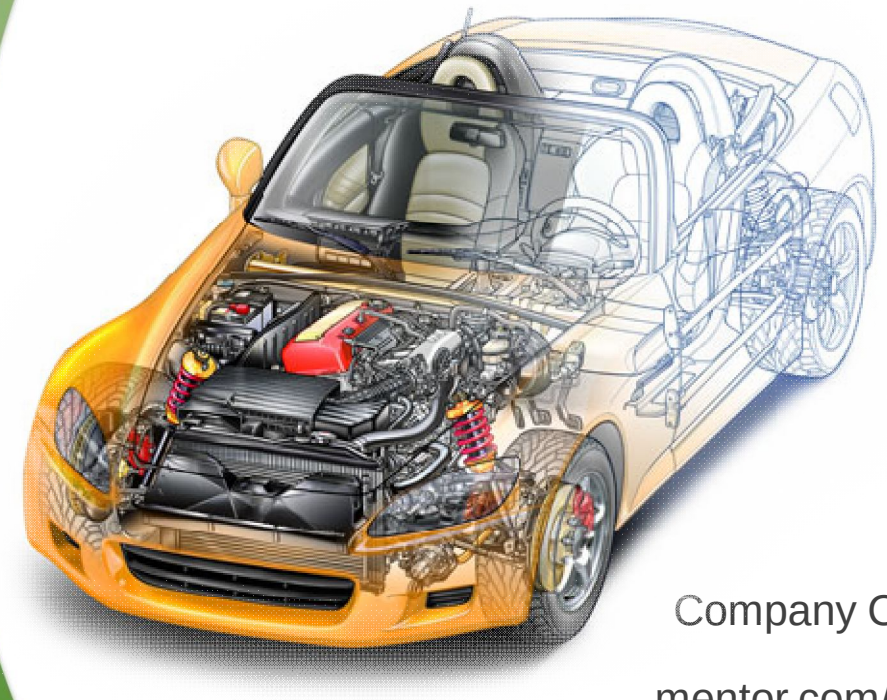
mentor embedded

Open Source
making its way within automotive



Company Confidential

mentor.com/embedded

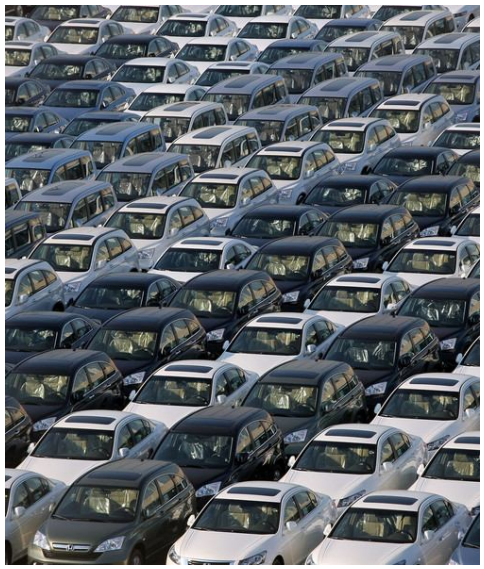
Mentor Graphics®

Disclaimer

This content is based on personal experience and/or research. Slides were reviewed to only contain publicly-available information.

mentor embedded

Launched Linux based Systems



Company Confidential

mentor.com/embedded

Mentor Graphics®

First Linux Instrument Cluster

Innovators get Linux to boot in 1 second

DOING REAL-TIME LINUX IS HARD ENOUGH. MAKING IT BOOT IN 1 SECOND TOOK REAL INNOVATION.

MontaVista Software has always been a leader in embedded-Linux commercialization. The company has developed Linux-development platforms since 1999, when founder Jim Ready pledged to bring "100% pure Linux" to the world under the GNU (GNU's not Unix) GPL (general public license). Since then, MontaVista has specialized in embedded and real-time Linux.

Its approach is not simply an RTOS (real-time operating system) that runs Linux as one of its tasks. The company has changed the Linux kernel to provide determinism and real-time performance in a real Linux operating system. Cavium Networks recently acquired the company, which just announced the release of Version 6 of its operating system.

In addition to designing real-time Linux, MontaVista has been working on the development of real-fast Linux, a Linux operating system that boots in less than 1 second. The team who worked on the project includes Alexander Kaliadin, Nikita Youshchenko, and Cedric Hombourger. Many on the team also worked on the MontaVista real-time Linux. "One of the first things we did years ago was to make the Linux scheduler pre-emptive and deterministic," says Hombourger. These fast-boot developments are not necessarily limited to real-time or an embedded Linux; however, they can get a conventional Linux distribution to boot in 1 second, as well.

