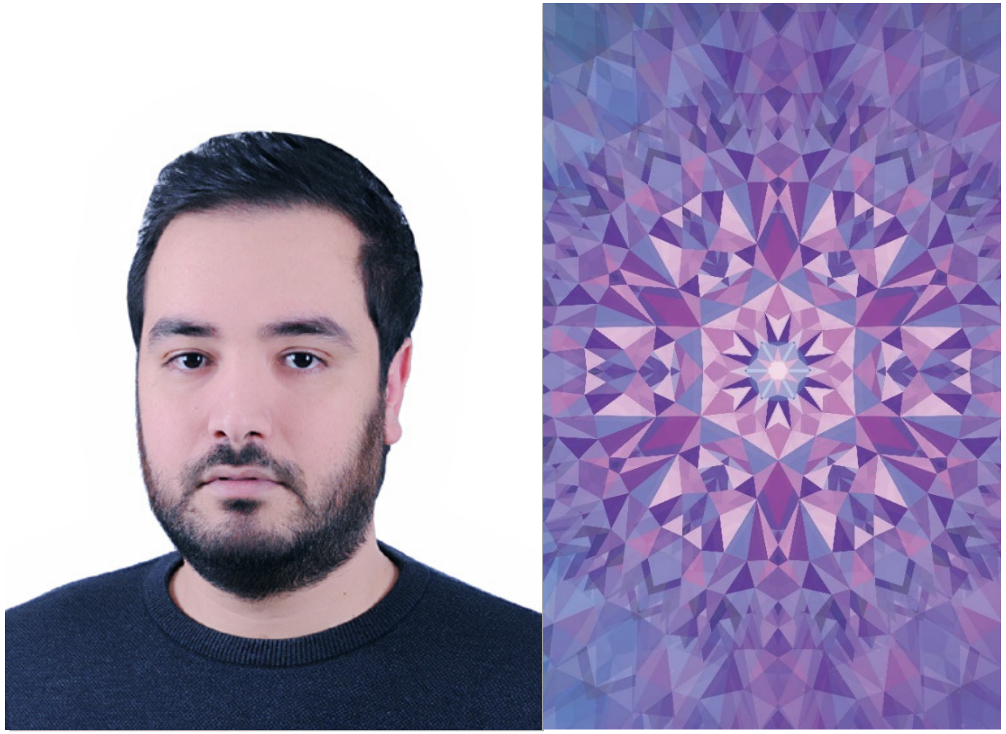
ITU KALEIDOSCOPE
ONLINE 2021

6-10 December 2021

Deviceless: A serverless approach for the Internet of Things

Zakaria Benomar
Department of Engineering,
University of Messina,
Italy



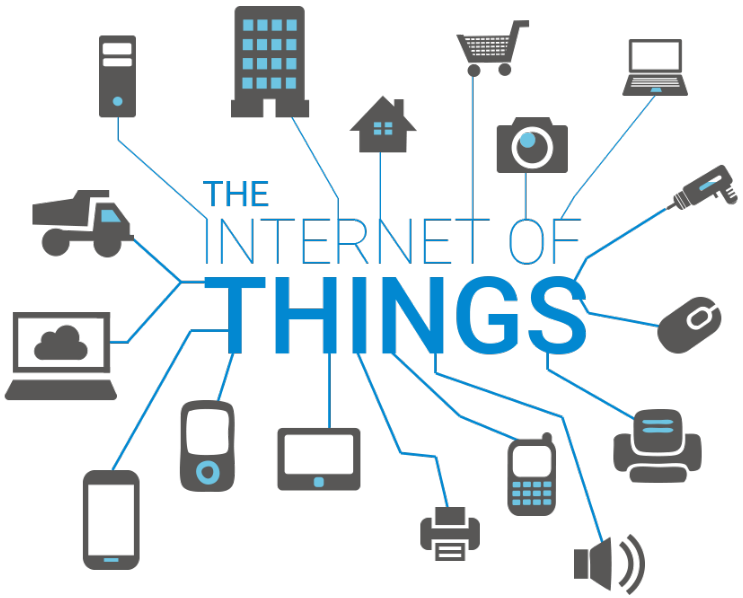
Session: Invited paper

Outlines

- Motivation
- Cloud and IoT integration
- Technology enablers
- Stack4Things architecture
- The rise of Serverless Computing
- Deviceless: extending Serverless to the network edge
- Use case: Node-RED extension
- Conclusions and future works

Motivation

- How to manage in a scalable and powerful way the proliferation of (increasingly smarter) mobile and IoT devices?



IoT ecosystem

- Mobiles
- Cyber Physical Systems
- Smart appliances
- Sensors/Actuators
- Wearables
- Vehicles ...

Motivation

- Microcontroller boards or single board computers with sensors/actuators attached to (analog/digital) gpio pins or serial bus
- A wide range of interfaces

- Smart objects providing interactions with physical world
- Wi-fi/bluetooth connectivity

