HelenOS – Operating System Built of Microservices

http://www.helenos.org/

CHARLES UNIVERSITY
Faculty of Mathematics and Physics

Martin Děcký

decky@d3s.mff.cuni.cz

Department of Distributed and Dependable Systems

D3S

HelenOS

Motivation

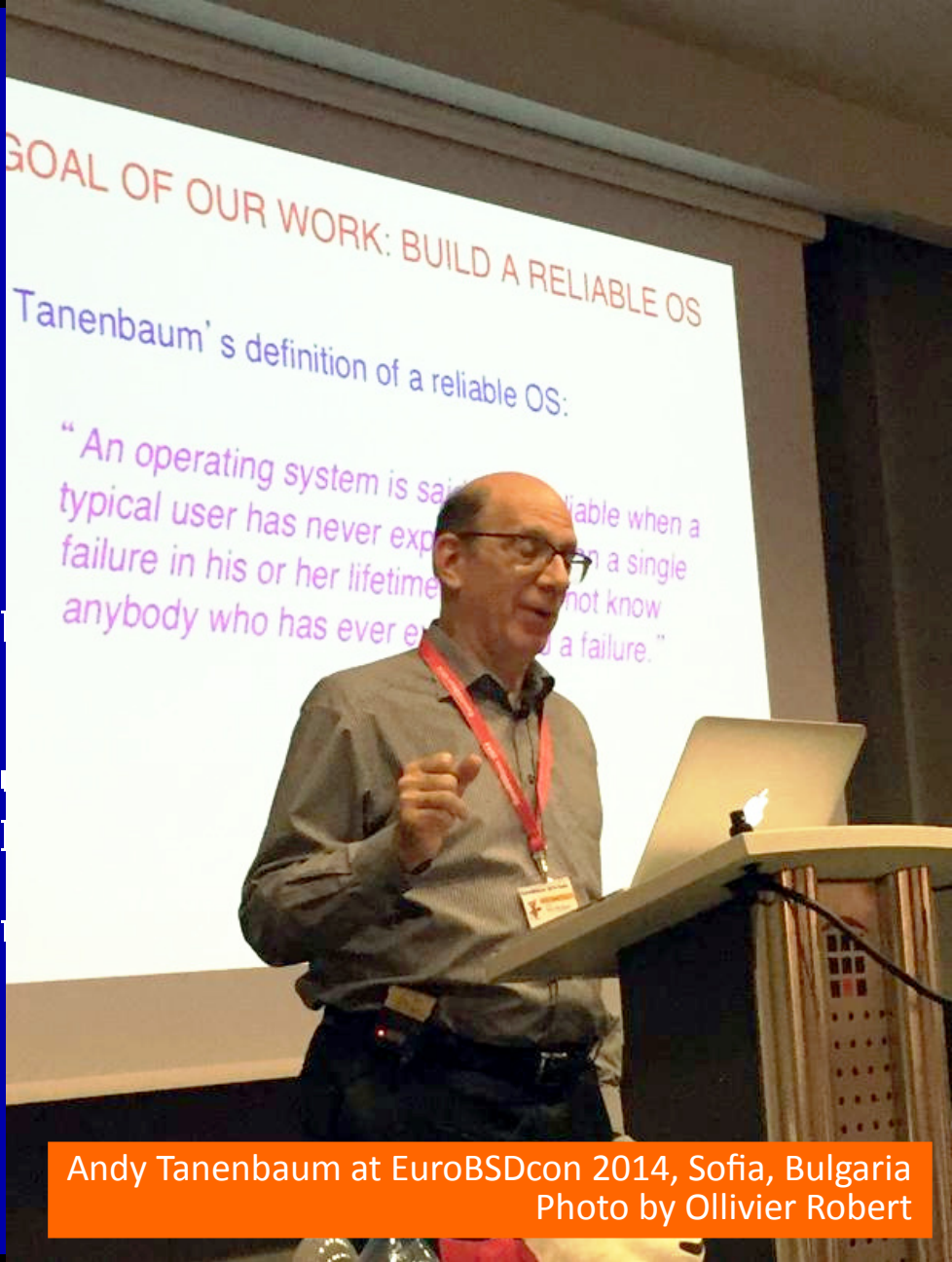

Windows
A fatal exception 0E has occurred at 0028:C562F1B7 in VXD ctpci9x(05)
+ 00001853. The current application will be terminated.
* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will
lose any unsaved information in all applications.
Press any key to continue

Windows

A fatal exception 0E has occurred at 0028
+ 00001853. The current application will

* Press any key to terminate the current
* Press CTRL+ALT+DEL again to restart yo
lose any unsaved information in all ap

Press any key to co



Andy Tanenbaum at EuroBSDcon 2014, Sofia, Bulgaria
Photo by Ollivier Robert

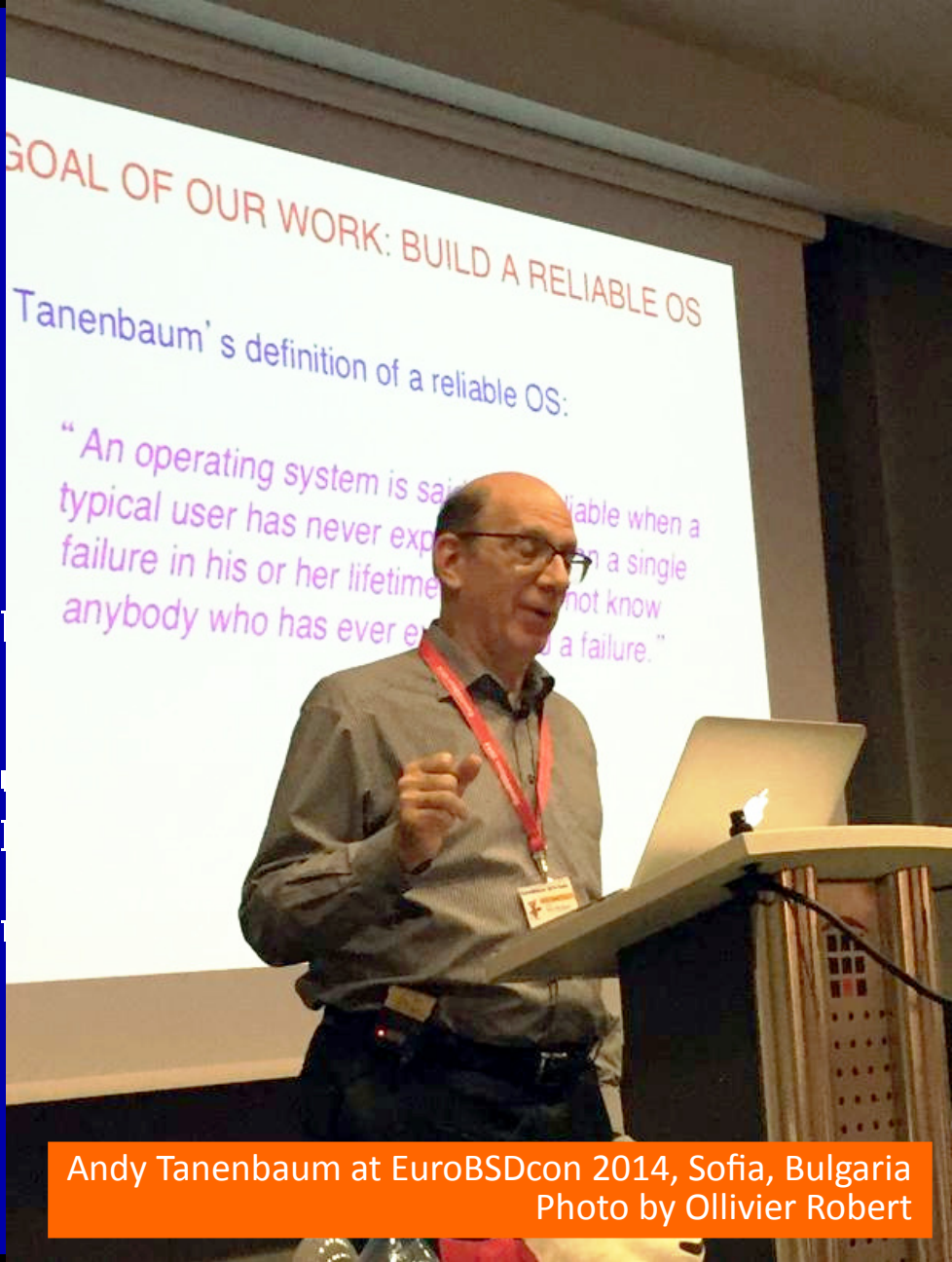
“An operating system is said to be reliable when a typical user has never experienced even a single failure in his or her lifetime and does not know anybody who has ever experienced a failure.”
[Tanenbaum 2014]

