

"Understanding serious Debian deployments"

Debian is one of the most successful operating systems built completely on Free Software. It contains a very large amount of software that people every day acquire, study, modify, sell, use to run a business, a piece of machinery, or even a country.

It is powerful, but it has an unusually large scale and independent nature that need to be understood.

The talk will cover, both theoretically and with examples of successful deployments, why people use Debian, how Debian works, and how it is possible to interact with the Debian project.

- Why use Debian

- A system with lots of software to use and learn and grow on (it is so big that it's perfectly possible to live as "if it's not in Debian, then it doesn't exist")
 - software libraries are there to use
 - source code is there to learn and borrow
 - because of the Debian Free Software Guidelines, everything you do with Debian, you could build on top and start a business on it
 - meticulous attention to licenses
- A system your IT department can control (see the city of Munich)
- A system to build your system on (see for example Custom Debian Distributions, and the concept of a Greater Debian world)
- Customized versions for special uses

- How Debian works

- motivations of developers (?)
 - because you like it
 - because you use it
 - because you need itconvergence of a diversity of goals and motivations
cooperation of competing actors
Debian never had a business phone contact, and this is hard to understand for who is used at the corporate software world.
Actually, it has many thousand phone contacts, which is just as hard to understand, but it leads to exciting thoughts.
- common ground among all Debian participants
 - social contract
 - the social contract is the only shared vision
 - technical policies for everything else
- steering
 - with work: what you need, you do. Everyone else is a volunteer, and cannot be forced, neither in their choices nor in their deadlines. If what you need does not get done by someone else, you need to do it yourself.
 - trying to minimise isolation: try to get what you need with the minimum amount of code, reusing of existing components, involving existing groups
- entering Debian
why the New Maintainer process (we need to trust the identity of people, we need to trust their skills)
- governance
Debian Project Leader, Technical Committee, Delegates, Developers, Condorcet-based voting system

- Examples

- ERPOSS3 (aims at security)
- City of Munich (aims at vendor independence)
- Skolelinux (builds on a solid base, focusing development on the target)

- environment, and being as close as Debian as possible as the key that makes them able to focus on customisation instead of core maintenance)
- Ubuntu (fork&merge, and the importance of the merge)
 - Dhongzha Linux (localization and technological independence)
 - Edos using it for research on large scale software systems (Free Software is now found at the edge of systems innovation)
- Promote development locally
- Translation in your language (whatever it is, however unmarketable it is)
 - Give work to local people, develop know-how on the local territory
 - Take from outside only what is needed (the core of the work, or only the extra competences, or the main workforce, or even nothing)

In Europe there are more Debian Developers than in the United States: we are not at all underdeveloped in Free Software as we are with commercial software.