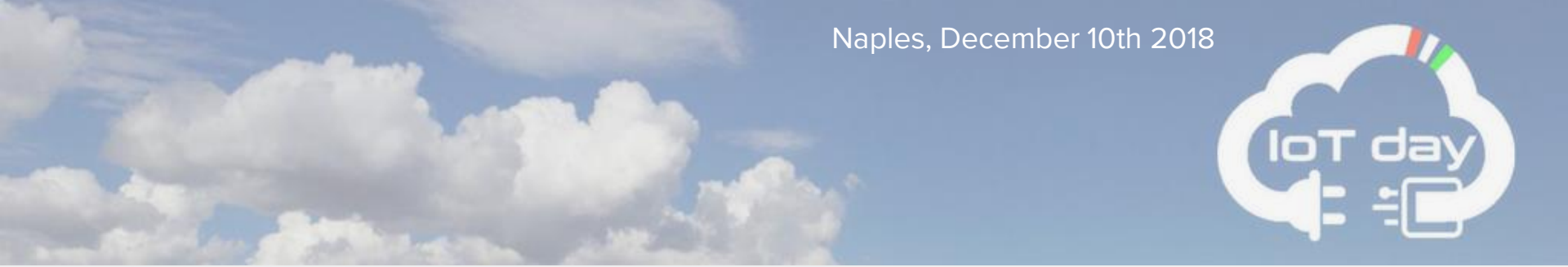
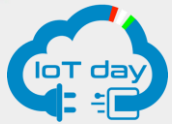


Loading a container on a small device is a good idea or a crazy one?

Valter Minute
@VMinute

redhat

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II



Who am I

- Valter Minute
- @VMinute on twitter
- Software developer, hacker, maker
- I work for Toradex AG (www.Toradex.com)
  - BSP/drivers developer
  - Currently working on container tools
- Microsoft MVP since 2009
- Working on devices since 1998

Arm System on Modules
Reliable
Long-term Maintenance
Scalable
From Stock

Production-ready Software
Yocto-based Linux
Windows Embedded Compact
Development Tools
Long-term Maintenance

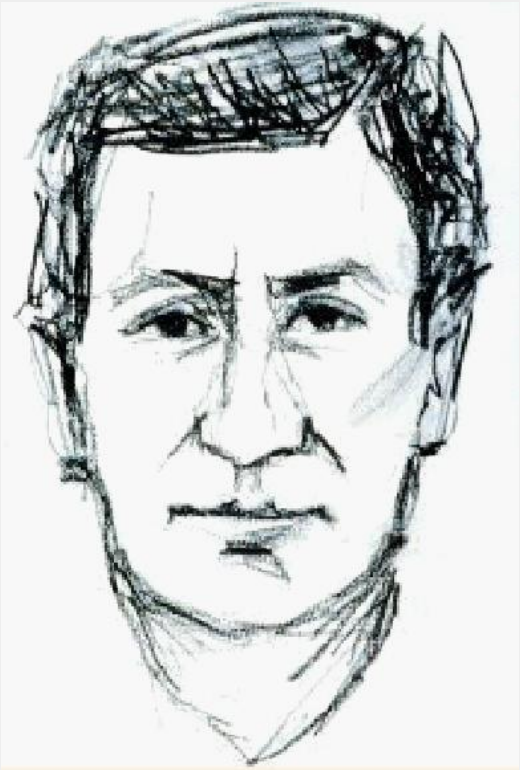
Toradex
Swiss. Embedded. Computing.

Ease-of-use
Support
Ecosystem

IoT day

Who are you?

- Software developers?
- Embedded developers?
- Makers?
  - Arduino
  - Raspberry Pi
- Linux users?
- Docker users?



Container

- An intermodal container is a 20 or 40 foot long metal box
- They became popular in late 40s early 50s, boomed in the 60s
- Fixed size, well defined specs
- All containers look the same, they are boring
- Containers can be easily loaded on ships, trains, trucks
- Containers defined a standardized way to ship goods

How containers changed the world

- Before containers ports were a huge mess of different ships, cranes, warehouses etc. one for each kind of transported good
- With containers
  - Bigger ships that are less expensive to operate
  - Standardized cranes, forklifts, trucks
  - 60% of goods moved on the sea are inside a container
- 6K container ships, 20 million containers moving in this moment
- Containers changed the way we eat
  - Refrigerated containers allow transportation of bananas, pineapples and many other kinds of foods that are now very common in our supermarkets

Does one size fits it all?