- Smartphones with sensors on-board
- Wi-fi/ bluetooth/ 3-4G connectivity

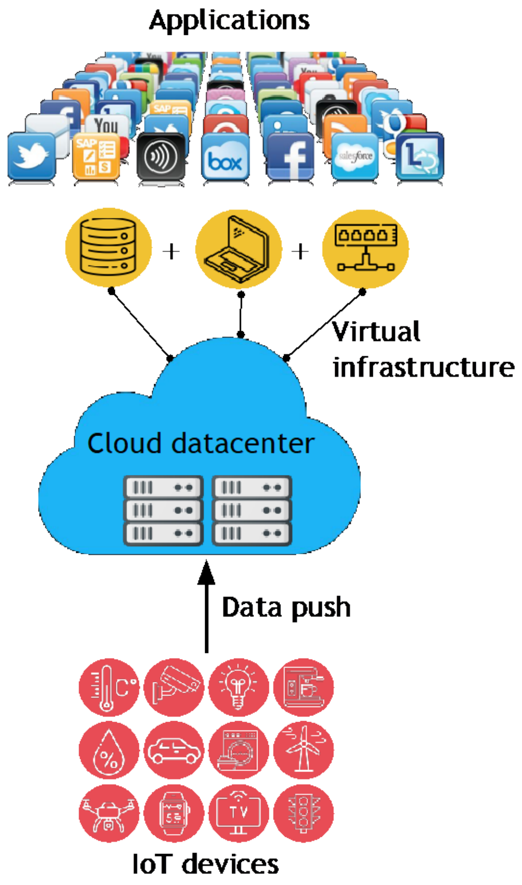
Cloud and IoT integration

(1) Data-oriented approach

- The Cloud is used to deal with IoT data management.
- IoT devices send data to the Cloud.
- The Cloud is leveraged as is.
- Apps are built on top of standard cloud facilities (e.g., VMs, storage, networking).
- Apps make use of stored (non-real time) IoT data
- the only operations permitted are data manipulation ones (no interaction with the devices).

→ What about actuation operations?

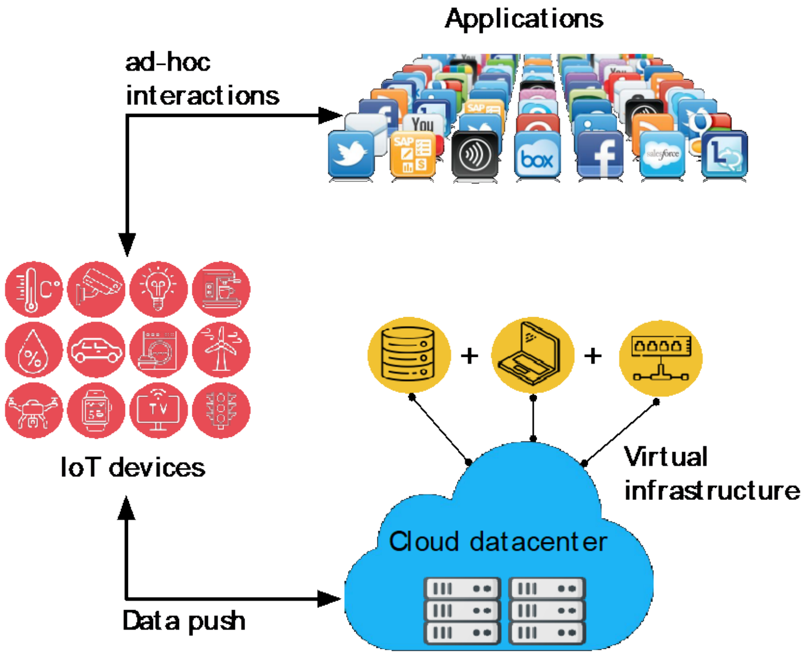
- Data-centric-oriented solutions are based on sending all the generated data towards a data center (such a solution can incur significant operational expenditure in terms of bandwidth, storage and processing cost).



Cloud and IoT integration

(2) Application-specific (vertical) approach

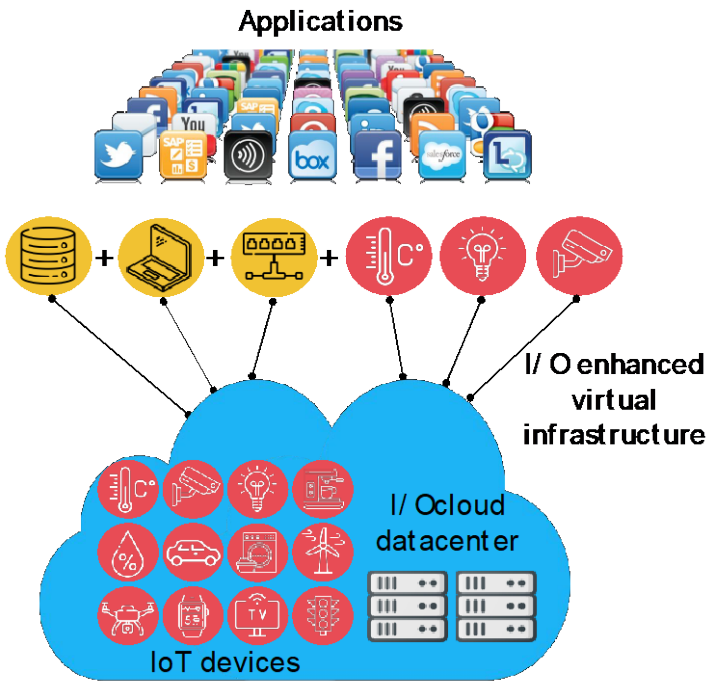
- The application uses ad-hoc mechanisms to interact with IoT devices (SDK-based solutions).
- No explicit interactions between Cloud components and IoT infrastructure.
- Apps developers cannot share the IoT infrastructure.
- Each user has to set up its own infrastructure (CAPEX/OPEX problems).
- Authorizations to deploy IoT nodes in public domains for large-scale deployments can be hard to acquire (e.g., smart cities).
- The approach is based also on sending all the IoT data to the Cloud.