Windows

A fatal exception 0E has occurred at 0028
+ 00001853. The current application will

* Press any key to terminate the current
* Press CTRL+ALT+DEL again to restart yo
lose any unsaved information in all ap

Press any key to co



Andy Tanenbaum at EuroBSDcon 2014, Sofia, Bulgaria
Photo by Ollivier Robert

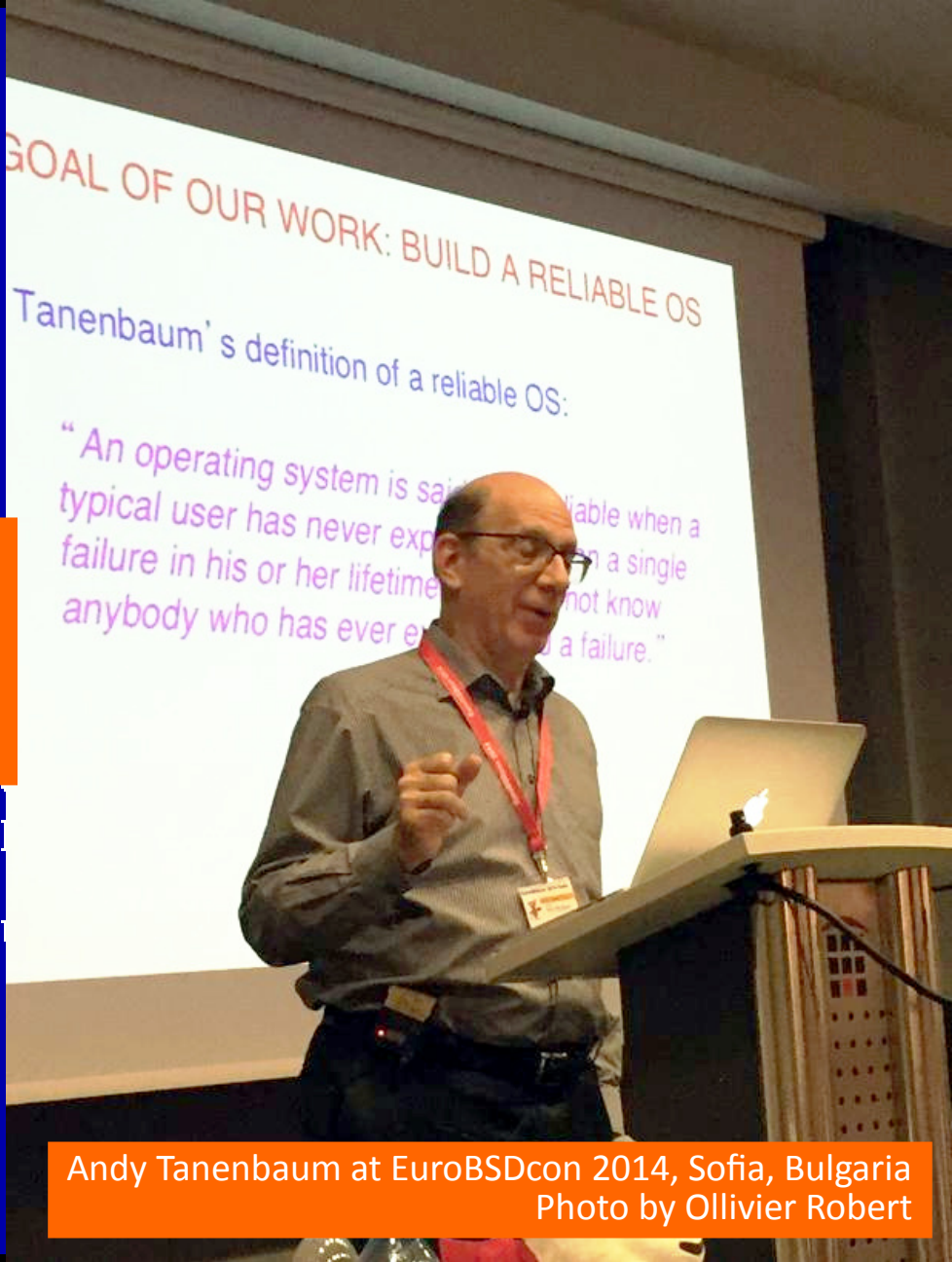
“An operating system is said to be reliable when a typical user has never experienced even a single failure in his or her lifetime and does not know anybody who has ever experienced a failure.”
[Tanenbaum 2014]

Windows

“There are no demonstrated examples of highly secure or highly robust unstructured (monolithic) systems in the history of computing.”
[Shapiro 2006]

lose any unsaved information in all ap

Press any key to co



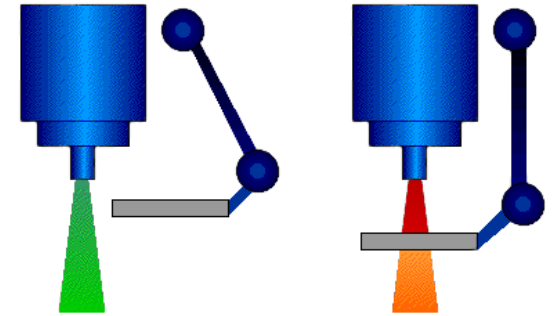
Andy Tanenbaum at EuroBSDcon 2014, Sofia, Bulgaria
Photo by Ollivier Robert

