Keynote: Serverless Cyber-Physical Applications

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

Mar 8, 2018 | Science Meets Industry, Dresden, DE
Sensing the World

[Roberto Minerva et al., 2015; IoT Developer Survey 2017; own works 2015/2018]
"... physical devices, the state of which changes according to well-defined rules."

Digital

Liquid flows
Mechanical contraptions
Electric components

Analog

"... a physical system configured so that it is governed by equations identical to the ones you want to solve"
*-Computing Paradigms 1968bis

Approximate Computing

Digital

CoTime-Digital

Hybrid

Analog

[Yannis Tsividis, IEEE Spectrum 02/2018; Steven Fine]
"... to provide decentralized, cloud-friendly, and collaborative micro services to end-users" + competing definitions ca. '16

Required: dedicated research + teaching on *-paradigms

"... simultaneous combination of heterogeneous, hierarchical, and distributed computing resources" + underspecified definitions ca. '12

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

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

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

Services → e.g. nanoservices, functions

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

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

**Complex systems**
- Compositions

**Physical entities**
- Components

**Building blocks**
- Objects

**Physical foundation**
- Code

**IPSC**

- Hardware
- Software
- Hybrid (CPS)
- Hierarchical (CPSoS)
- Robots
- Devices
- Containers
- Functions
- Sensors
- Actors
- Logic
- Control Loop
- Adaptronic Materials
- Reconfigurable Hardware
- Virtualisation
- Programmable Platforms

*ICCLAB SPLAB*
“Serverless” Computing

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

![Diagram of FaaS with Pricing Rules, Environment, Signature, Permissions, Resource Limits, and Data Bindings]

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Languages</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Lambda</td>
<td>Node.js, Java, Python / C#</td>
<td>Service</td>
</tr>
<tr>
<td>Google Cloud Functions</td>
<td>Node.js</td>
<td>Service</td>
</tr>
<tr>
<td>Apache OpenWhisk</td>
<td>Node.js, Swift, Docker* / Python</td>
<td>OSS</td>
</tr>
<tr>
<td>→ IBM Cloud Functions</td>
<td></td>
<td>Service</td>
</tr>
<tr>
<td>Azure Functions</td>
<td>Node.js, C# / F#, Python, PHP, ...</td>
<td>Service</td>
</tr>
<tr>
<td>OVH Functions</td>
<td>Node.js, Python, Perl, Go, Bash</td>
<td>Service</td>
</tr>
<tr>
<td>Webatask.io</td>
<td>Node.js</td>
<td>OSS + Service</td>
</tr>
<tr>
<td>Hook.io</td>
<td>Node.js, ECMAScript, CoffeeScript</td>
<td>OSS + Service</td>
</tr>
<tr>
<td>Effe</td>
<td>Go</td>
<td>OSS</td>
</tr>
<tr>
<td>OpenLambda</td>
<td>Python</td>
<td>Academic + OSS</td>
</tr>
<tr>
<td>LambCI Docker-Lambda</td>
<td>Node.js</td>
<td>OSS (re-engineered)</td>
</tr>
<tr>
<td>Lever OS</td>
<td>Node.js, Go</td>
<td>OSS</td>
</tr>
<tr>
<td>Fission</td>
<td>Node.js, Python</td>
<td>OSS</td>
</tr>
<tr>
<td>Funktion</td>
<td>Node.js</td>
<td>OSS</td>
</tr>
<tr>
<td>Kubeless</td>
<td>Python</td>
<td>OSS</td>
</tr>
<tr>
<td>IronFunctions</td>
<td>Node.js, Java, Python, Go, ...</td>
<td>OSS</td>
</tr>
<tr>
<td>→ Fn</td>
<td></td>
<td>OSS</td>
</tr>
</tbody>
</table>
Serverless Computing Tools

Execution

- "Swiss Army Knife of Serverless Computing" - Snafu
  
- Legacy code transformers - Lambada, Termite, FaaS converter

![Snafu logo]

![Termite logo]

![Podilizer logo]
Case: Body Informatics

Local code

Analytics

Hosted code

Storage + Compute Clouds

RPC HTTP map-reduce

Sensors

Multiplexer

Cloud

Functions

Stealth layer

RPC HTTP dispersion encryption

Bluetooth BT BLE ANT+

Compute Clouds

Storige +

Storige +
Case: Body Informatics

Body Informatics

Hardware

Cloud Software Analytics

Hybrid (CPS)

Devices

Functions

Weight Sensors

Aggregation Logic

Programmable Platforms

Simple interfaces ("services", "applications")

Complex systems ("compositions")

Physical entities ("components")

Building blocks ("objects")

Physical foundation ("code")
Case: Cross-Border Data Processing
Case: Cross-Border Data Processing

Simple interfaces
("services", "applications")

Complex systems
("compositions")

Physical entities
("components")

Building blocks
("objects")

Physical foundation
("code")

Cross-Border Data Processing

Hardware Replication

Dew Software InciMgt

Hybrid (CPS)

Devices
Containers
Functions

Pressure Sensors
Regulation Actors
Incident Logic
Safety Control Loop

Adaptronic Materials

Programmable Platforms
Case: Autonomic Robotic Application

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

- **Hardware Federation**
  - Radar Sensors
  - Steering Actors
- **Cloud Software**
  - ROS
- **Hybrid (CPS)**
- **Simple interfaces**
  - “services”, “applications”
- **Complex systems**
  - “compositions”
- **Physical entities**
  - “components”
- **Building blocks**
  - “objects”
- **Physical foundation**
  - “code”

- **SLAM Application**
- **Containers**
- **Robots**
- **Recognition Logic**
- **Cloudified Control Loop**
- **Streams**
- **Virtualisation**
- **Programmable Platforms**
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?