"The methods we have developed are independent of whether [you use] a real-time kernel," notes Kaliadin. He says that the team first considered netbooks and mobile Linux fast-boot approaches. It could not adapt them, however, because they needed a fairly heavyweight Linux distribution, and the requirements for a 1-second boot are far more extreme than those of any netbook or notebook application. "We had to devise a set of new techniques in both the kernel and the boot loader," says Kaliadin.

A customer who needed his automotive-dashboard system to come up quickly prompted the development of a fast-booting Linux. Kaliadin remarks that the customer loved the MontaVista OS but had a fixed requirement to show data on the screen in less than a second. "Our first reaction was that it was impossible," says Kaliadin.

The 1-second-boot-time achievement came in three stages. The first stage was optimization in all the obvious places, including the boot loader. The team could eliminate some of

CONTINUED ON PAGE 53

The MontaVista team includes Alexander Kaliadin (left) and Cedric Hombourger (right).

Platform:

- Application processor: Freescale MPC 5121
- Kernel: Linux 2.6.24
- Comm with automotive processor via RS232 link

Noteworthy:

- 1st known use of Linux in a production car
- At your dealers since 2011
- 1s boot time!
- Minimalistic Linux system
- OpenGL-ES based HMI

OSV contributions:

- Port u-boot and Linux to custom board
- Early gear display < 1s
- Full cluster application < 3s
- Field upgrade via USB sticks
- End Of Line Calibration

First Linux Infotainment System

Platform:

- Processor: Tri-core ARMv6
- Kernels: Linux 2.6.34 + T-Engine (no hypervisor)
- Electrical MOST50, LIN and CAN buses

Noteworthy:

- 1st known Linux-based infotainment system
- At your dealers since fall 2012 (NA market)
- 8s boot time!
- OpenGL-ES based HMI

OSV contributions:

- Port u-boot and Linux to custom board
- Fast boot
- MediaPlayer (USB, iPod, Bluetooth AVRCP)
- Pandora radio
- Telephony (BT 3.0, Phone, Messaging, PhoneBook)

First Mass Production Linux Infotainment System

Platform:

- Processor: Tri-core ARMv6
- Kernel: Linux 2.6.34 + T-Engine (no hypervisor)
- Electrical MOST50, LIN and CAN buses

Noteworthy:

- Same platform for NA, Europe and China car lines
- In market soon!

OSV contributions:

- Enhance MY13 platform
- Add HTML5-based application framework (WebKit based, JavaScript vehicle/applications APIs)

First Mass Production Linux Infotainment System

Platform:

- Processor: Texas Instruments ARM Cortex A8
- Android 2.2

Noteworthy:

- TODO

OSV contributions:

- TODO

Current Trends

Company Confidential

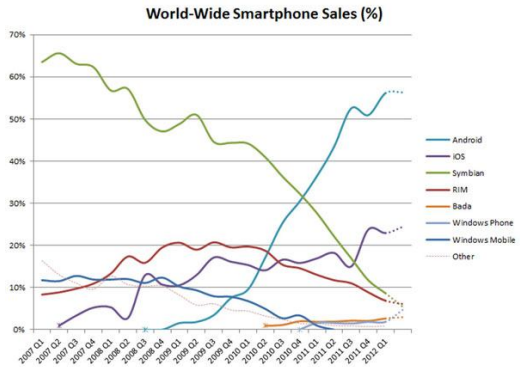
mentor.com/embedded

GENIVI-based systems

GENIVI: consortium found in 2009 to establish a Linux-based Operating System, middleware and platform for the automotive in-vehicle infotainment industry

Android-based systems

Incredible adoption in the mobile space, automotive to follow?



While some already produced cars with Android in the head units, others are still figuring out how to deal with it…

Possible reasons:

⇒ Android isn't Linux and it isn't GENIVI / AGL either…

⇒ Nice but driven/owned by Google…

⇒ How can I meet my early Audio requirements?