Software Dependability

- **IEEE definition**
  - *"Dependability is a measurable and **provable** degree of system's availability, reliability and its maintenance support."*
- **Laprie J. C.: *Dependable Computing and Fault Tolerance***
  - *"Dependability is also affected by other measures, such as safety, security, integrity and confidentiality."*

Cautionary Tale: Therac-25

- **Radiotherapeutic medical device**
  - Derived from Therac-6
    - Two basic modes of operation
    - Safety features in hardware instead of software
  - 6 confirmed accidents between 1985 – 1987
    - 3 confirmed deaths with a root cause of radiation burns
    - Software race condition
      - Poor software design and QA
      - Misleading user interface
      - Root cause: Poor understanding of software reliability issues



Cautionary Tale: Ariane 5

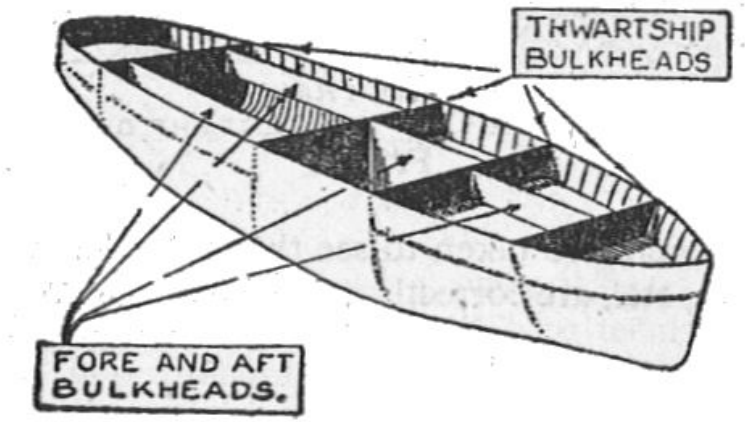
- **ESA heavy lift launch vehicle**
  - Derived from Ariane 4
    - A reliable and time-proven vehicle
  - Exploded on its maiden voyage on June 4th 1996
    - 39 seconds after lift-off
    - $370 million in damage
    - 64bit float containing velocity truncated to a 16bit integer in a non-critical software component
      - Caused an uncaught exception that propagated to the control component
      - A safety component triggered mission abort
      - The non-critical component served no actual purpose

Microservices

- **Wikipedia definition**
  - *"A variant of service-oriented architecture (SOA) that structures an application as a collection of loosely coupled fine-grained services."*
- **Benefits**
  - Improved modularity, understandability, verifiability
  - Improved run-time fault isolation, anti-fragility