Cloud and IoT integration

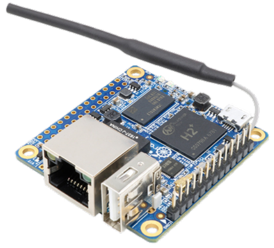
(3) full thing "cloudification" (I/Ocloud)

- Adapt the Cloud "as-a-Service" approach to IoT.
- Offer IoT infrastructures as a extension of a Cloud deployment.
- Cloud users that have access abstracted VMs can also access Virtual IoT nodes with attached sensors/actuators.
- Separation of concerns between infrastructure and application (when needed) --> offer virtual IoT nodes with virtualized sensors and actuators.
- Virtual IoT nodes can be deployed at the network edge (on top of physical IoT nodes).
- Device computation offloading.
- Enabling a low-level abstraction of the IoT nodes and resources (this is important for applications code portability)



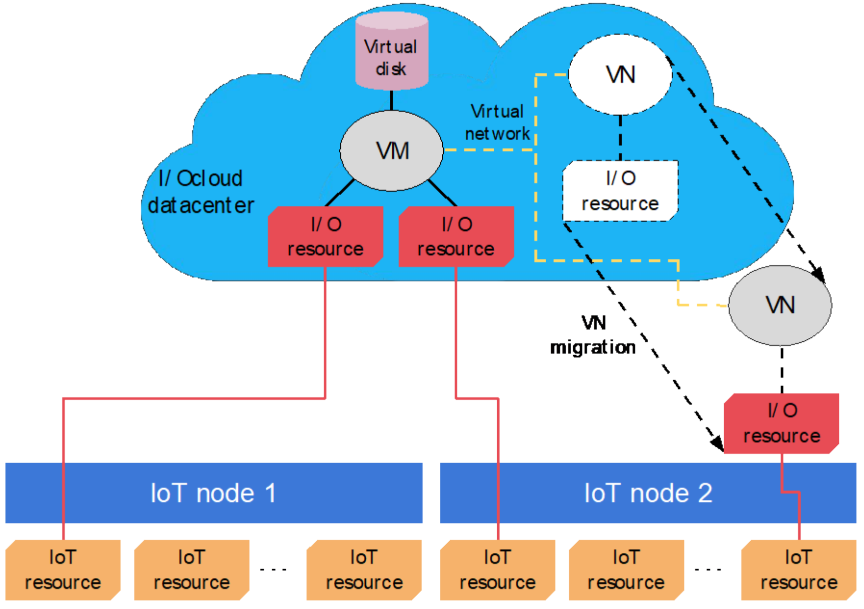
Cloud and IoT integration

(3) full thing “cloudification” (I/Ocloud)

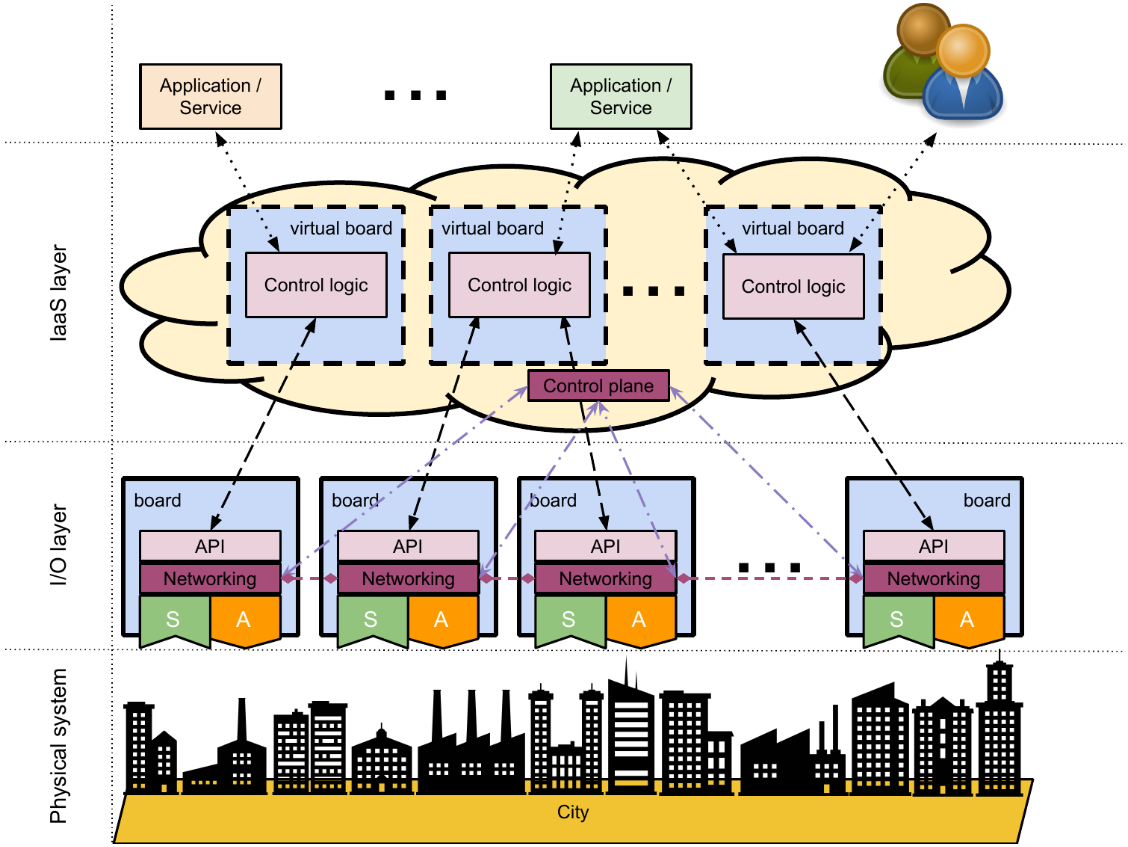


- MPU-powered boards with Linux-based OSs.
- Physical pins are exposed through the File system (standard system calls as for regular files: write and read operations).

