MICROEJ®

Java Technology in the Small IoT Devices

Eclipse IoT Day, Grenoble, March 9th 2017

https://projects.eclipse.org/projects/iot.edje

ABOUT THE PRESENTER

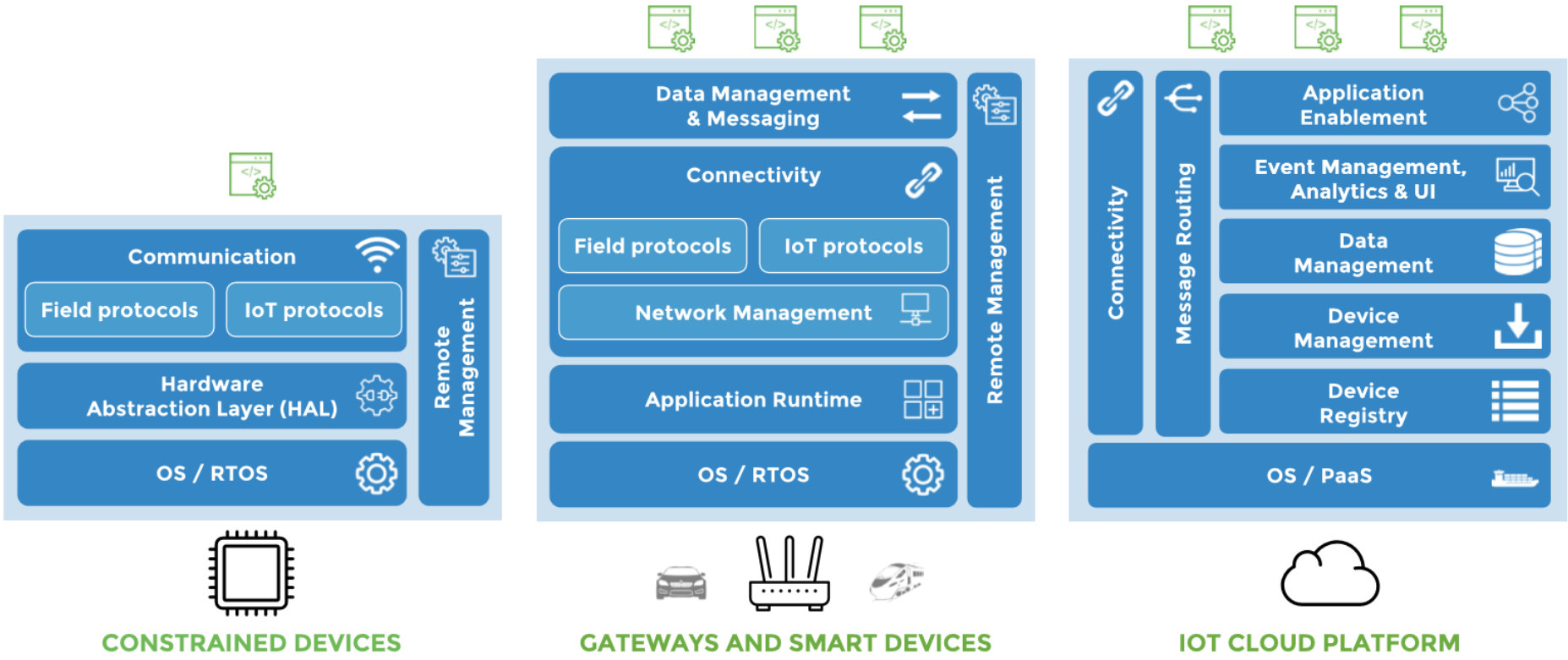
Laurent Lagosanto

Senior Architect at MicroEJ®

18 years of “in the Small” activities, mostly about Java Technology

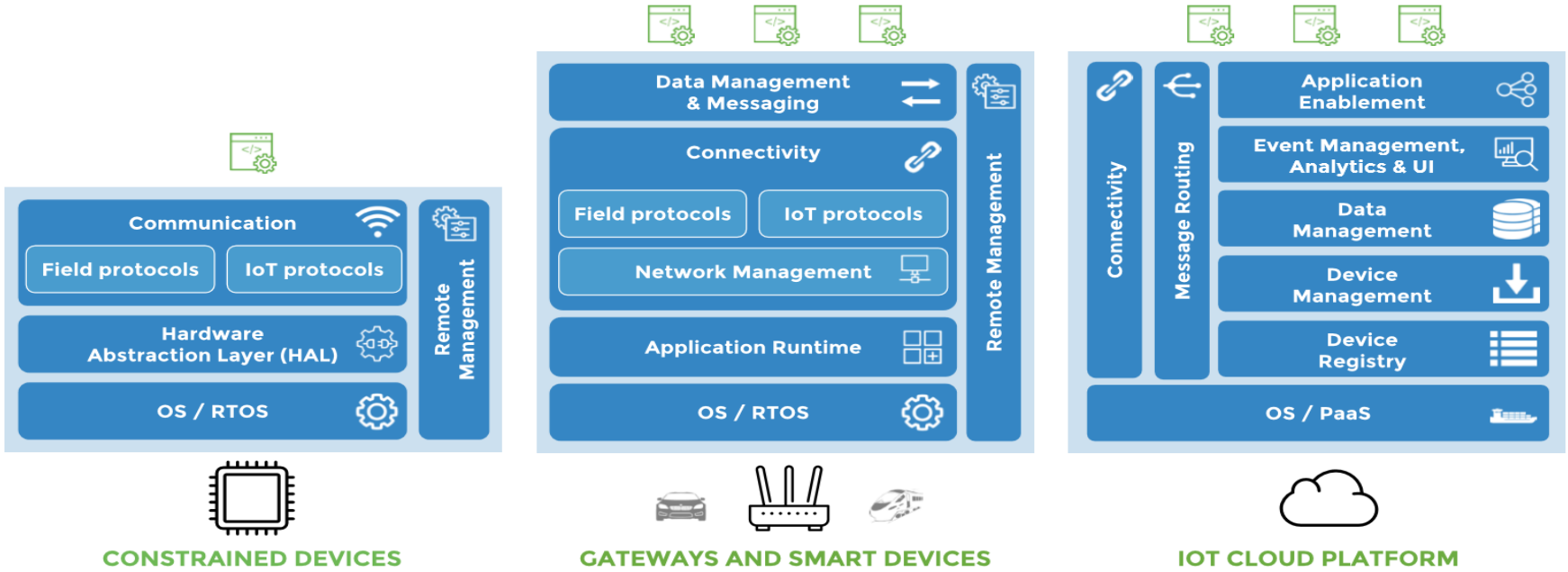
laurent.lagosanto@microej.com

ECLIPSE IOT : THREE VERTICALS



Checkout Eclipse IoT Whitepaper from 2016 Q4

JAVA TECHNOLOGIES ARE PERVASIVE, REALLY ?



CONSTRAINED DEVICES	GATEWAYS AND SMART DEVICES	IOT CLOUD PLATFORM
Monolithic Firmwares	Java Applications	Java Web Services
RTOS + Apps	Java OSGi Containers	Java EE Engines
Native Protocol Stacks	Java IoT Protocol Stacks	Java IoT Protocol Stacks
Eclipse IDEs	Eclipse IDEs	Eclipse IDEs

JAVA TECHNOLOGIES IN THE SMALL, WHY NOT?

« Too big »

« Too complex »

« Too slow »

« Not secure »