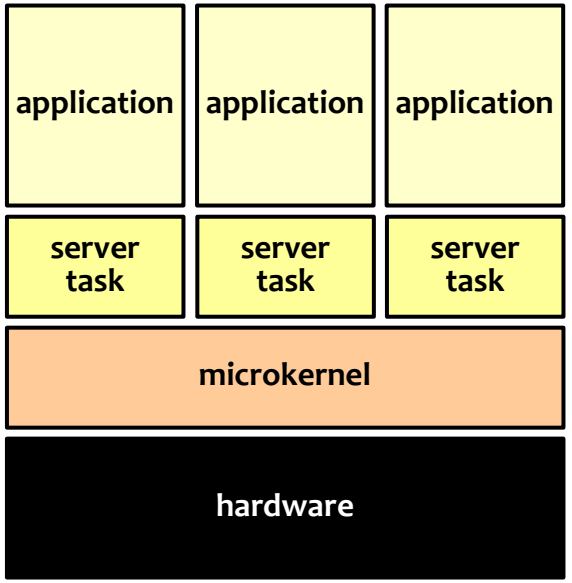
Microservices

- **Not an entirely new idea ...**



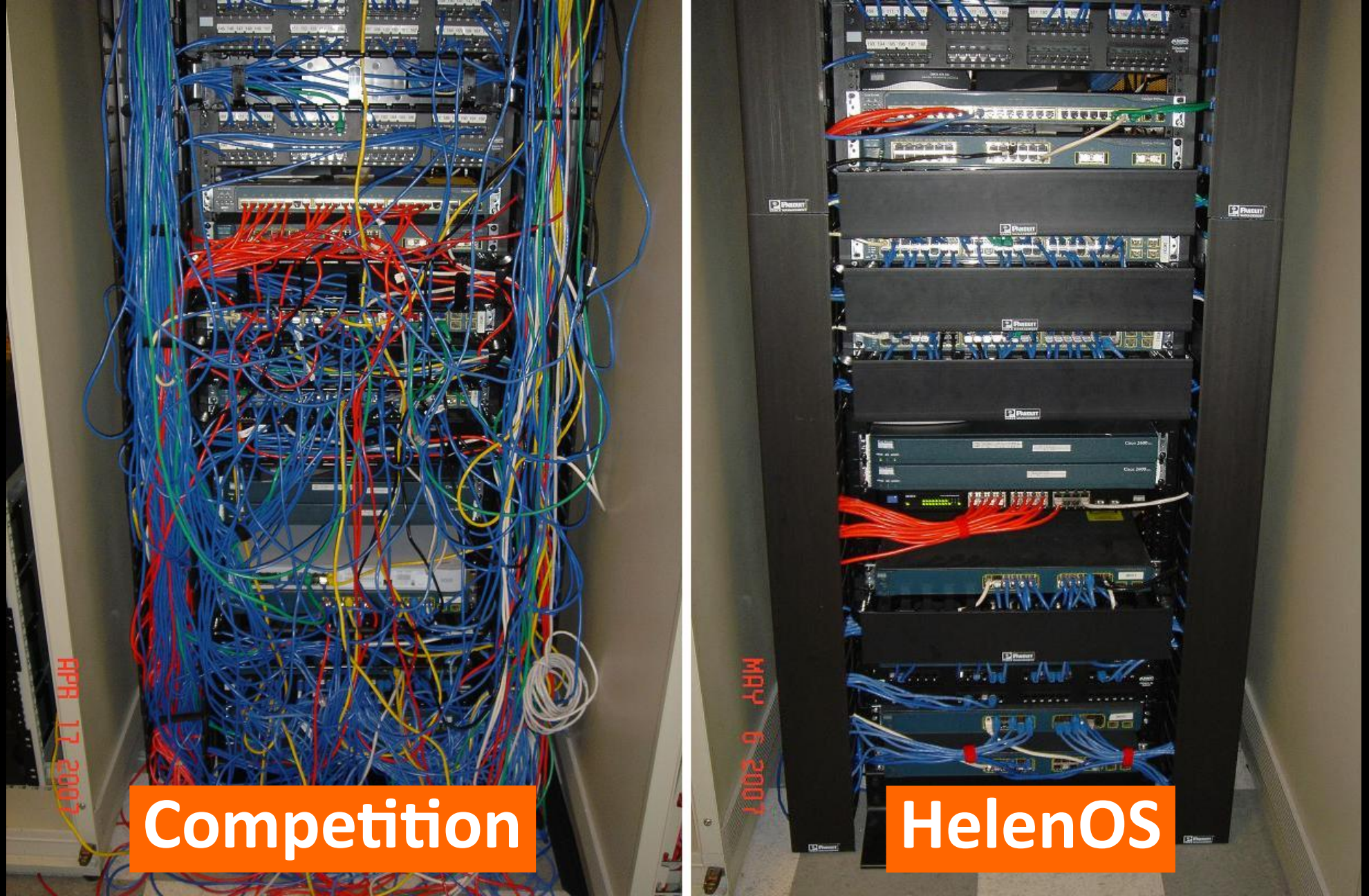
HelenOS in a Nutshell

HelenOS

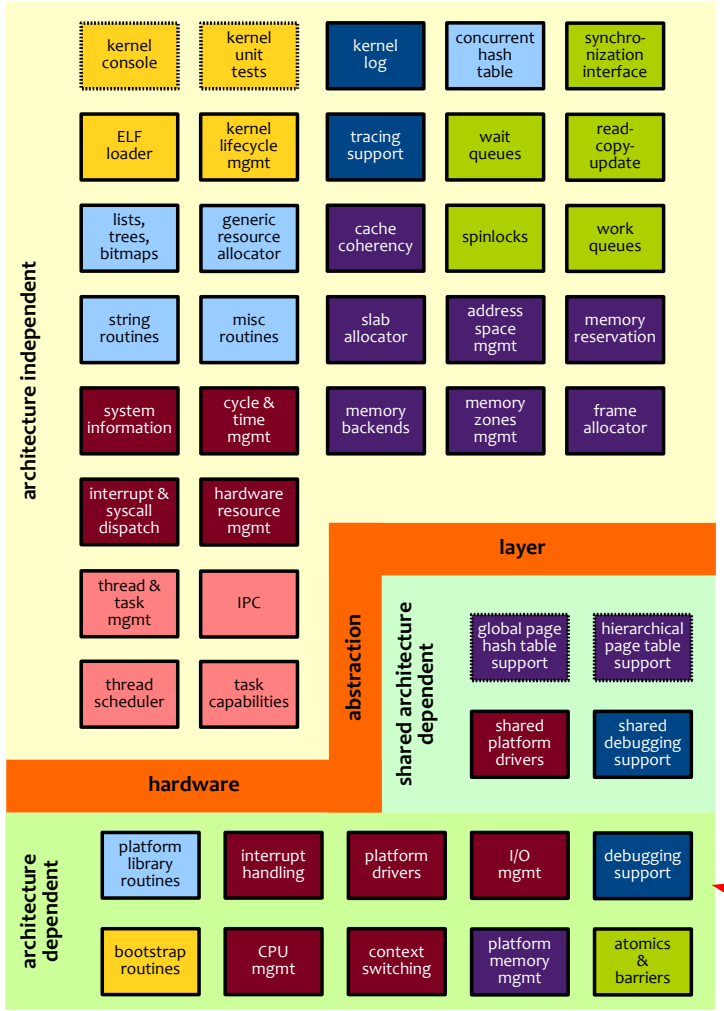


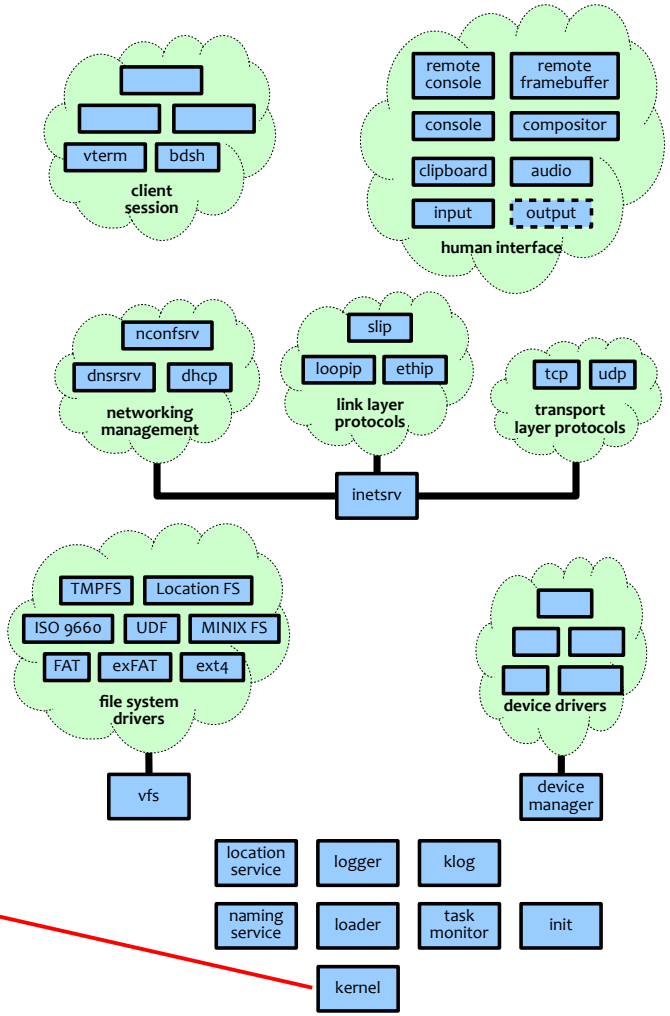
open source general-purpose multiplatform microkernel multiserver operating system designed and implemented from scratch