- NO
- Some containers are not filled completely
- Some kind of products had to be changed
- Containers dictate the size and shape of many products
- Some kind of materials or goods are still not shipped inside containers (oil, grains, cars)

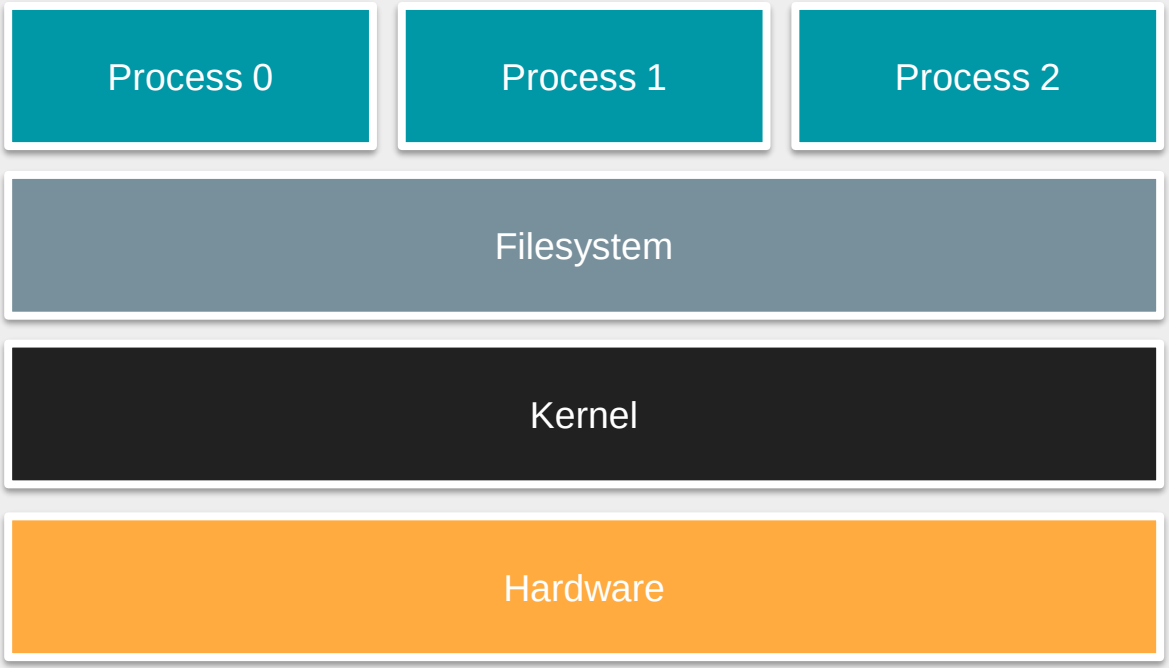
Containers in Software

- Lightweight Virtual Machine
  - User-mode virtualization
  - Isolation
  - Portability
  - Native drivers
- Easy way to package software applications
  - All user-mode dependencies in the same package
  - Fully dedicated environment
  - Easy to clone and distribute

