Standards, the kernel and Open Source

http://veillard.com/Talks/CLKBeijing2012.pdf

Daniel Veillard
veillard@redhat.com

redhat

Standards: why ?

Main purpose is interoperability

- Public description of the technology
- Test suite or conformance checks

Main benefit is cheaper, ubiquitous technology

- Not tied to one vendor
- Larger user base
- Competition between vendors

redhat

Classic example: camera/phone cable

redhat

Standards: How ?

Various standard bodies, process is usually:

1) Multiple actors
2) Agreement to make a public specification
3) Shared work on creating that specification
4) Multiple initial versions (with feedback)
5) Vote and publication as a standard
6) Maintainance

Usually takes a few year, often painful

redhat

Standardization groups

And many others ...

redhat

What about the Linux kernel ?

- Linux is in the C language, which is standardized !
- Linux initial success was tied to the POSIX API
  - Implementing the standard gained a lot of applications
- We rely on standardized hardware
  - Buses (PCI/I2C...)
  - Protocols to talk to disks and other peripherals
  - Boot process
- We rely on standardized networking
  - From the lowest level: frame/packet level
  - Up to the application: Web, Video, etc ...

redhat

Looking more closely

Looking at the linux kernel code 3.6.1 source code

- IETF RFC standards:
  - Reference 212 different RFCs in the code base (544, 791, 792, 793, ... up to 5961, 6106, 6164, 6298)
- Many many references to IEEE
  - 802.11/802.15.4 for all the wireless
  - 1394/1212/1284 for firewire/SCSI/parallel
  - 754 floating point arithmetic
- ISO standards:
  - CD filesystems, character sets, networking
- ...

redhat

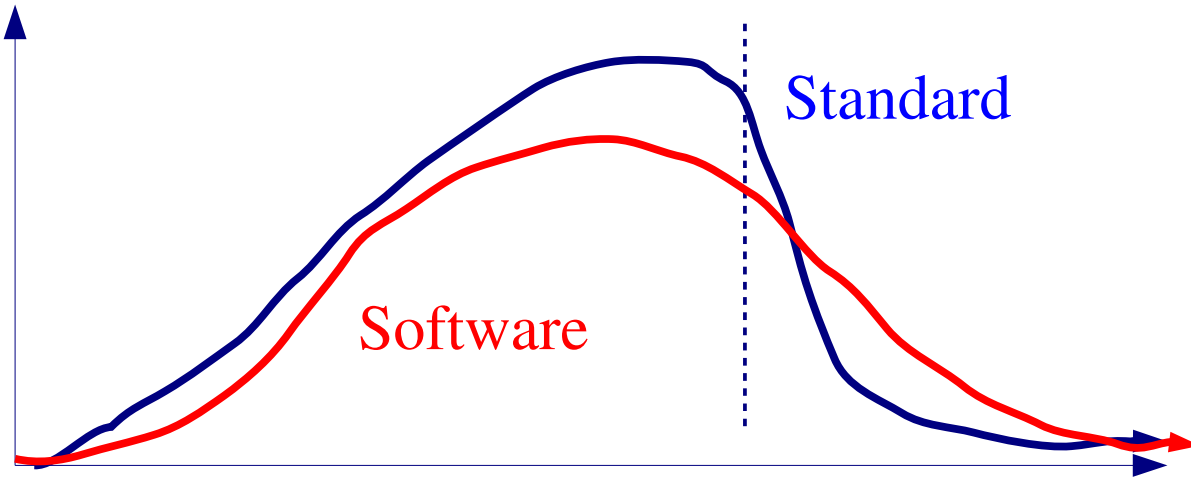
Why should I care ?

- Sometimes you won't have the choice:
  - Interoperability is crucial
  - New hardware is coming
- Sometimes you have the choice:
  - Which standard(s) to implement
  - Finding relevant standards can be challenging
  - You may not like it, make sure you don't exclude it by design
- Sometimes you want to be involved:
  - It may still be time to fix it !
  - They may need your implementation to finish
  - Providing test case and suites helps interop

redhat

Standard and Software in parallel

- Parallel developments
  - Same deadline
  - Reference code
- Good points:
  - Feedback
  - Timing is good
  - Positive perception
- Bad point:
  - What if the standard doesn't pass
  - Frequent changes to the code as the draft evolves