« I don’t need dynamic loading »

« I don’t need a Java Virtual Machine, I have Linux »

LET'S DEBUNK SOME MYTHS…

Size can be controlled, if you pick the right features & APIs (ever heard of Java Card ?)

You are NOT forced to interpret bytecodes, you know !

Linux cannot go NOT everywhere ! (Cortex-M anyone ?)

No dynamic loading ? So your product software will never be updated ?

A Virtual Machine provides sandboxing, even if you don't have an MMU

Bytecodes can be downloaded safely : they can be verified

An automatically managed memory is safer, more efficient, than if statically allocated

AND SOME OTHER ADVANTAGES:

Java Language is Object Oriented and helps build Services Oriented architectures

(think about what you can do with Class.forName(...) or, instanceof ...)

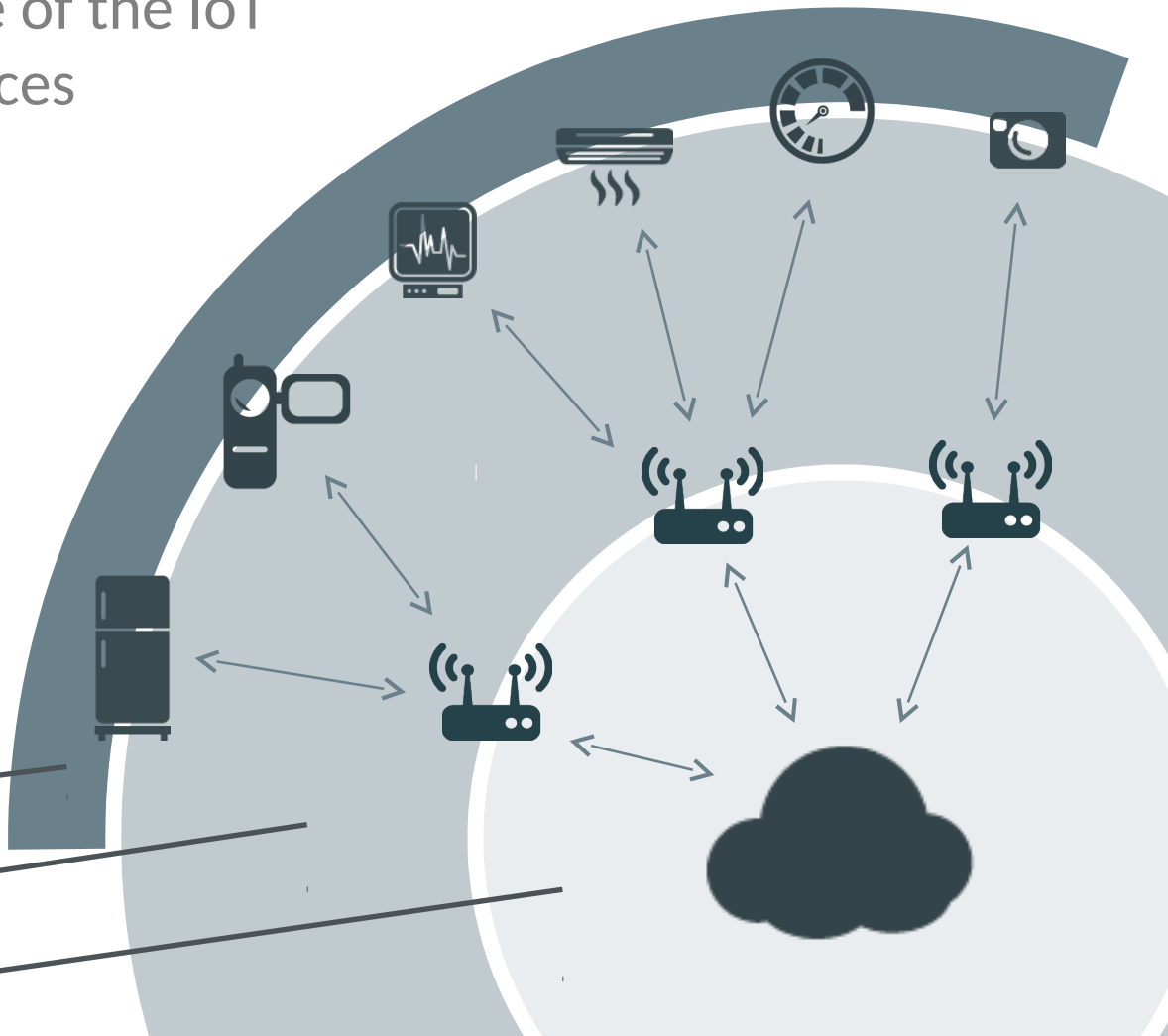
Java APIs are well defined, and cover a wide functional range