Department of Distributed and Dependable Systems D3S

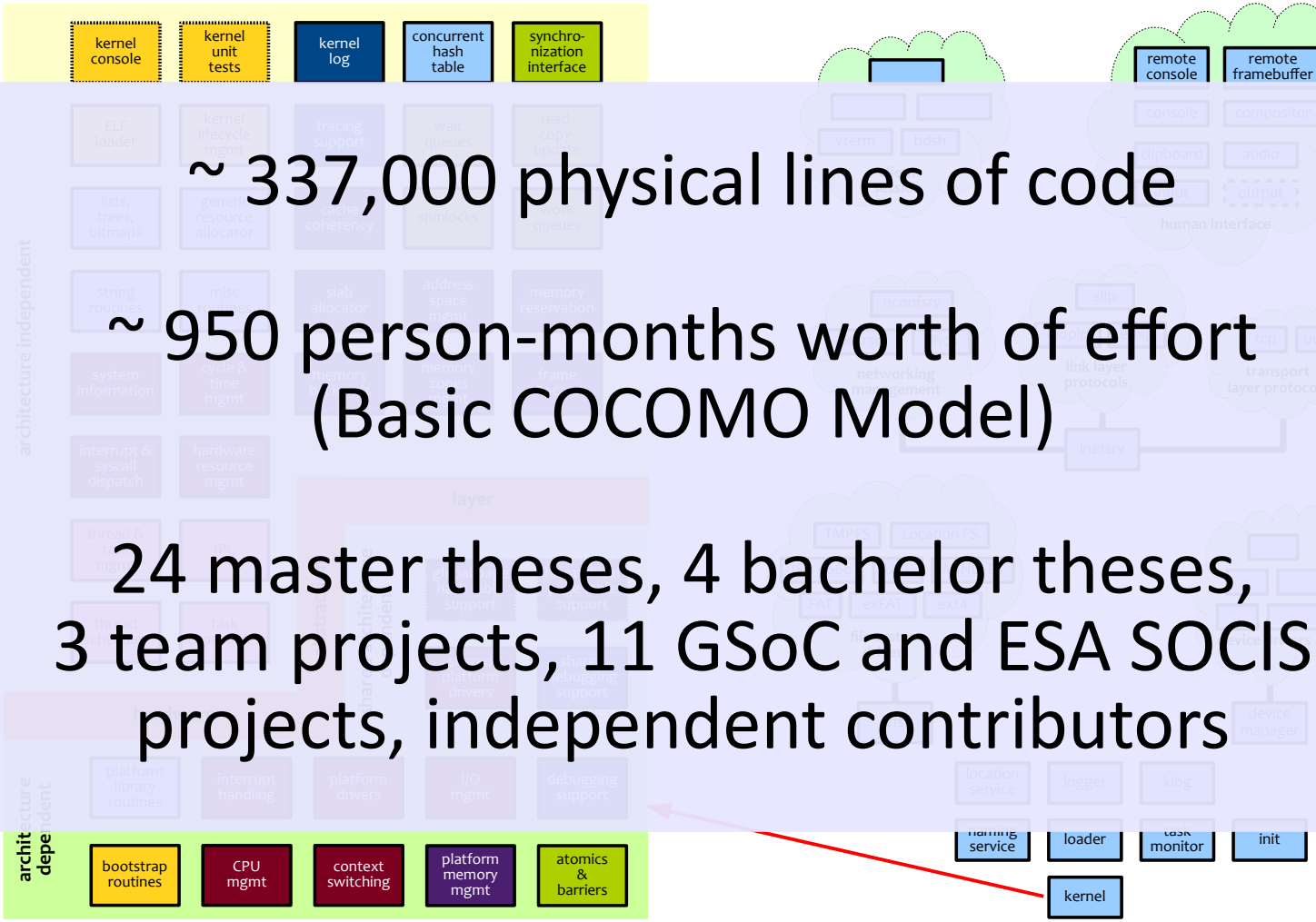

Competition
HelenOS

HelenOS in a Nutshell





HelenOS in a Nutshell

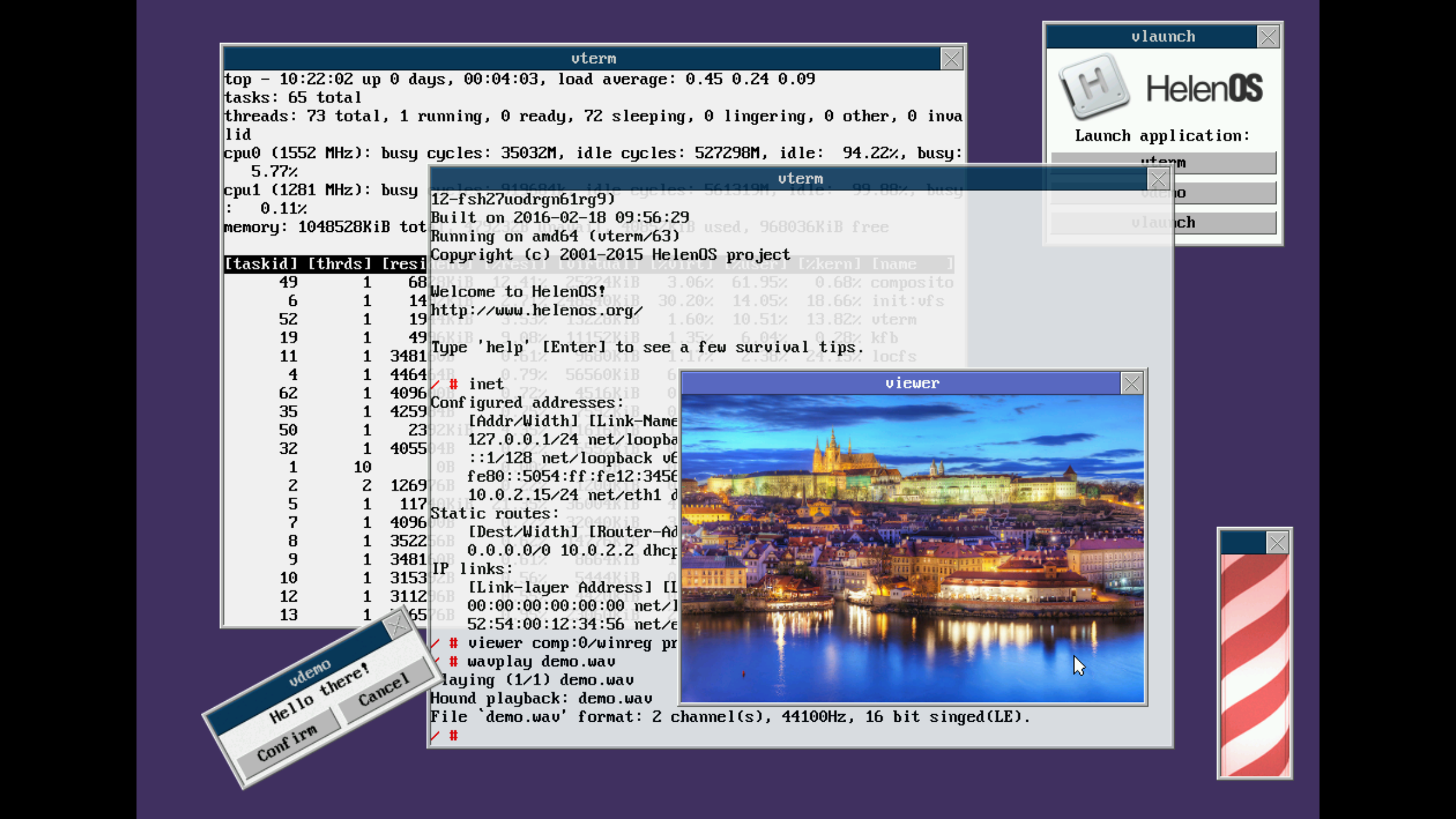


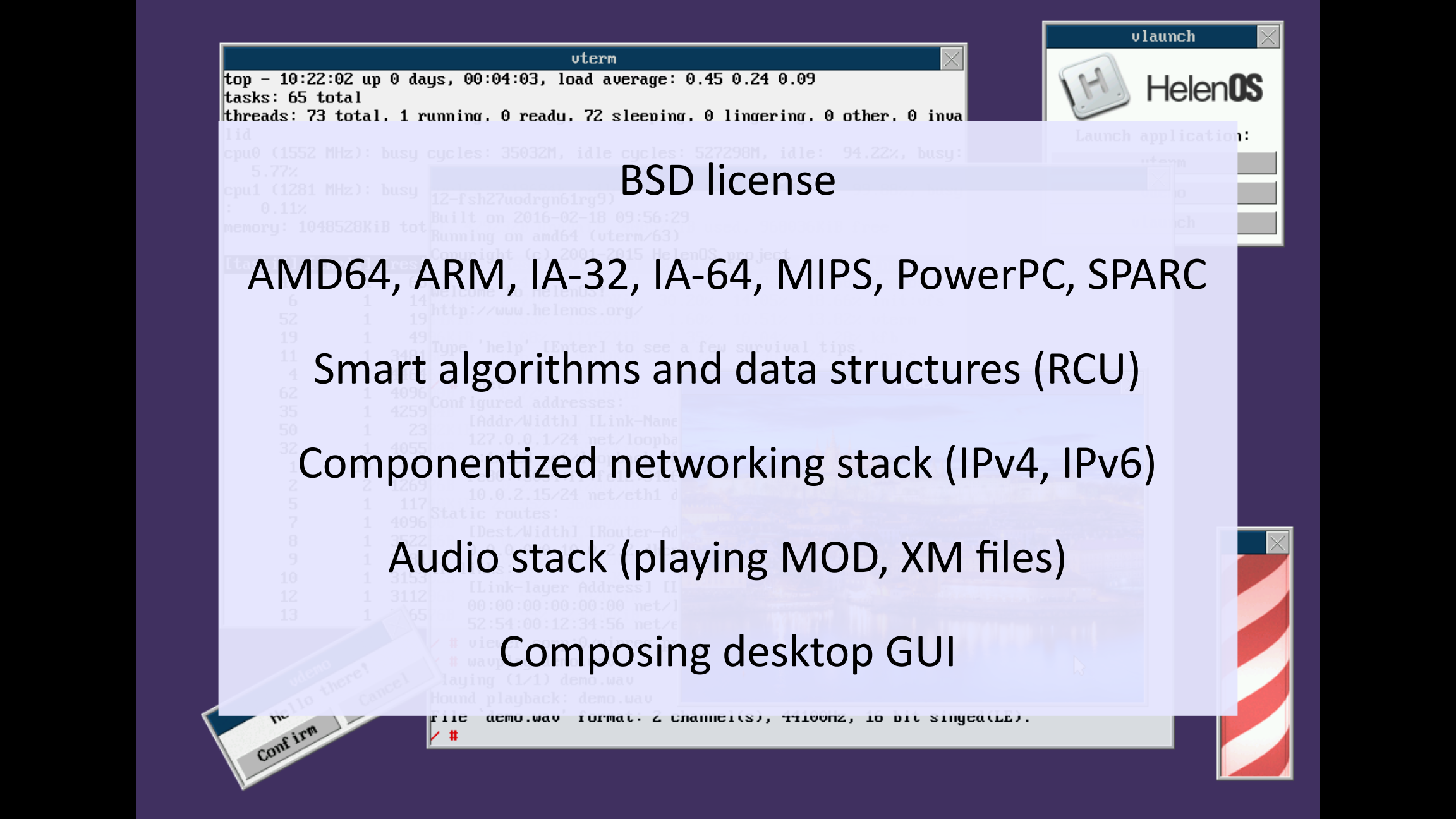
~ 337,000 physical lines of code

~ 950 person-months worth of effort
(Basic COCOMO Model)

24 master theses, 4 bachelor theses,
3 team projects, 11 GSoC and ESA SOCIS
projects, independent contributors

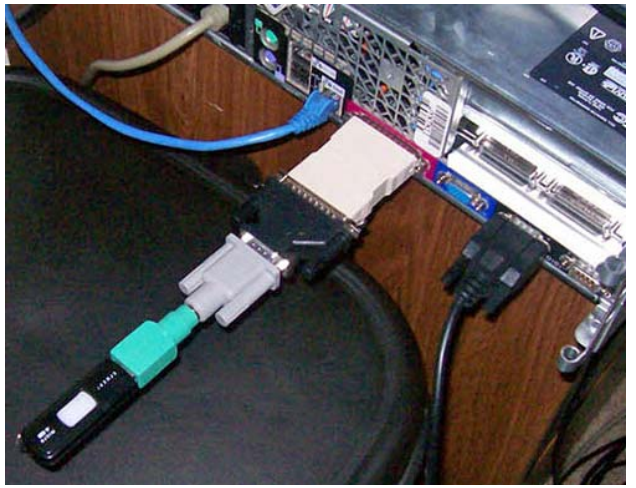
Department of Distributed and Dependable Systems


vterm
top - 10:22:02 up 0 days, 00:04:03, load average: 0.45 0.24 0.09
tasks: 65 total
threads: 73 total, 1 running, 0 ready, 72 sleeping, 0 lingering, 0 other, 0 invalid
cpu0 (1552 MHz): busy cycles: 35032M, idle cycles: 527298M, idle: 94.22%, busy: 5.77%
cpu1 (1281 MHz): busy
: 0.11%
memory: 1048528KiB tot
[taskid] [thrds] [resi
49 1 68
6 1 14
52 1 19