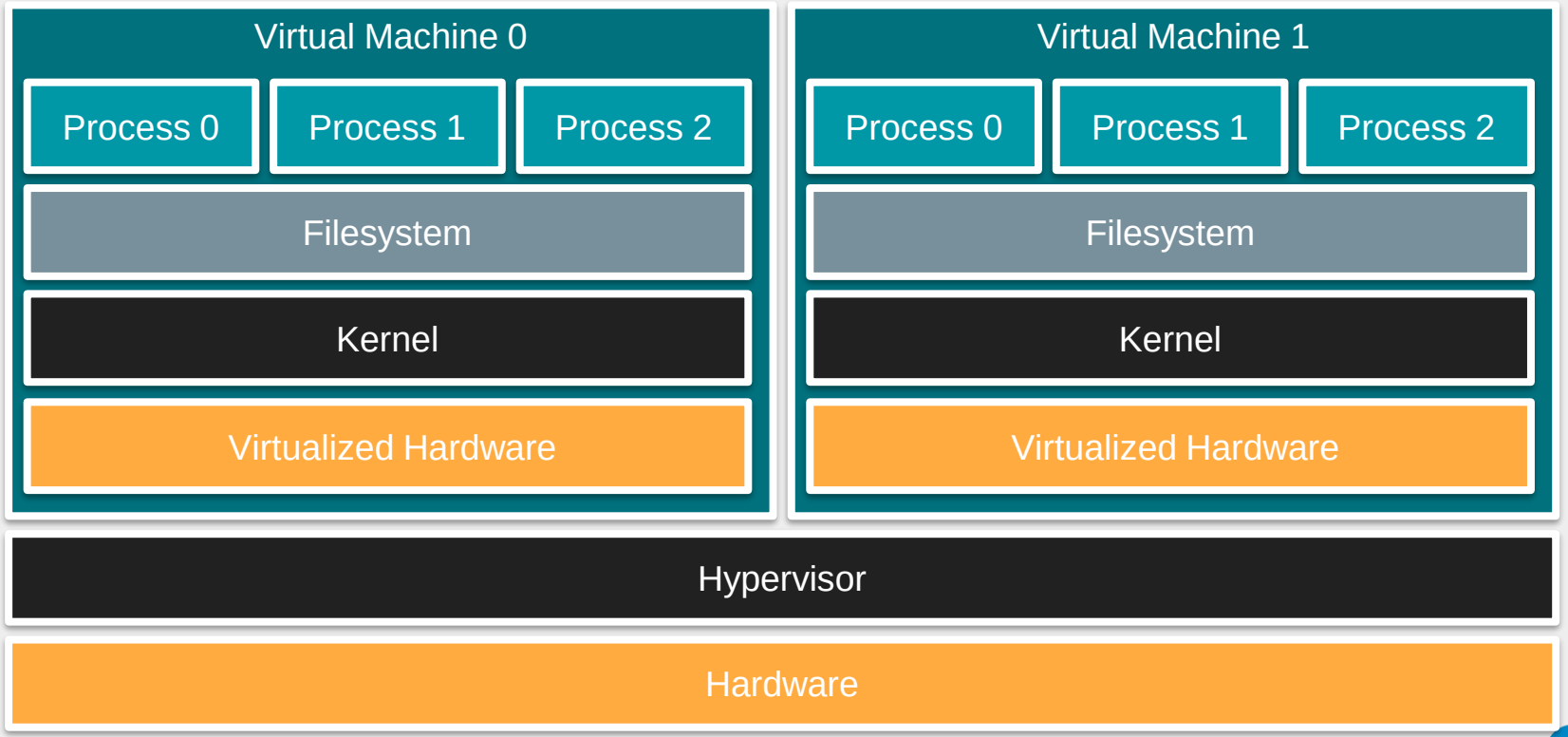
Containers explained in less than 5’

(but don’t pretend an exhaustive explanation)

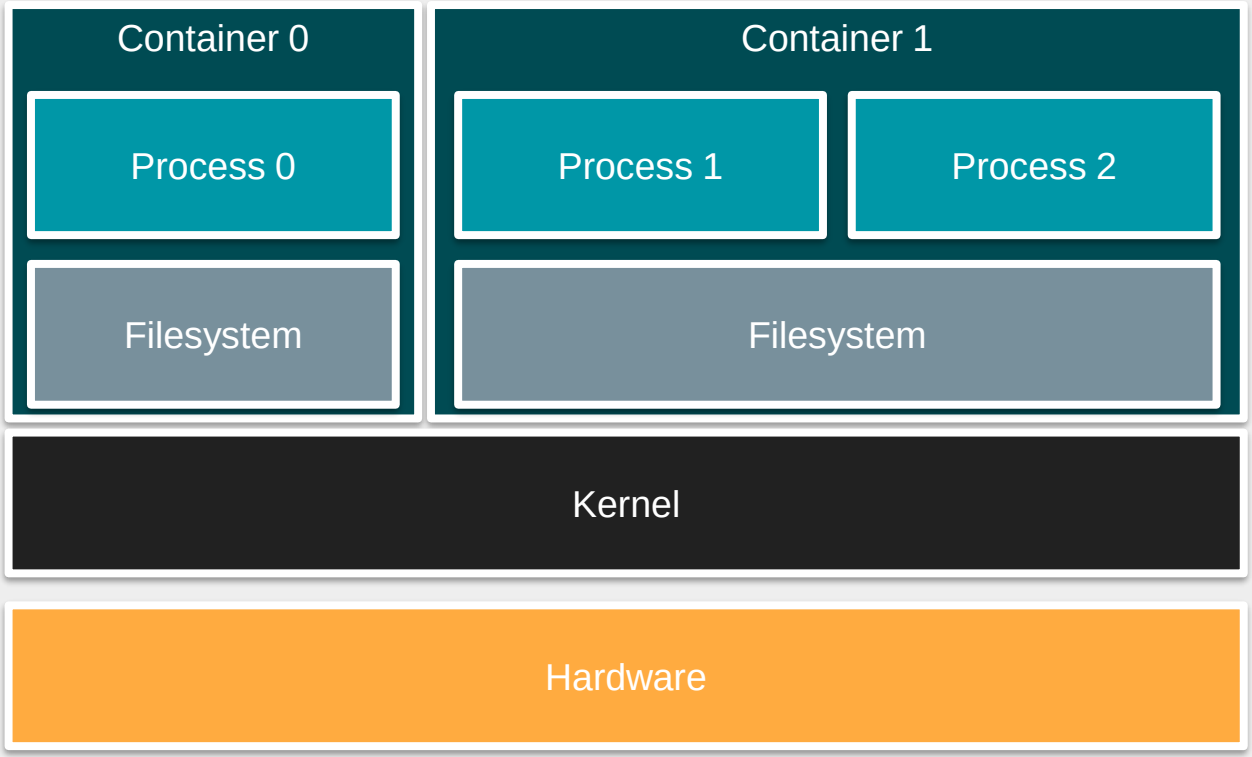
Linux (and most of modern OSs)



