Open Source in Industrial Automation – already a Reality

Carsten Emde
Open Source Automation Development Lab
(OSADL) eG
Linus Torvalds, October 1991:
"[...] I'm working on a free version of a Minix-look-alike for AT-386 computers. [...] I am willing to put out the sources for wider distribution. It is just version 0.02 [...]"
Let's have a closer look at embedded systems
Linux on its way to a universal operating system for embedded systems

Hurdle #1: Is the Open Source license (here: GPLv2) suitable?

Hurdle #2: Can Linux Real Time?

Hurdle #3: Can Open Source software be certified for safety, e.g. SIL2?
Hurdle #1: Is the Open Source license (here: GPLv2) suitable? Why would this be a problem?

- GPLv2 (GNU General Public License) is a license with strong “copyleft” effect. ("Copyleft" denotes a type of Open Source license that requests the use of the original license for additions or modifications, although they could be licensed at the author's choice, otherwise.)
- Additions or modifications of a Linux driver, for example, must be licensed under the GPLv2 as well – thus possibly disclosing trade secrets.
- In addition, it is not possible to implement patented procedures such as Profinet or Ethercat real-time Ethernet in the Linux kernel.
- Nevertheless, information, disclosure and licensing obligations must be followed whenever the software is conveyed.
Hurdle #1: Implications of the GPL

Linux Developer (creates)
Licensor

Chip manufacturer (creates)
Licensor
Licensee

Board manufacturer (creates)
Licensor
Licensee

Machine builder (creates)
Licensor
Licensee

End user (uses)
Licensee
Hurdle #1: Implications of the GPL

- **Linux Developer** (creates)
  - Licensor
  - Licensee

- **Chip manufacturer** (creates)
  - Licensor
  - Licensee

- **Board manufacturer** (creates)
  - Licensor
  - Licensee

- **Machine builder** (creates)
  - Licensor
  - Licensee

- **End user** (uses)
  - Licensee

Enablement
Hurdle #1: Is the Open Source license (here: GPLv2) suitable? Yes or no?

• It is not a hurdle when the decision making to introduce Open Source software is made by the software developer instead of the management.

• It is estimated that in the majority of cases there is no strategic decision of the management to introduce Open Source software – but it certainly should.

• The recent increase of GPL troll activity is creating a certain awareness of the GPL license obligations, but it is too late to give up on Open Source software.

• So the Open Source license normally is not a hurdle, but is able to create some frustration later on.
Hurdle #1: Is the Open Source license (here: GPLv2) suitable? Proposal.

The pyramid of differentiation

"Uniqueness line"

"Differentiating" know-how

"Non differentiating" know-how

Yes

No
Hurdle #1: Is the Open Source license (here: GPLv2) suitable? Proposal.

The pyramid of differentiation

- Operating system
- Operating system drivers
- Operating system libraries
- Application libraries
- Application

"Uniqueness line"

"Differentiating" know-how

"Non differentiating" know-how

Open Source

No Open Source
Hurdle #2: Can Linux Real Time? In 2000?

No!
Hurdle #2: Can Linux Real Time? In 2016?

- However, an external (PREEMPT_RT) patch is required to enable real-time.
- But maintenance, development and upstream consolidation of the patches is a well-funded official Linux Foundation project now.
Hurdle #2: Can Linux Real Time? Unpatched kernel
Hurdle #2: Can Linux Real Time? Patched kernel
Hurdle #2: Can Linux Real Time? Patched kernel

No outlier in more than 180 billion preemption measurements during more than two years
Hurdle #3: Can Open Source software be certified for safety, e.g. SIL2?

- Conforming to the standard (the “normal” $1_s$ route)
  **Open Source: No, this is impossible**

- Proven in use ($2_s$)
  **Linux: No, this is (nearly) impossible**

- “Conforming non-conforming development” ($3_s$)
  **Linux: Yes, this may/should be possible**
What is “Conforming non-conforming development”? 

• Careful analysis of the non-conforming development process („can't be that bad, if it succeeds in creating good software”)

• Thorough comparison of the non-conforming with the standardized development process:
  - Matching or overfulfilling areas → Document them
  - Non-matching or lacking areas → Complement/substitute them
How to complement non-matching or lacking areas?

• Static code analysis
• Additional safety layer (e.g. to catch I/O timeout)
• Long-term recording to assess reliability and stability
• Other yet unknown procedures that will be developed during the first project phase
Conclusion: Yes, there is an applicable certification route that can be taken

Thus:

• Find a certification authority
• Define a suitable procedure
• Recruit a core team
• Let grow a sufficiently large community
• Estimate the budget and raise funds
• Go!

“SIL2LinuxMP” was born!
What is SIL2LinuxMP?

- OSADL project with mixed funding
- Certification authority: TÜV Rheinland, Cologne, Germany
- Estimated schedule: 2015 to 2017
- Bootloader, root filesystem, GNU/Linux RTOS, C library
- Two subsequent project phases
  - Phase #1: Creation and definition of the processes
  - Phase #2: Certification based on these processes
Is there another hurdle?

Development of Open Source software is done in a community. But successful use of software requires more than the source – you need

- maintenance,
- quality assessment,
- training,
- installation,
- legal advice,
- certification
- and probably a lot more.

If all of this were done individually, the entire economic advantage of Open Source software would be lost!
Saving the best till last ...

No, this is not (and never was) a hurdle, since there is the Open Source Automation Development Lab (OSADL).

OSADL takes care of everything that is needed to successfully and painlessly deploy Open Source software in industrial products.
OSADL growth since 2005
Conclusion

There were some less well known and some well known hurdles to use Open Source software in industrial automation.

- Acceptance of Open Source software (with and without copyleft)

- Real-time as an example of a particular software capability not needed for mainstream applications

- Certification for use of Open Source software in a SIL2 safety-critical environment
Conclusion

There were some less well known and some well known hurdles to use Open Source software in industrial automation.

- Acceptance of Open Source software (with and without copyleft)
- Real-time as an example of a particular software capability not needed for mainstream applications.
- Certification for use of Open Source software in a SIL2 safety-critical environment.