19 1 49
11 1 3481
4 1 4464
62 1 4096
35 1 4259
50 1 23
32 1 4055
1 10
2 2 1269
5 1 117
7 1 4096
8 1 3522
9 1 3481
10 1 3153
12 1 3112
13 1
vterm
12-fsh27uodrgn61rg9)
Built on 2016-02-18 09:56:29
Running on amd64 (vterm/63)
Copyright (c) 2001-2015 HelenOS project
Welcome to HelenOS!
http://www.helenos.org/
Type 'help' [Enter] to see a few survival tips.
/ # inet
Configured addresses:
[Addr/Width] [Link-Name
127.0.0.1/24 net/loopba
::1/128 net/loopback v6
fe80::5054:ff:fe12:3456
10.0.2.15/24 net/eth1 d
Static routes:
[Dest/Width] [Router-Ad
0.0.0.0/0 10.0.2.2 dhcp
IP links:
[Link-layer Address] [I
00:00:00:00:00:00 net/l
52:54:00:12:34:56 net/e
/ # viewer comp:0/winreg pr
/ # wavplay demo.wav
laying (1/1) demo.wav
Hound playback: demo.wav
File `demo.wav' format: 2 channel(s), 44100Hz, 16 bit singed(LE).
/ #
viewer
vlaunch
HelenOS
Launch application:
vterm
vdemo
Hello there!
Confirm
Cancel