Hardware Virtualization

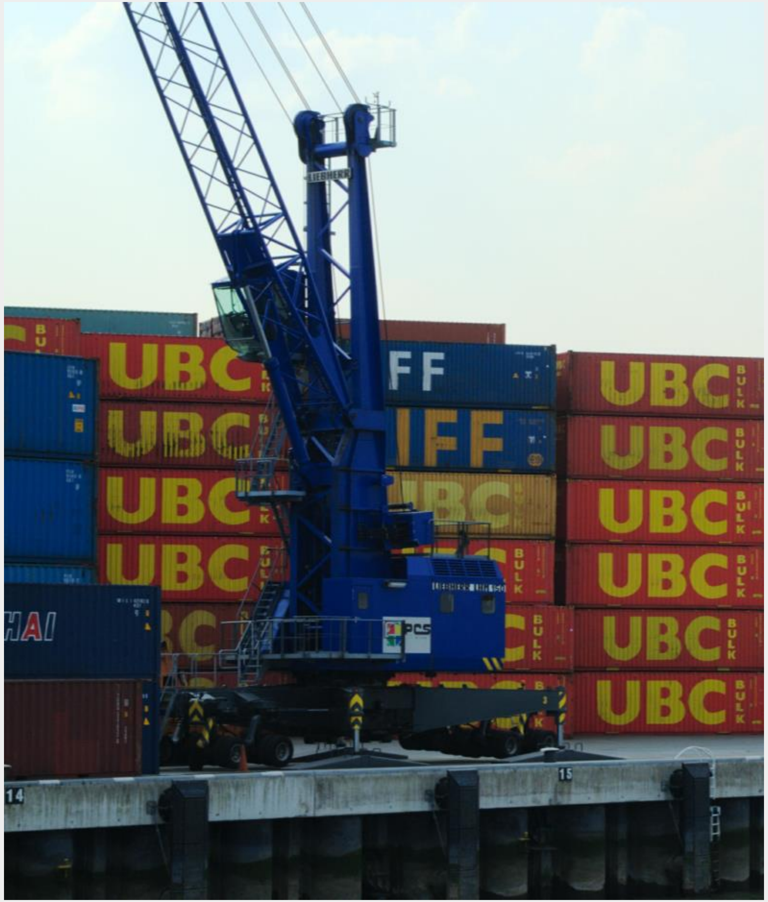


Containers


Container 0
Process 0
Filesystem
Container 1
Process 1
Process 2
Filesystem
Kernel
Hardware

Container advantages

- An application and his dependencies can be “packed” inside a single shipment unit
- Container runtime can easily deploy and run containers on top of an existing operating system
- Runtime and tools can operate on containers using different distros
- Containers can use network but can also be restricted