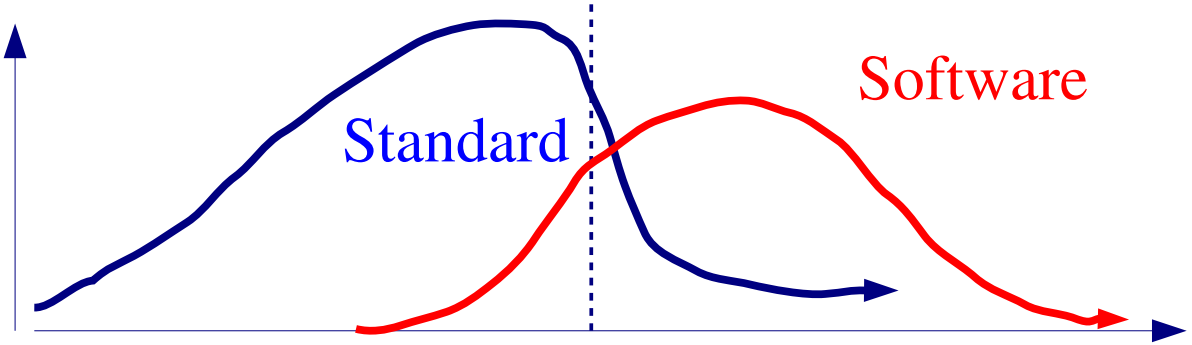
That situation is not very common

Late implementor

- React upon demand
  - Existing need
  - Spec looks okay
- Good points:
  - Spec is stable

    Existing User base
  - Minimal effort
  - Benefit from earlier implementor efforts
- Bad points:

    Too late to change the specification
  - Competing with existing implementations



That situation is very common, usually the easiest

redhat

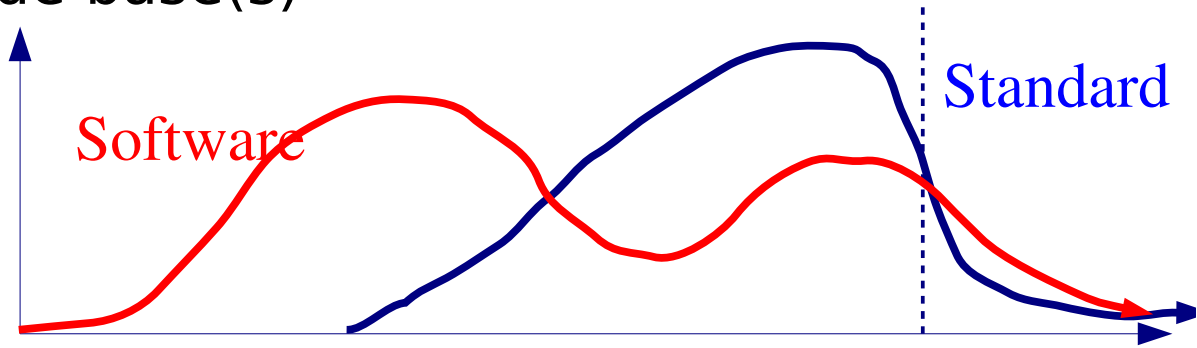
Early implementor

- Standardize existing code base(s)
  - Software works
  - Build a standard
- Good points:
  - Clear direction

    Existing User base
- Bad points:

    Existing user base
  - The specification will change, your code too
  - Competing with other people on the standard choices

That happens, this can be hell



redhat

Open Source specific

- Our code is public, are the standards (or drafts) too ?
- Do we have the resources to implement the spec fully ?
- Collaboration with the Working Group can be great:
  - Feedback loop integrate the Open Source Process
  - Who pays for the membership fees ?
  - Can be very time consuming
- One very hard issue : Patents
  - Affects us harder than proprietary code
  - Different standard bodies approaches
    - Royalties free (W3C)
    - RAND (Reasonable non discriminatory)
  - Workarounds are not always possible

redhat

Conclusions

- A lot of standards impact Open Source projects
  - Don't ignore them, be ready
  - Sometimes it is worth contributing
  - Be careful in your implementation
    - Interop is important
    - Avoid legal issues
- Some standard body are friendly to OSS
  - Free 'expert' access
  - Legal provisions to avoid patent issues

http://veillard.com/Talks/CLKBeijing2012.pdf