⇒ Control of the system to meet safety requirements

Virtualization / Containers

Need for multiple OS domains:

- Android, GENIVI, AUTOSAR, …
- Increased connectivity and security needs
- Isolate infotainment apps from automotive components

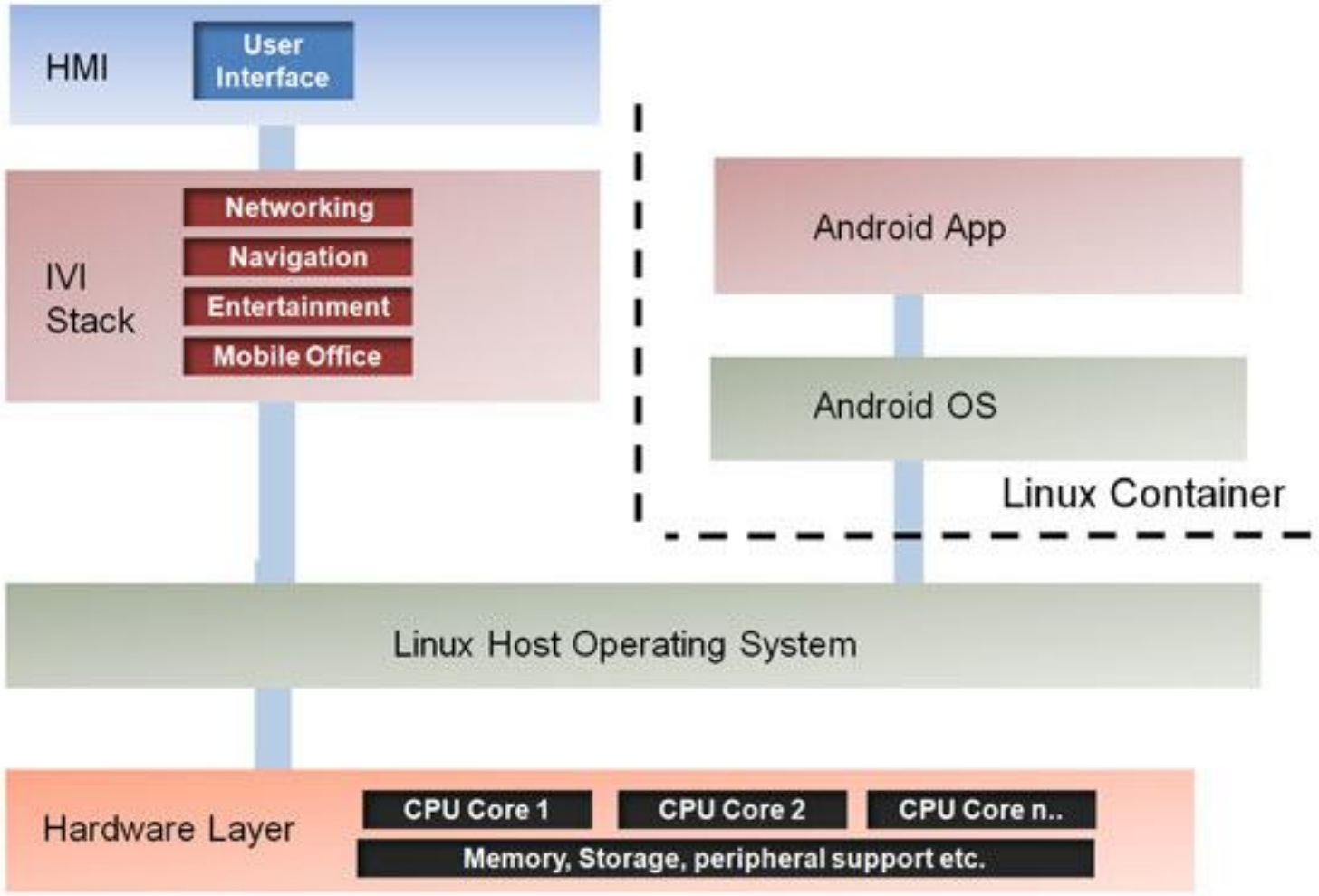
Powerful hardware platforms:

- i.MX6, Jacinto, R-Car

Solutions:

- Linux Containers (LXC)
- 3rd Party Hypervisors (e.g., OKL4, Integrity)