(Language level APIs, I/Os, Collections, Net, TLS, Junit, Crypto, ...)

Java Technologies are part of a huge ecosystem : developers, tools, trainings, open source projects and communities

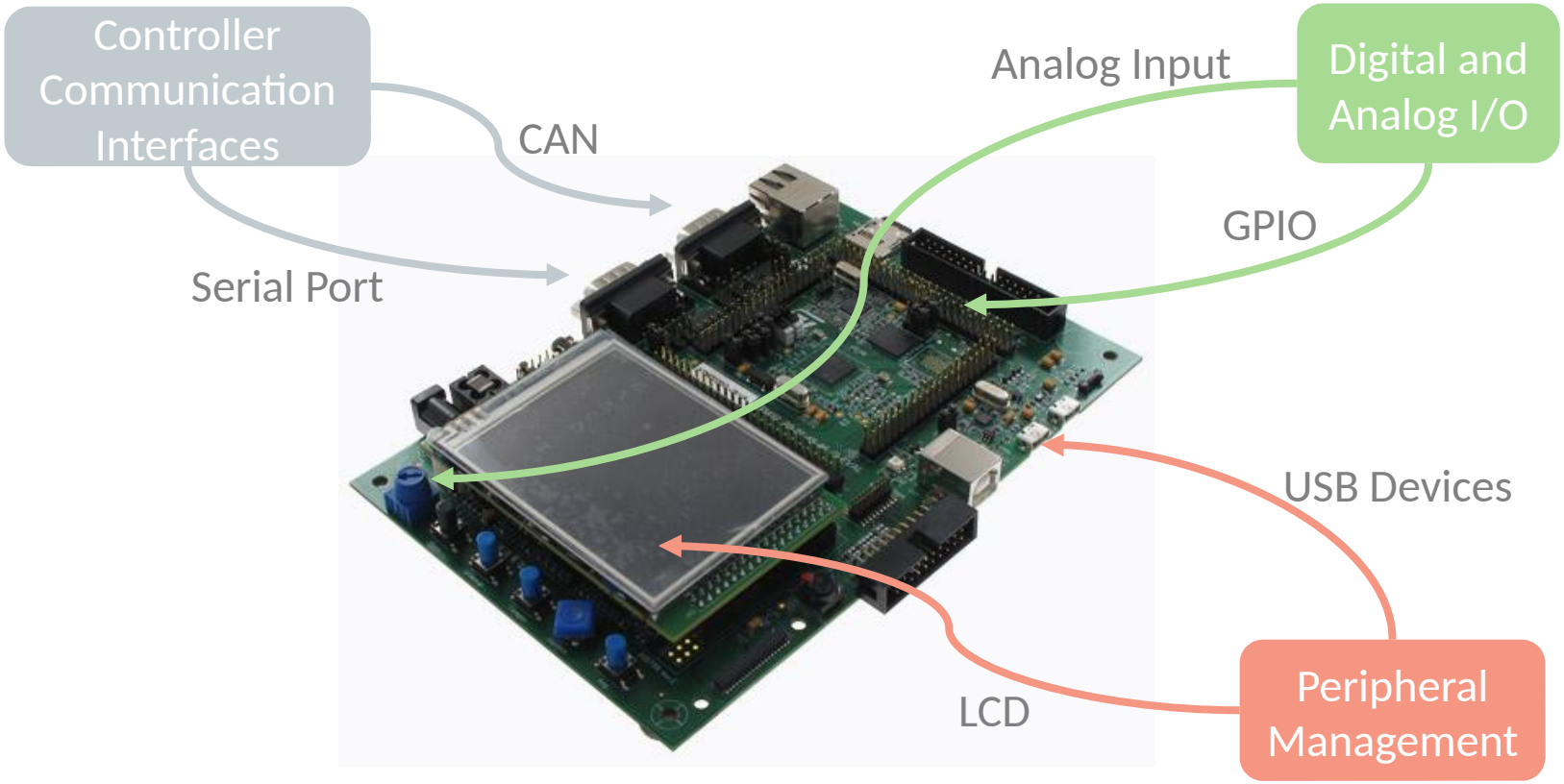
MICROEJ AND ECLIPSE : THE EDJE PROJECT

A Java library at the edge of the IoT
for sensor hubs and devices



EDJE API INITIAL FOCUS : PERIPHERALS AND I/O

Edje focuses on the following aspects

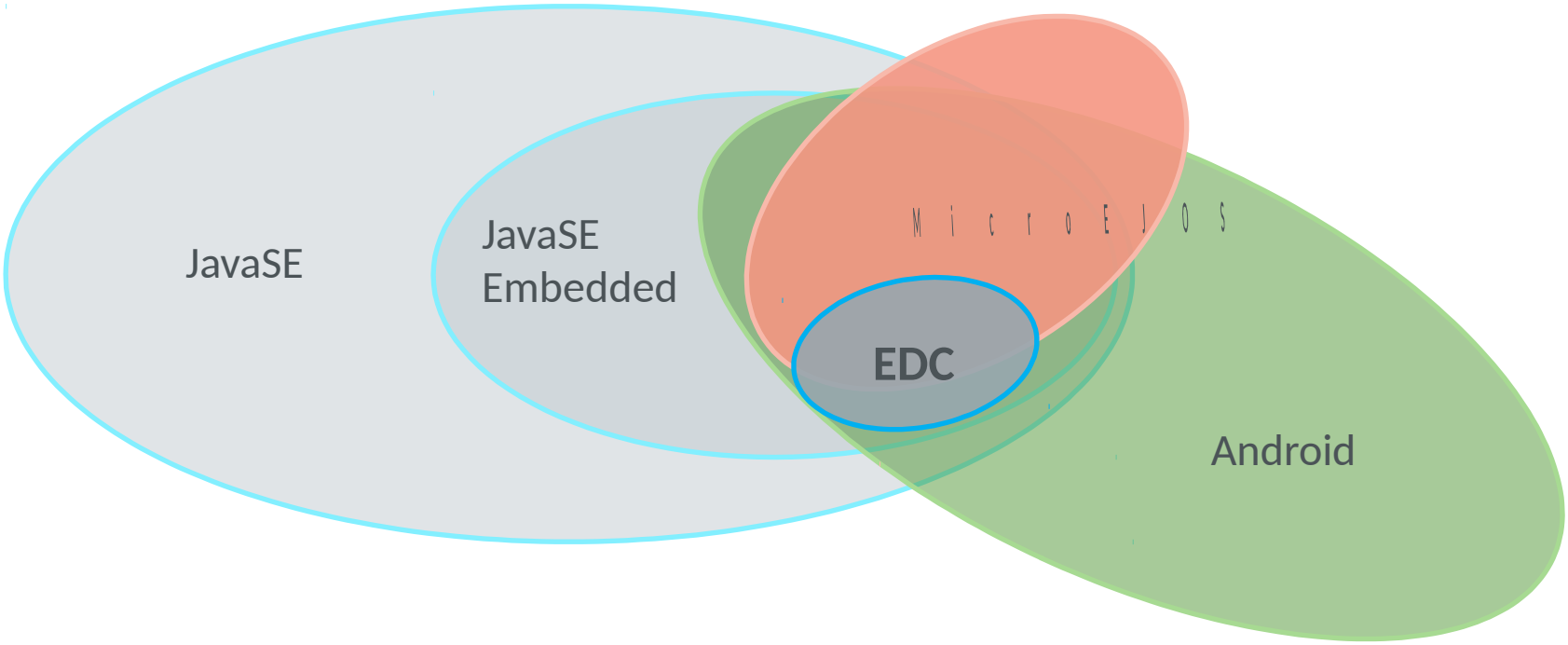


EDJE IS ALSO ABOUT DEFINING A COMMON API

List of minimum Java API

The minimum execution environment provided by an Edje-compatible device

Intersection between Java SE, Java SE Embedded, MicroEJ and Android

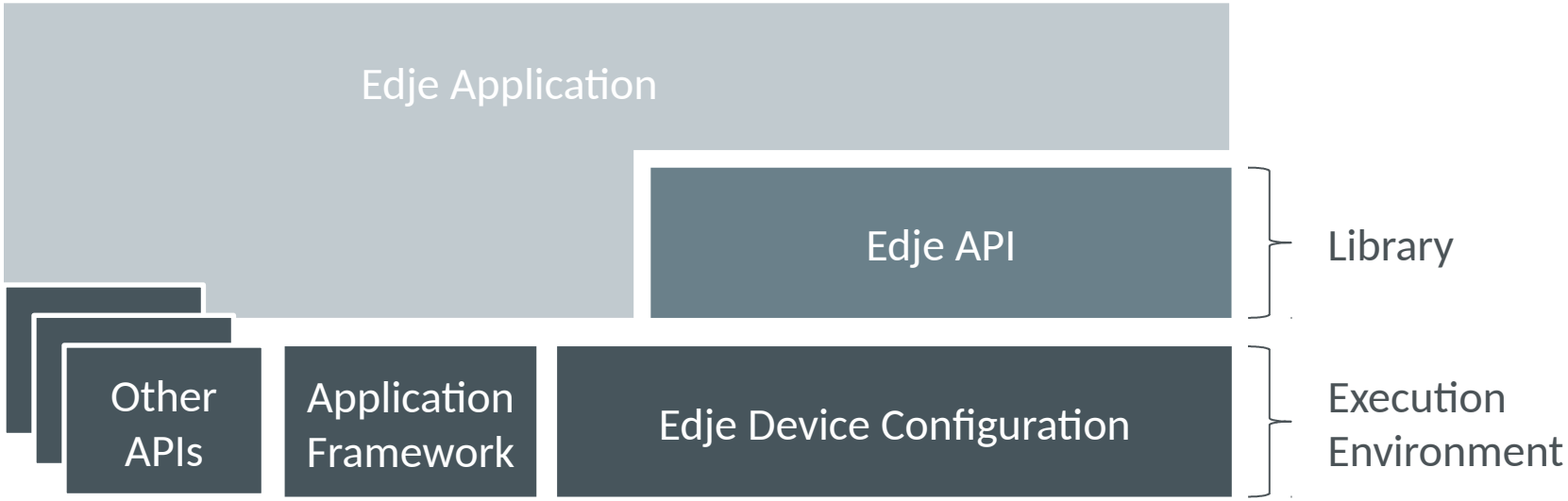


EDJE DEVICE CONFIGURATION (EDC)

List of minimum Java API

The minimum execution environment provided by an Edje-compatible device

Intersection between Java SE, Java SE Embedded, MicroEJ and Android



EDJE DEVICE CONFIGURATION (EDC)

How do you define such an API set:

- By looking at what customer applications need

- By trying to run useful open source libraries / services
  (Eclipse Paho, Eclipse Leshan, Eclipse Californium, cf the next presentation)

- By looking at other APIs on the market
  - Android (Things)
  - JDK Device I/O (Kura)

- It's an ongoing effort
  MicroEJ 4.1 is due next month, Embedded World is next week, EclipseConverge is next…. expect some news…

ABOUT MICROEJ

Independent Software Vendor, global player in the embedded IoT industry

- http://www.microej.com/about
- Software tool & runtime licenses
- Professional services, training & consulting