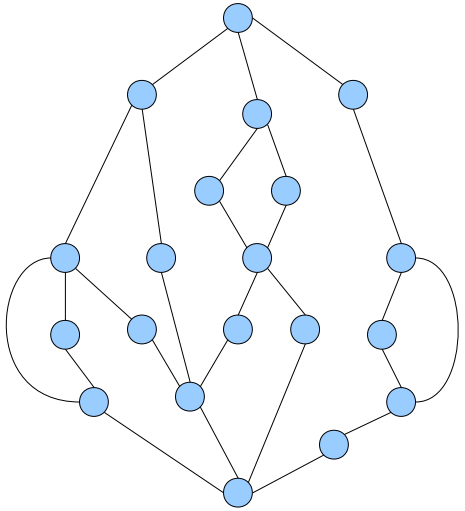

vterm
top - 10:22:02 up 0 days, 00:04:03, load average: 0.45 0.24 0.09
tasks: 65 total
vlaunch
HelenOS
BSD license
AMD64, ARM, IA-32, IA-64, MIPS, PowerPC, SPARC
Smart algorithms and data structures (RCU)
Componentized networking stack (IPv4, IPv6)
Audio stack (playing MOD, XM files)
Composing desktop GUI

The Case for "Reinventing the Wheel"

- **Clean-slate design**
  - Legacy designs and APIs may be broken, insecure, thread-unsafe, morally obsolete
    - However, not saying that all legacy is broken
  - Thinking out of the box



Formal Verification in a Nutshell



label transition system

□(open → ◇close)

temporal logic formula

model checker

or

OK

line 10: …
line 14: …
line 22: …
line 47: …

error trace

Formal Verification Example

- Changeset: *mainline,530*
- Commit log: *Fix uninitialized field in error path*
  - Found by *Clang Analyzer*
  - `kernel/generic/src/main/kinit.c`

```
 if (init.tasks[i].addr % FRAME_SIZE) {
     printf("init[%" PRIs "].addr is not frame aligned\n", i);
+    programs[i].task = NULL;
     continue;
 }
```

Conclusion & Future Work

- **State-of-the-art**
  - Solid and comprehensive research and development platform
  - Viability demonstrated on several practical use cases
- **Future work**
  - Performance tuning
  - Focusing on commercialization

Q&A

Join us!

HelenOS

www.helenos.org

Department of Distributed and Dependable Systems D3S

Backup slides

HelenOS Contributors

Sean Bartell
Tomáš Benhák
Dmitry Bolkhovityanov
Sergey Bondari
Tobias Börtitz
Zdeněk Bouška
Tomáš Brambora
Jan Buchar
Lubomír Bulej
Tomáš Bureš
Josef Čejka
Aurelio Colosimo
Manuele Conti
Martin Děcký
Matúš Dekánek
Jan Dolejš
Andrey Erokhin
Matteo Facchinetti
Beniamino Galvani
Matthieu Gueguen
Zbigniew Halas
Štepán Henek
Vojtěch Horký
Adam Hraška
Mohammed Hussain
Adrian Jamróz
Pavel Jančík
Martin Jelen
Petr Jerman
Jakub Jermář
Fan Jinfei
Jiří Kavalík
Michal Kebrt
Jakub Klama
Matěj Klonfar
Jan Kolárik
Michal Konopa
Petr Koupý
Stanislav Kozina
Sandeep Kumar
Maurizio Lombardi
Peter Majer
Jan Mareš
Julia Medvedeva
Lukáš Mejdrech
Vojtěch Mencl
Jiří Michalec
Ondřej Palkovský
Vineeth Pillai
Tim Post
Vivek Prakash
František Princ
Alexander Prutkov
Marin Ramesa
Pavel Římský
Oleg Romanenko
Jeff Rous
Thomas Sanchez
Ondřej Šerý
Ľuboš Slovák
Antonín Steinhauser
Petr Štěpán
Martin Sucha
Jiří Svoboda
Agnieszka Tabaka
Dominik Táborský
Jiří Tlach
Lenka Trochtová
Petr Tůma
Jakub Váňa
Radim Vansa
Laura-Mihaela Vasilescu
Ján Veselý
Jan Záloha
Jiří Zárevúcky

