour or require?

2) What is your approach towards micro/nano-services at the edge?

3) Which tooling is missing to make you/your customers more productive?

Service Engineering
Research Area
ICCLAB
SPLAB