- Virtual IoT nodes with Virtual File systems (clone of the real File system) as:
  - VMs with attached I/O pins.
  - VNs (lightweight containers) with attached I/O pins
- Ready to use Cloud resources (networking, storage, compute…)
- VNs can be instantiated at the network edge to meet applications demands (e.g., latency, privacy/security).
- Containers migration (from Cloud to Edge and vice versa)



Cloud and IoT integration



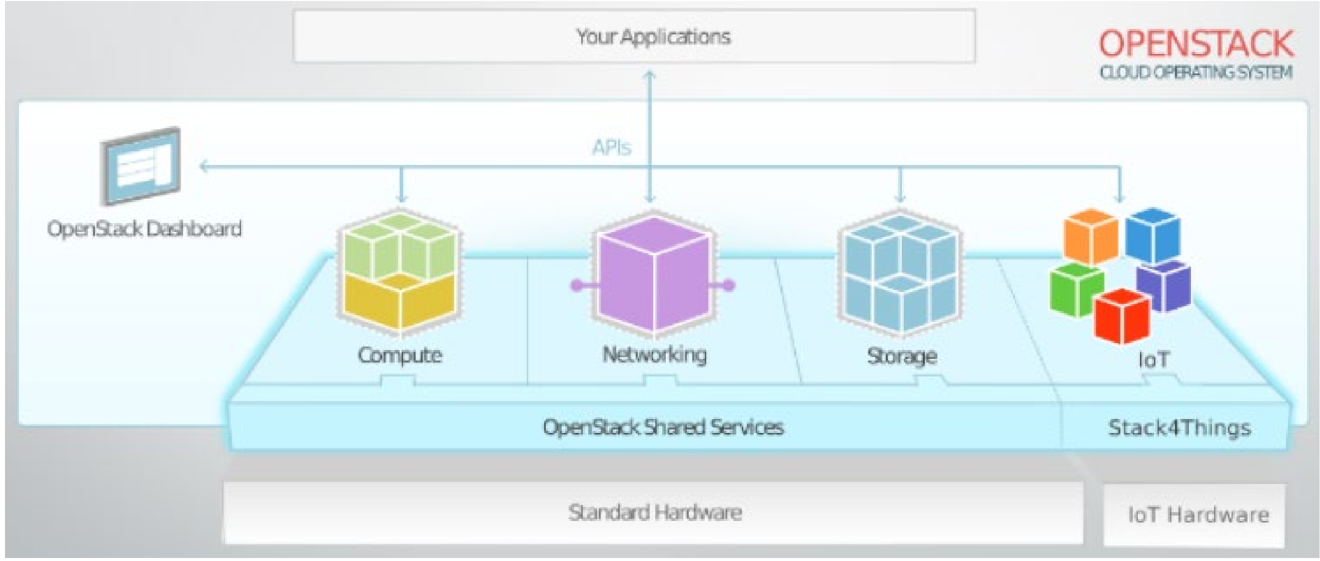
The Software Defined City paradigm

- Analogy with Software Defined Networking (SDN).
- Separation between the **I/O layer (data plane)** and the **Cloud layer (control plane)**.
- Extends the SD* approach to a cyber city system to enable the re-configuration of the underlying infrastructure.
- Several controllers exploit and implement the requested node topologies through generalized rules and according to predefined policies.

Technology enablers

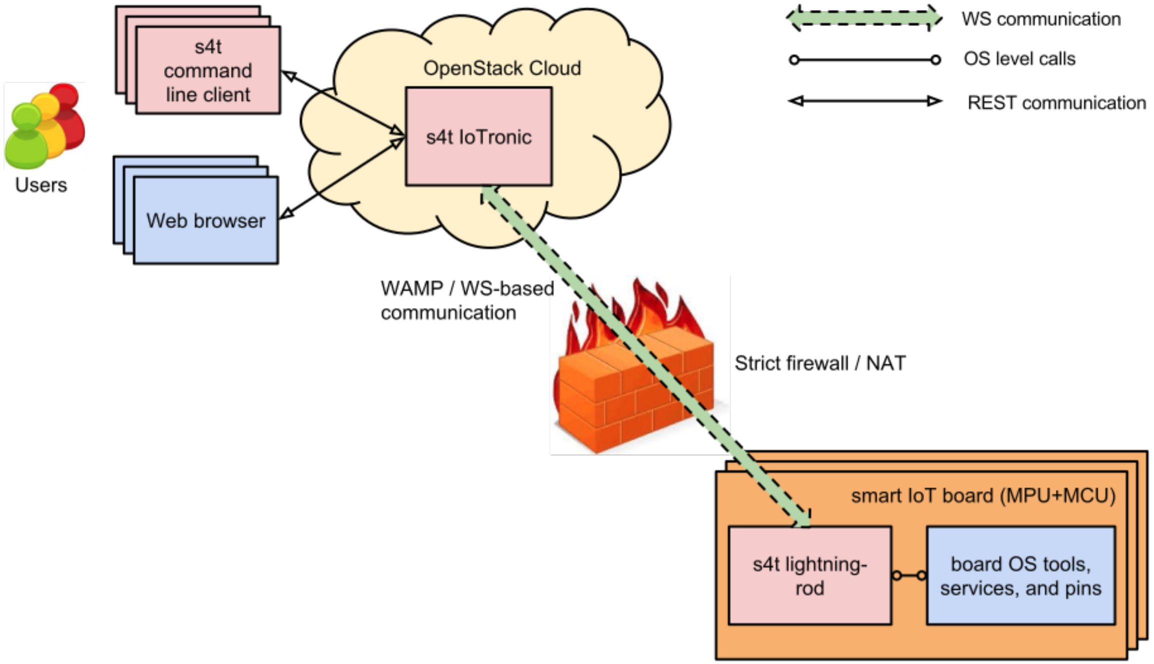
Technology enablers

- IoT resource management service for OpenStack Clouds
- OpenStack (unofficial) project
    - https://launchpad.net/iotronic
    - https://opendev.org/x/iotronic