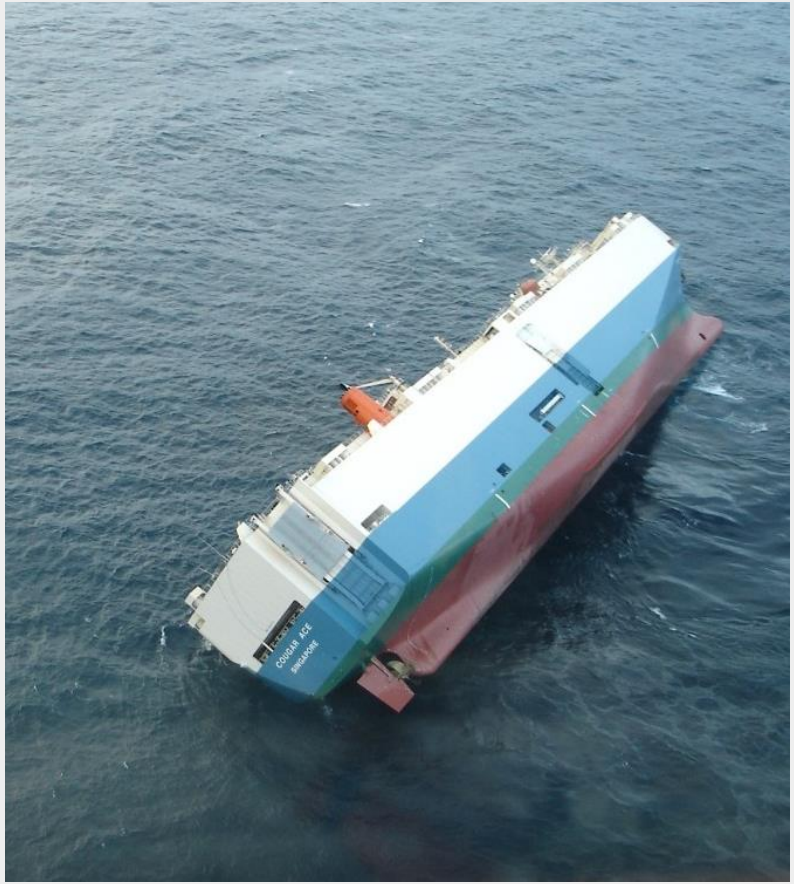


Container advantages (pII)

- Resources used by a container can be limited
- Containers don’t have the extra overhead required for hardware virtualization
- Layers can be shared, making deployment much faster
- Runtime instances could be easily created and destroyed

Container drawbacks

- Extra storage is needed
  - Each container has a different user mode filesystem, no shared libraries between containers
- Extra memory is needed
  - No shared libraries
- Additional layers of virtualization
  - Limited access to hardware
  - “Sandboxing”
- Less isolation
  - Containers may be less secure than VMs



Containers on embedded devices

- You will waste a lot of resources!
- You don’t need high level languages and frameworks on a device (see point 1)!
- Containers are designed for servers!
- Process isolation is enough!
- This will add complexity to my development cycle!
- If you can’t access hardware directly you will lose performances!
- Things invented for web development are for hipsters!

What about...Linux?

- Born as a multi-user server OS, now widely used on billions of devices (including your phone, disguised as a nice green robot)
- Process isolation by default
- HW is accessible only in kernel mode (more or less by default)
- Multi-user by design
- Devices uses lots of high-level frameworks and tools originally developed for PCs and servers.

Linux on embedded devices

- Today Linux can be used to build devices that
  - Get to market quickly (common driver model, support for many architectures and SOCs)
  - Are secure (kernel vulnerabilities are usually patched quickly after disclosure, user-mode layers are widely used, tested and hacked)
  - Are future-proof (Linux is here to stay, applications can be migrated to new kernels, new hardware is probably supported on Linux first)

So why we need containers?

- Wide range of tools requiring different libraries, versions etc.
  - DLL hell, just with a different extension
- Isolation of different components of the same solution
- More portable applications
  - AI, machine learning
- Independently maintained micro-services

Will Containers replace Linux applications?

- Linux has been widely adopted in embedded, mostly in complex devices
- One size does not fit all, Linux did not replace firmware/RTOSs on all devices, containers will not replace Linux applications on all devices.
- Devices are getting more complex, more connected, more critical for the infrastructure of a company or an organization
- But this is an option you should evaluate if you want to build a modern solution reducing development and maintenance costs

Change your perspective

- Devices are more and more often part of a complex solution
- Containers can be “building blocks” that will help you developing your software
  - Use the right tools for the different part of the system
  - Develop and deploy components with different release cycles
  - Extend your solution by adding new modules
- Base OS provides the services that your containers need and a secure way to manage them

Containerized OSs and solutions

- Fedora IoT
- ARM MBED/Linux
- Azure IoT Edge (modules are containers)
- Linux Microplatform
- Torizon (labs.toradex.com)

Thanks !

Questions ?