Stitching it all together



Instrument Cluster / Infotainment consolidation

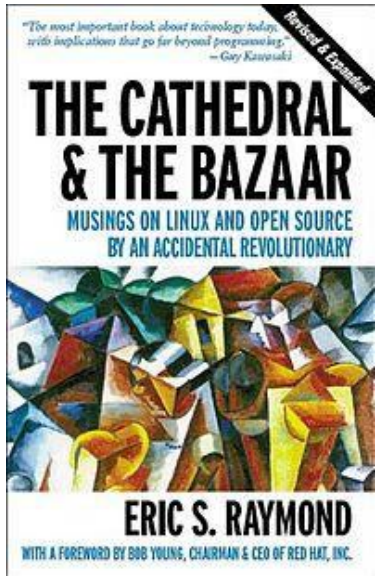
mentor embedded

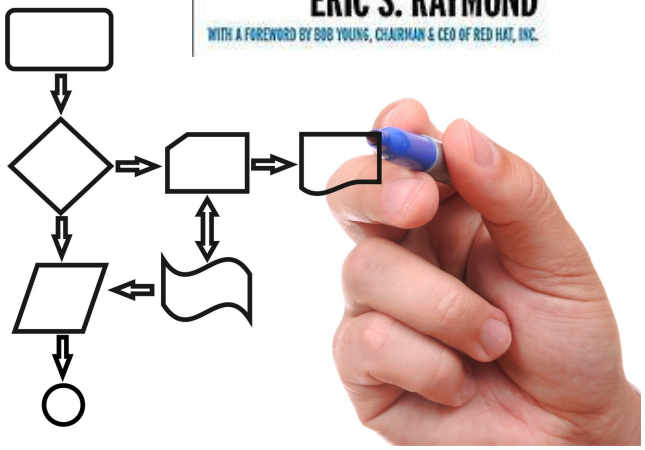
Company Confidential

mentor.com/embedded

Mentor Graphics®

Process-driven versus the Bazaar





Industry typically requires CMMI or SPICE processes:

⇒ Linux developers/architect need to be trained

Community projects remain independent:

⇒ Have their own roadmaps (possibly conflicting with yours)

⇒ May not be interested in fixing issues you've found

⇒ May change license (e.g., LGPLv2 => LGPLv3)

Different approaches:

⇒ Complete traceability from requirements to tests versus code-centric

⇒ Built for several years versus what's-the-next-big-thing-we-can-add

Automotive-specific challenges

- Match consumer devices features with automotive constraints
  - Heads Units are not phones/tablets, driver should not be distracted…
  - Vehicle Application Store framework/strategy, how to attract app developers?
- End Of Line Calibration
- Low/High Voltage handling
- Early audio/video
- Dealing with proprietary technologies, patents and certifications
  - Apple (IAP, Siri), Audible (Audio books), Microsoft (ASF, MTP), Pandora, …
  - Made for iPod, Bluetooth, Wi-Fi, MOST, …
  - AAC, Apple Lossless, MP3, …
- Bluetooth
  - BlueZ not up-to-par, 3rd party stacks typically used
  - Many phones behave differently despite being compliant
- Critical Hardware Resources (memory, CPU, flash…)
- Updates:
  - Partial vs System wide
  - Dealer vs Customer upgrades

And a blocked production line can cost millions…

Fragmentation

Standards:

- GENIVI, AGL, Android, …

- Technologies

License soup

- ⇒ Open source projects use a variety of licenses
- ⇒ Projects claim being under certain licenses but may include code with other license(s)
- ⇒ Use the right tool(s) to identify code you are including in your product(s)
- ⇒ Understand how OSS components are glued together to avoid license contaminations / incompatible mixes
- ⇒ Have an Open Source Software board combining legal and technical experts:
  - ⇒ Whitelist/blacklist projects
  - ⇒ Review obligations

mentor embedded

Keys to success

Company Confidential

mentor.com/embedded

Mentor Graphics®

Use Code Analysis Tools

1. Select the tools you need:
   - LINT, Coverity, Klocwork, gcov, …
   - Fossology, Blackduck Protex, …
2. Define reasonable targets:
   - Components to scan
   - What to look for / fix (e.g., define warning priorities)
3. Incorporate the selected tools into your development processes
4. Define an upstreaming strategy

Have the right Subject Matter Experts

You'll need seasoned engineers on key components such as:

- D-Bus
- GStreamer
- Linux kernel
- openSSL or other crypto frameworks
- Qt or other application frameworks
- u-boot
- Wayland
- WebKit
- …

They should:

- Understand the component architecture
- Know design/implementation limitations / caveats
- Have experience working with the component community
- Have insights on the component roadmap