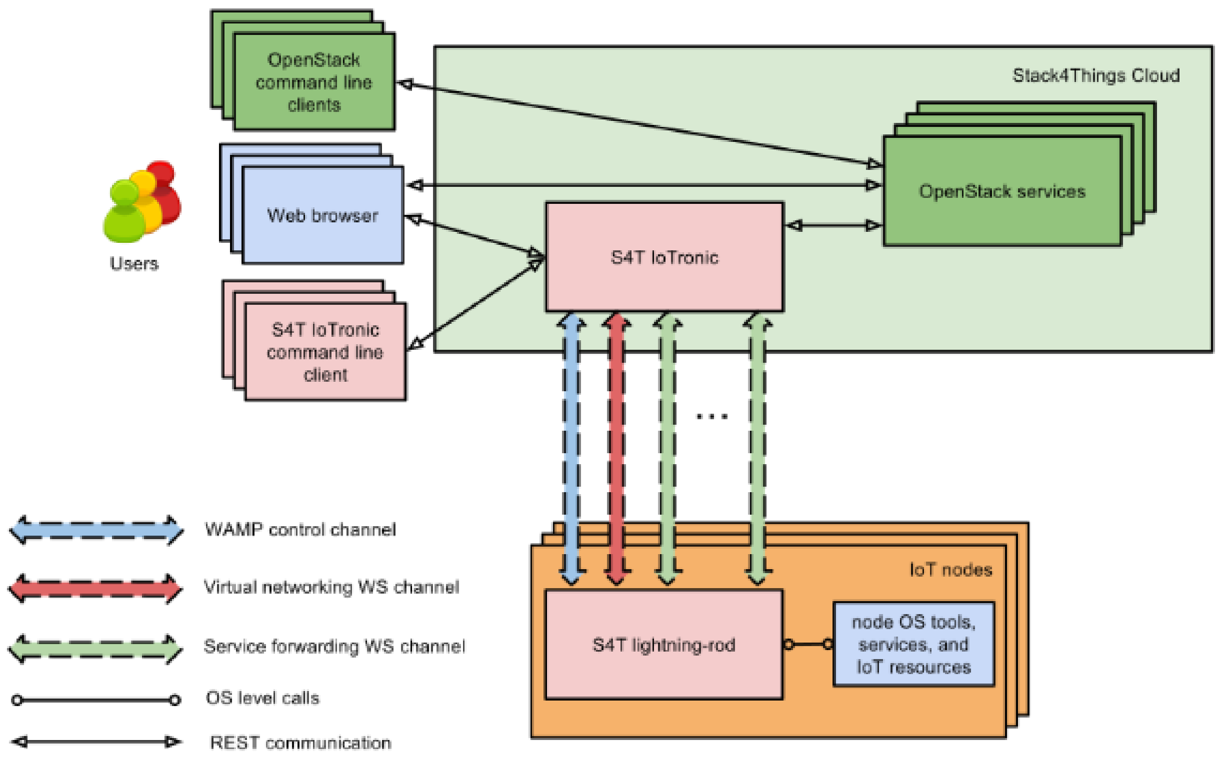
Stack4Things architecture

- A Cloud OpenStack compatible subsystem called ***IoTronic***.
- A Device-side agent named ***Lightning-Rod***.
- Communications between the Cloud and the devices are based on WebSocket tunnels with a reverse tunnelling mechanism to bypass NATs and firewalls.



Stack4Things architecture

- Use of a software probe on the device-side (lightning-rod)
- OpenStack compliant service (IoTronic)
- Use of WAMP and plain WebSocket control channels
- REST interfaces