KEY FINANCIALS

- $5M Series B funding end 2011 – $20M R&D total investment

Offices in France, Germany, USA

Expertise in embedded, virtualization, software engineering & process

Partnerships with key IoT, silicon, embedded SW and HW (EMS) vendors

PARTNERS & CONSORTIA – STRONG BUSINESS PARTNERS

HARDWARE

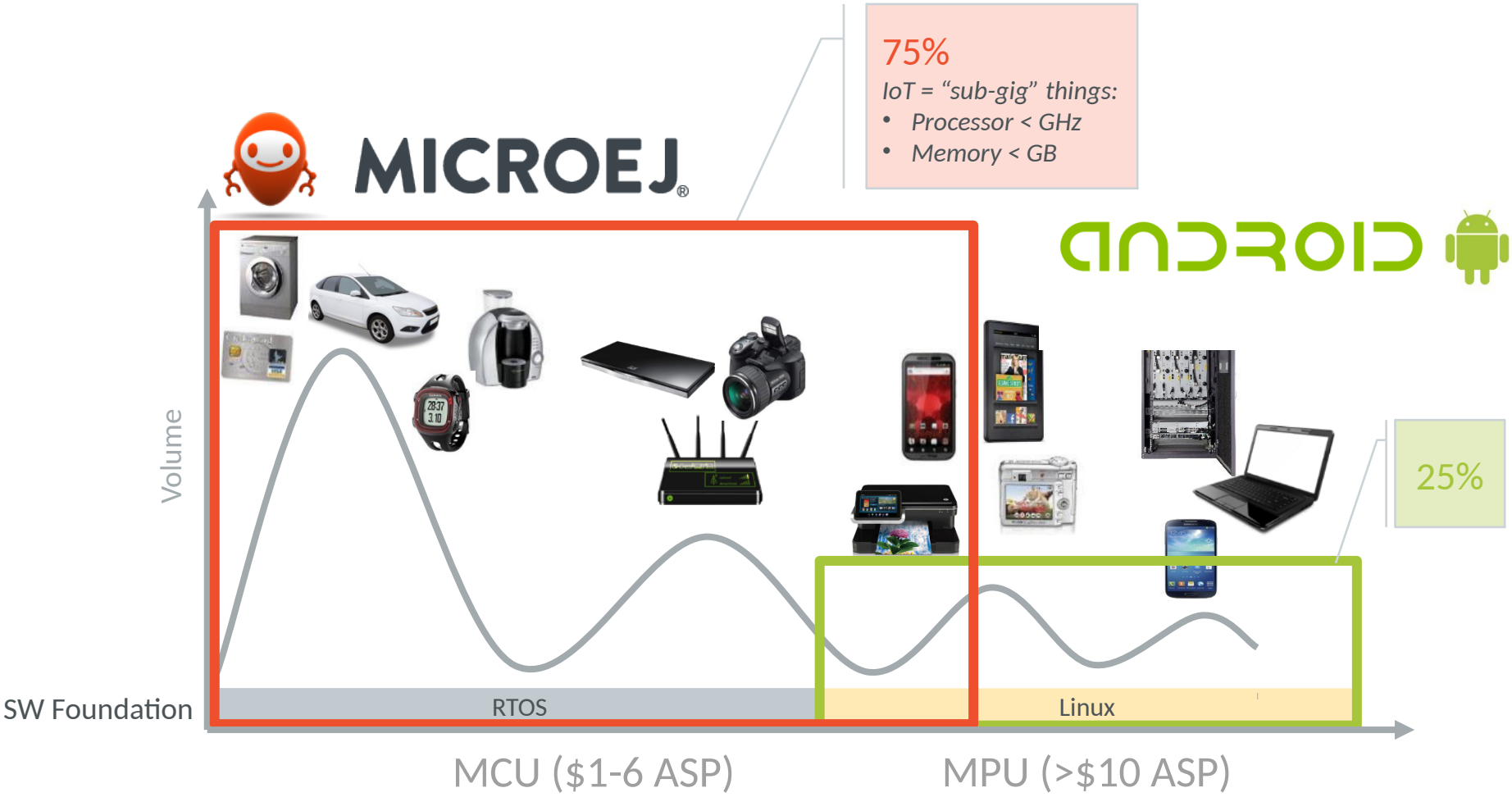
SOFTWARE

CLOUD

SERVICES

ALLIANCES

OPERATING SYSTEMS FOR ENABLING THE IOT

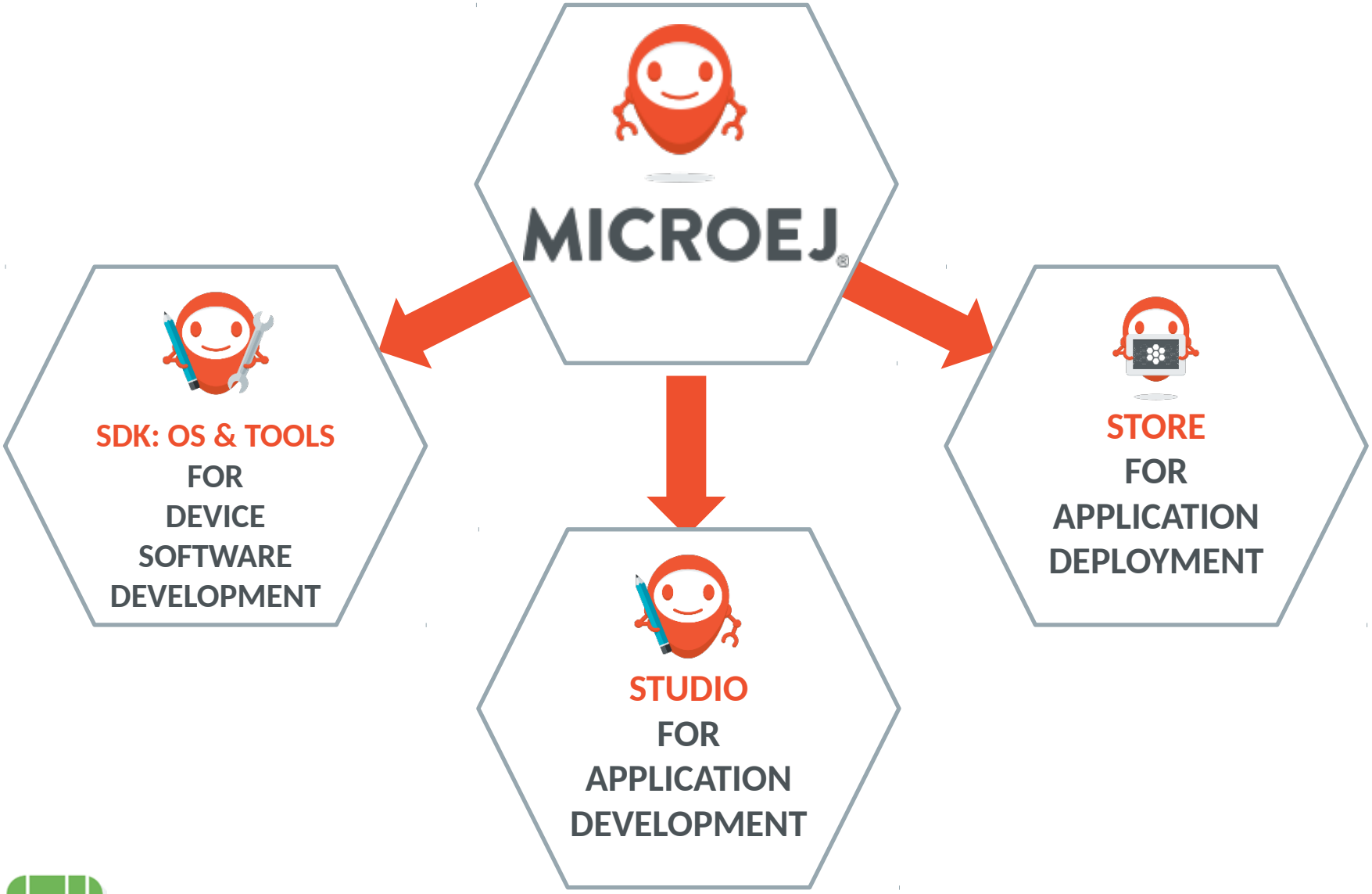


AN EXAMPLE OF FOOTPRINT REDUCTION

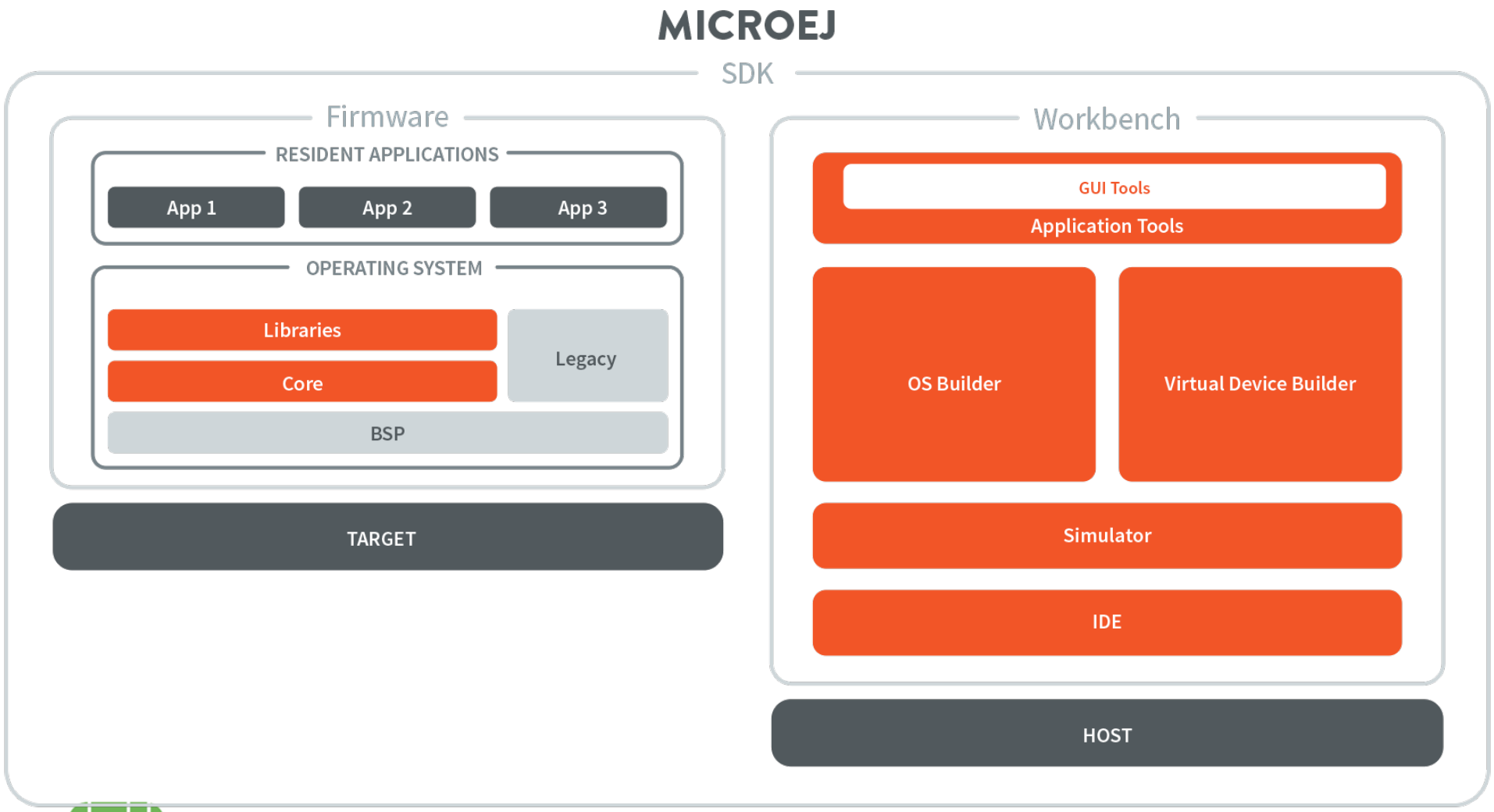
63% Bill of Materials savings
87% power consumption savings

	RTOS (Linux) + Java platform	RTOS (Any/None) + Java platform
Power	410 mAh – batteries recharge every day	Batteries recharge every two or three weeks
Processor	Qualcomm Snapdragon 400 MSM8226, ARM based Quadcore 1.2 GHz	ARM Cortex-M4 based MCU, 100 MHz
RAM (data)	256 MB	Less than 0.5 MB
FLASH (Code + Resources)	2,000 MB	2 MB
Boot Time	35,000 ms	50 ms