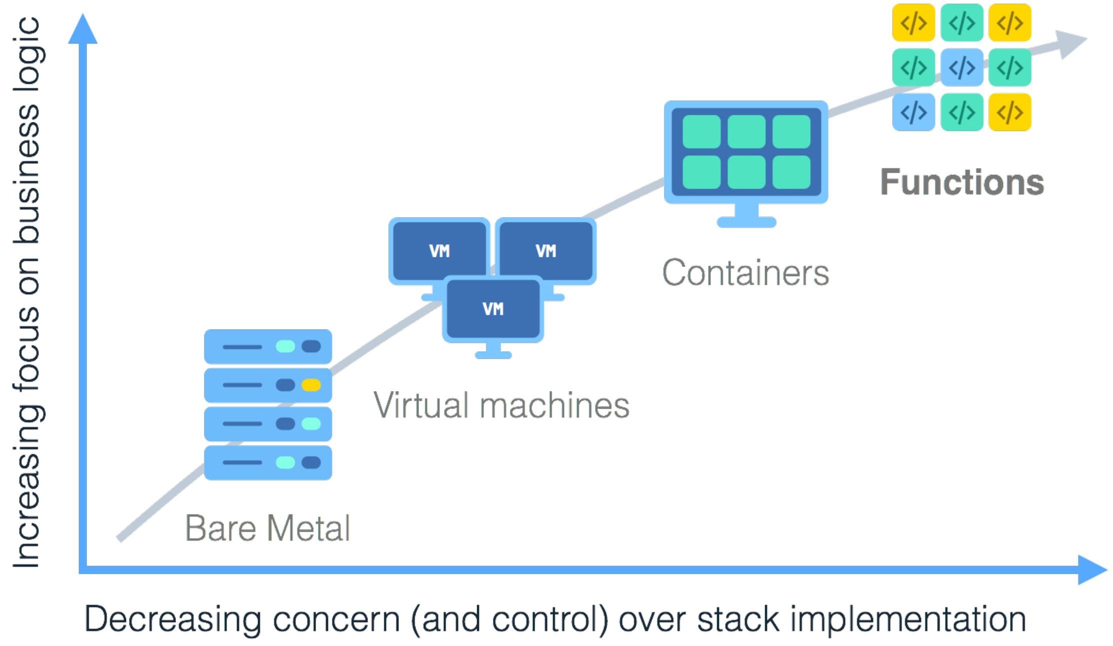


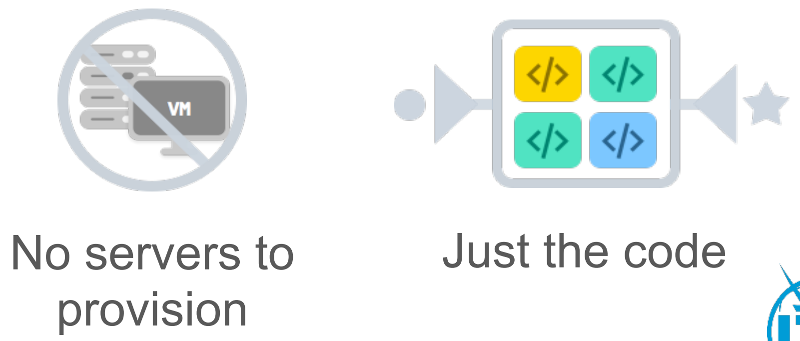
The rise of Serverless Computing

- In the IaaS model, the user has to manage the server configuration.
- The provisioning period of VMs and containers is long even if the tasks to handle can be short in time.

→ **Significant increase in terms of cost.**

- Serverless computing:
  - Runs code in response to events (event-programming model) --> **Think about IoT**
  - Worry-less about servers (i.e., scalability).
  - Users needs only to write the functions. All the rest, is managed by the Cloud provider.
  - The functions run on **event-triggered** and **ephemeral** containers (may only last for one invocation).





The rise of Serverless Computing

- Serverless is a cloud-native platform for **short-running**, **stateless computation** and **event-driven** applications which **scales up and down** instantly and automatically and charges for actual usage at a **millisecond granularity.**

- Why is Serverless attractive?
  - Making app development & ops dramatically faster, cheaper, easier.
  - Drives infrastructure cost savings.

- Comparison (based on AWS Frankfurt, Germany):
  - AWS VM with a Linux OS: 1 vCPU, 2 GB memory: **$25.00 for one month**
  - AWS Lambda function execution (1ms) with 128 MB of memory: **$0.0105 for 5 million function execution.**

Source: https://aws.amazon.com

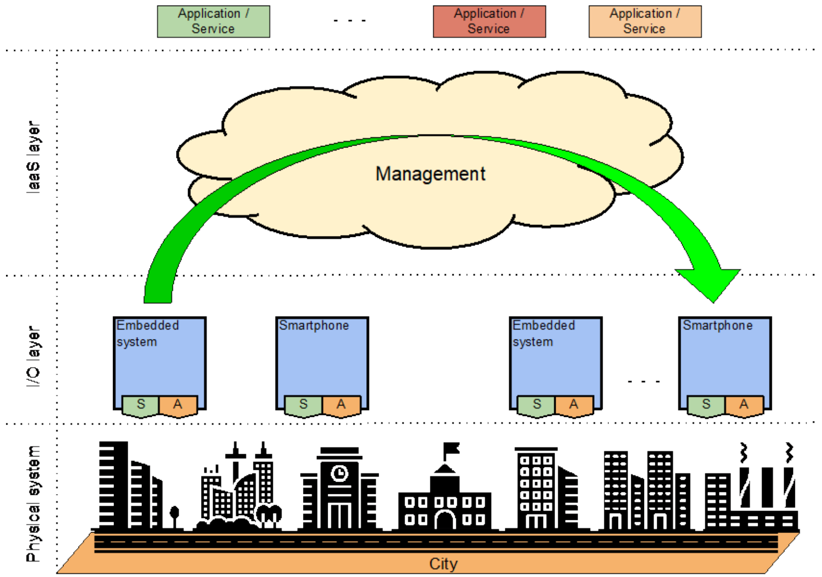
	On-prem	VMs	Containers	Serverless
Time to provision	Weeks-months	Minutes	Seconds-Minutes	Milliseconds
Utilization	Low	High	Higher	Highest
Charging granularity	CapEx	Hours	Minutes	Blocks of milliseconds

Source: Jason McGee, IBM; Serverless Conference 2017.

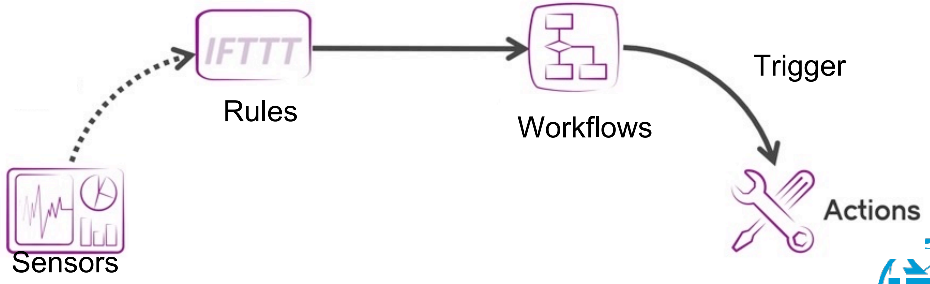
Deviceless: extending Serverless to the network edge

- Our approach, **Deviceless** is meant to extend the Serverless computing model down to the network edge.
- Use a Serverless-like methods to interact with remote sensors/actuators.
- In addition to IoT-as-a-Service provided by the I/Ocloud paradigm, a user can use Deviceless.
- Provide event-programming model for I/Ocloud without resorting to VNs provisioned for long periods (when not needed).
- Deviceless functions/actions runs on ephemeral stateless containers (may only last for one invocation).
- May help in the establishment of policies for "**closing the loop**" for the applications.
- Configuring triggers for a range of (dispersed) actuators based on sensing activities from geographically distributed sensing resources.

Mechanisms to rewire such a "nervous system" into a number of elastic control loops.

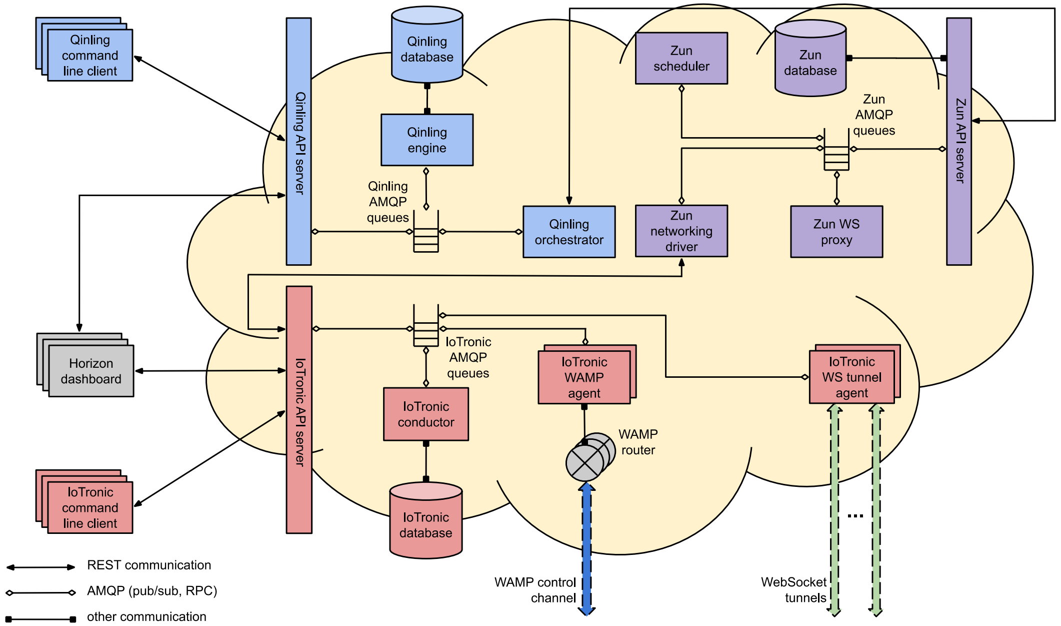


SD City as closed-loop system

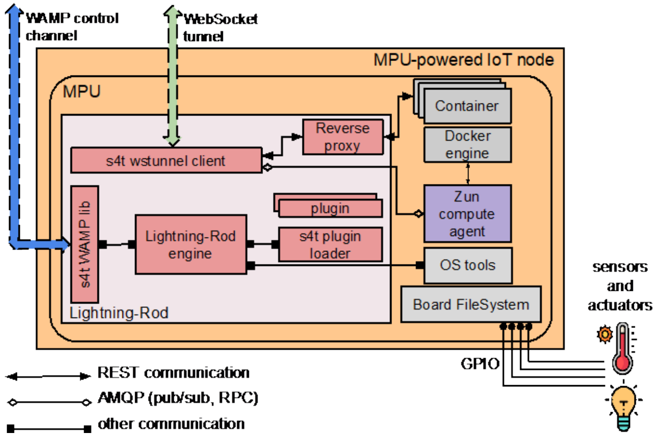


Deviceless: extending Serverless to the network edge

- We extended the functionalities of **Qinling** (the Serverless subsystem in OpenStack) and **Zun** (the subsystem responsible of managing containers).
- Integration of Qinling and Zun within S4T.
- Manage functions execution on IoT nodes.
- Use **IoTronic** as a new **networking driver** for Qinling:
  - In Cloud-based deployments, Qinling uses the overlay networking IP addresses to reach out the containers.
  - In our approach, the containers where functions should be executed are deployed at the network edge (behind NATs and firewalls).
  - IoTronic uses Websocket tunnels to use the remote containers and use an new id to identify them.



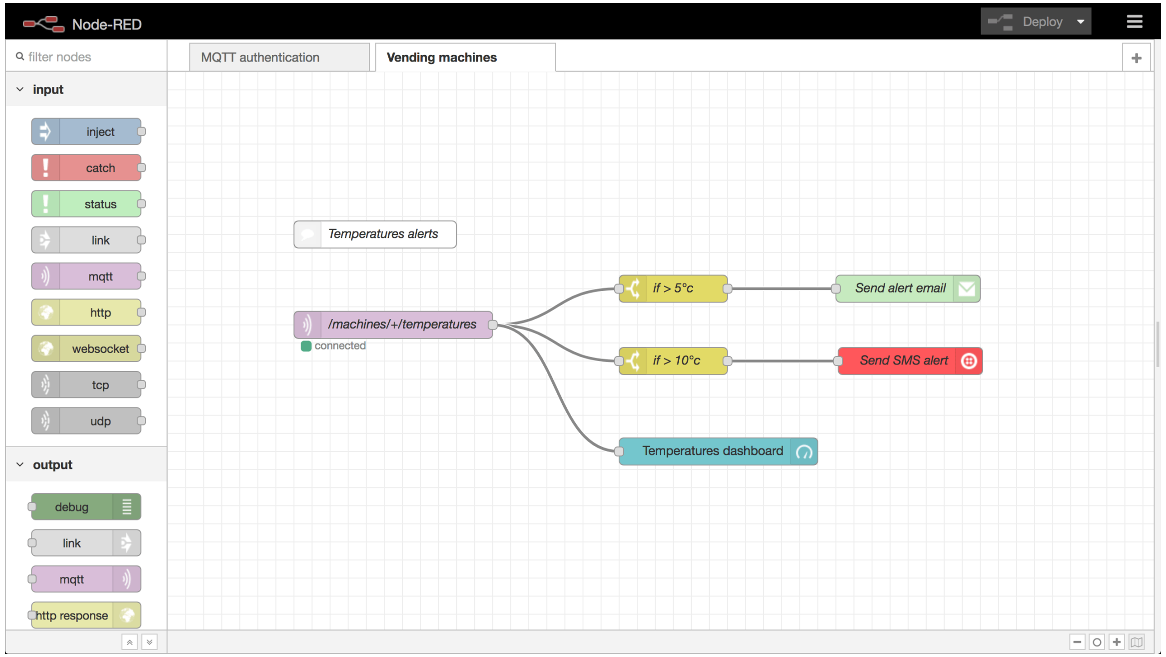
Cloud-side architecture.