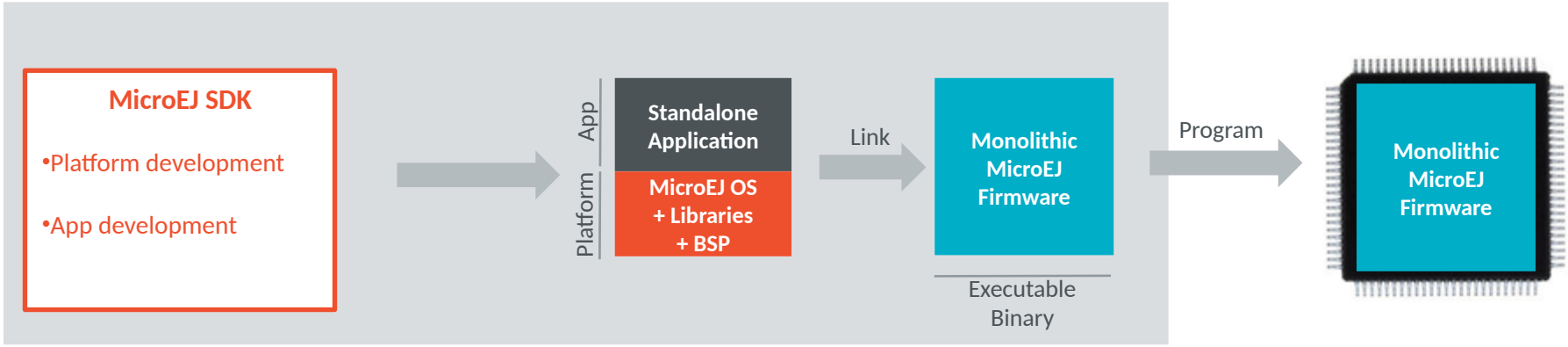
MICROEJ FLAGSHIP PRODUCT LINE



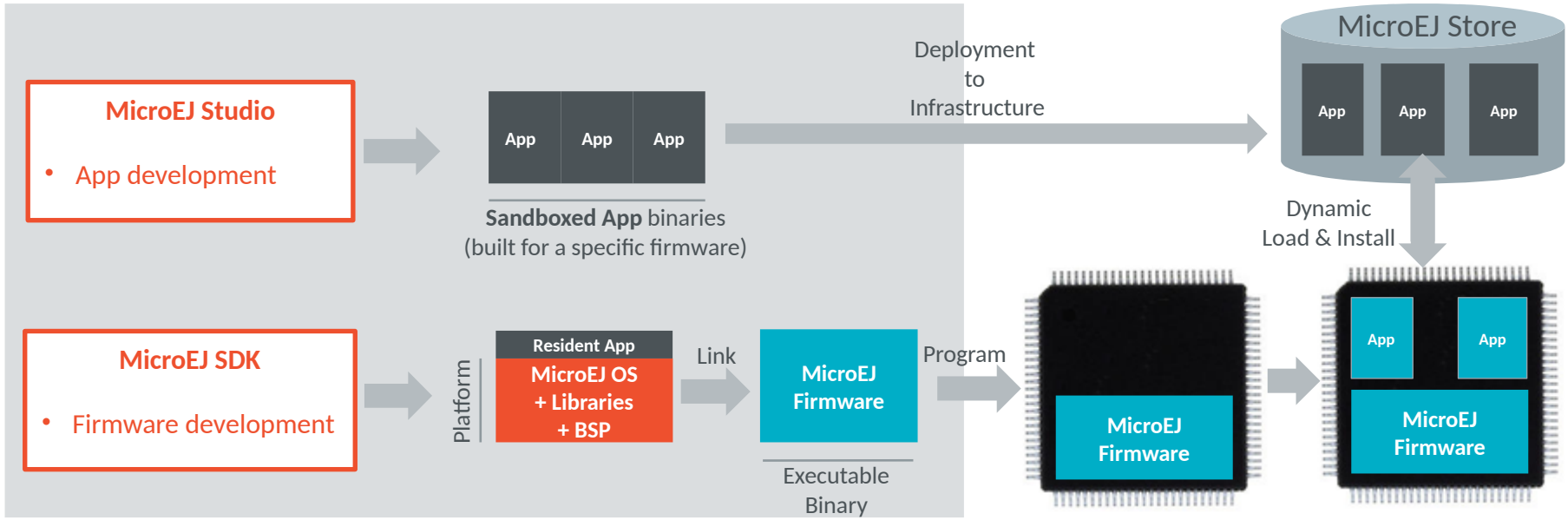
MICROEJ SDK COMPONENTS



SINGLE-APP DEVELOPMENT WORKFLOW



MULTI-APP DEVELOPMENT WORKFLOW



WHAT TYPES OF PRODUCTS CAN YOU BUILD ?

Sowee was announced by EDF at CES 2017

The offer includes a remote controller that runs a mix of MicroEJ and Eclipse IoT Java Technologies:

paho

LESHAN

MICROEJ®

LET'S SEE THIS IN ACTION : LIVE DEMO

MicroEJ & Eclipse IoT Java Technologies:

- **running on Renesas PEACH (Cortex A9, 10MB RAM, Arduino connector, Ethernet)**
- **a LWM2M client connected to leshan.microej.com**
- **a MQTT client connected mqtt.microej.com**
- **an example app playing with Leds and publishes on a topic**
- **a heartbeat service that publishes uptime on a topic**

THANK YOU

FOR YOUR ATTENTION!

https://projects.eclipse.org/projects/iot.edje
https://developer.microej.com/

HOW DO YOU TEST OUT EDJE ?

- Get a MicroEJ Compatible board
  e.g. STM32F746G-DISCO board from ST Microelectronics (~50€)
  (more boards are listed on developer.microej.com)

- Get a free copy of MicroEJ Studio at https://developer.microej.com
  follow the instructions in https://developer.microej.com/getting-started.html

- Clone the developer branch at https://github.com/MicroEJ/edje
  (it will be soon in https://github.com/eclipse/edje)

- Build Edje libraries from MicroEJ Studio
  (right-click « build with EasyAnt » on the two libraries projects you'll get from github)

- Run the sample applications from MicroEJ Studio
  (right click « Run As MicroEJ Application » and the app will be remotely installed and started on the board)

NEED MORE THAN JUST EDJE ?

- Get a free copy of MicroEJ Studio at https://developer.microej.com
  follow the instructions in https://developer.microej.com/getting-started.html

- MicroEJ Foundation Libraries:
  http://developer.microej.com/javadoc/microej_4.0/foundation/index.html

- MicroEJ Addon Libraries:
  http://developer.microej.com/javadoc/microej_4.0/foundation/index.html

- MicroEJ GitHub: (for examples, libraries, tools)
  https://github.com/MicroEJ