Board-side architecture.

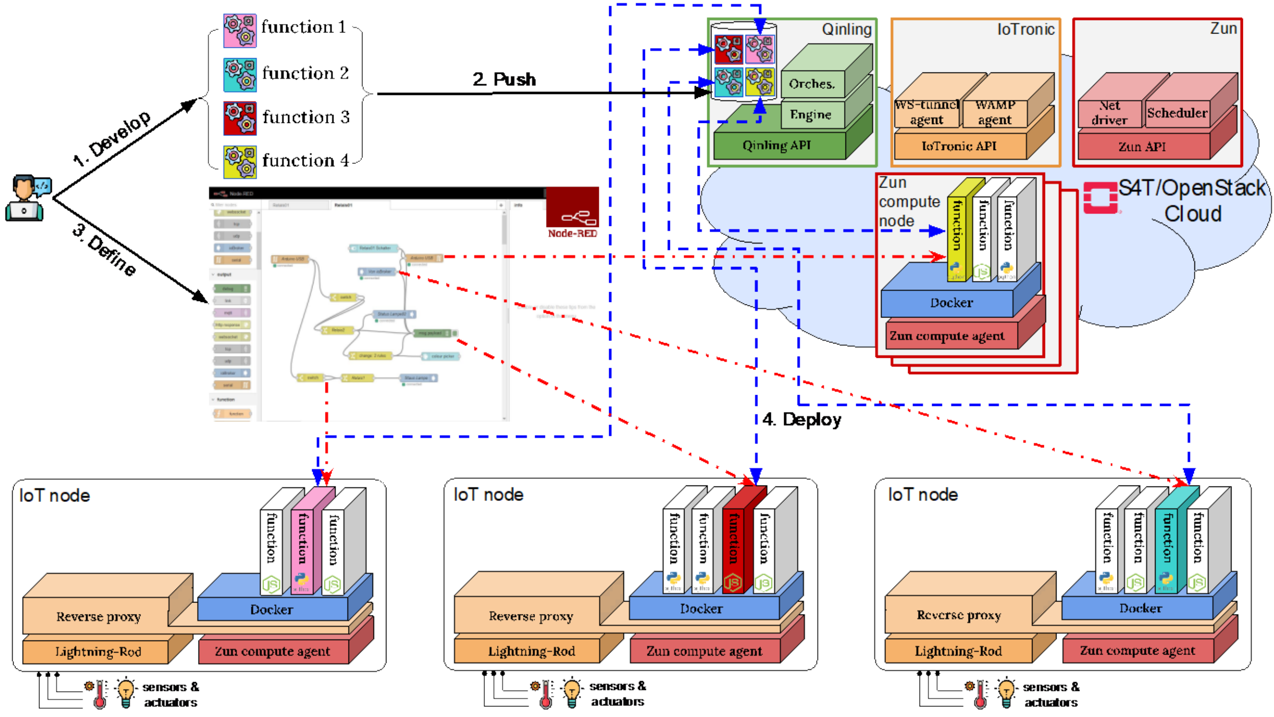
Use case: Node-RED extension

- **Node-RED** is a is a flow-based development tool for visual programming for wiring, hardware devices, APIs and online services.
- It provides a browser-based flow editor to create JavaScript actions.
- Users can create complex workflows by minimal effort (drag and drop nodes from the left panel).
- While Node-RED have been found to be useful on its own as data flow tool, several IoT scenarios require the coordination of computing resources across a distributed environment: on **servers**, **gateways** and **devices** themselves.
- Node-RED cannot deal with workflows using a distributed infrastructure.



Use case: Node-RED extension

- We exploited the Deviceless paradigm to extend the capabilities of the Node-Red flow-based development tool for visual programming.
- We added a new type of nodes that exploit, underneath, the functions managed by Qinling.
- User can design workflows/pipelines among IoT devices deployed at the network edge.
- The solution can also be used in conjunction with the Cloud-based Serverless computing model.
- Instead of using only JavaScript to create actions/functions, our approach extends the Node-RED programming languages choices to include other languages such as Python.
- Uses don’t have to setup the Node-RED service on the IoT devices.



Use case: Node-RED extension

	Idle		10 requests	
	CPU	RAM	CPU	RAM
Reverse proxy	0%	1.1%	3.4%	1.1%
WS tunnel client	0%	4.1%	4.2%	4.1%
Zun agent	0%	9.7%	0%	9.7%
Lightning-Rod	0%	0.8%	0%	0.8%
Total	**0%**	**15.7%**	**7.6%**	**15.7%**

- CPU and RAM usage on a Raspberry Pi.
- The function used is a simple print on the screen.

Conclusion and future work

- Presentation of the I/Ocloud approach.
- Introduction of the Deviceless paradigm.
- Extend Node-RED to use distributed IoT devices based on the **Deviceless** paradigm.
- The approach presented has been used to create monitoring applications as well as genomic analysis.
- The “supplement 49 to ITU-t y.3500-series” outlines the efficiency of adopting the Serverless computing model at the network edge.
- As future work, we would like to adapt the architecture of S4T and use it with the the **ETSI MEC** architecture to orchestrate the execution of functions on the IoT devices.

ITU KALEIDOSCOPE
ONLINE 2021

Thank you!