Google

SUMMER of CODE

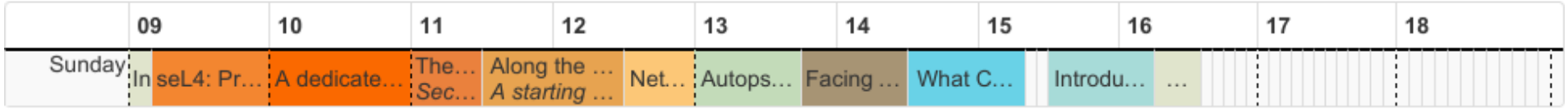
TENTH YEAR

2014

esa

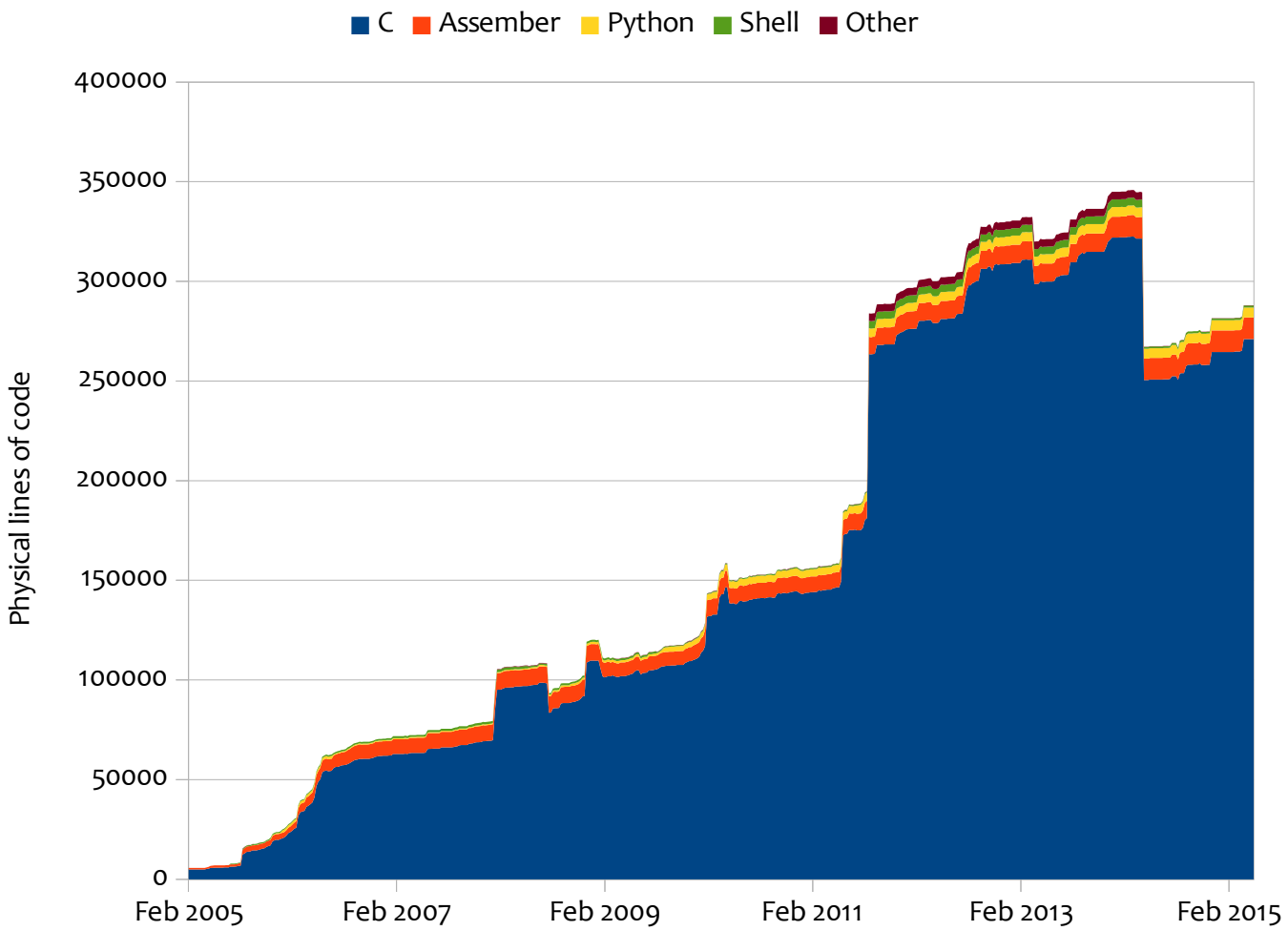
Microkernels devroom

Room: K.3.201
Calendar: iCal, xCal

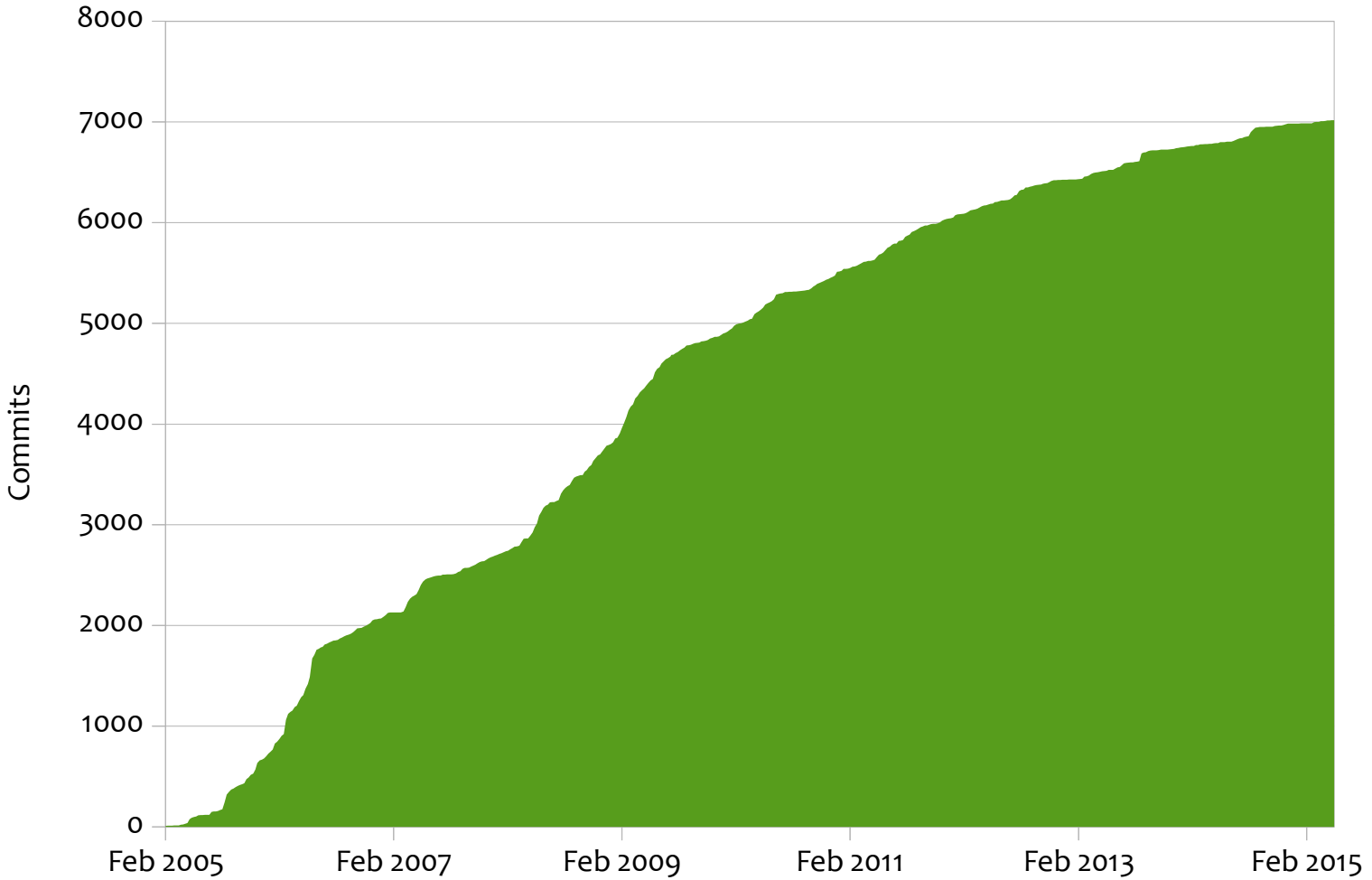


Event	Speakers	Start	End
Sunday			
Introduction	Vasily A. Sartakov	09:00	09:10
seL4: Present and Future	Gernot Heiser	09:10	10:00
A dedicated kernel named TORO	Matias Vara	10:00	11:00
The FLK project *Security by the language, no MMU, no processes*	José Bollo	11:00	11:30
Along the GNU Hurd RPC way *A starting guide to contributing to the GNU Hurd*	Samuel Thibault	11:30	12:30
Networking (lunch)		12:30	13:00
Autopsy of a multiserver deadlock in the HelenOS filesystem layer	Jakub Jermář	13:00	13:45
Facing the Reality: What's new in the L4Re Operating System	Adam Lackorzynski	13:45	14:30
What Could Microkernels Learn from Monolithic Kernels (and Vice Versa)	Martin Děcký	14:30	15:20
Introducing a radically componentized GUI architecture	Norman Feske	15:30	16:15
Cloud services on top of uKernel	Vasily A. Sartakov	16:15	16:35

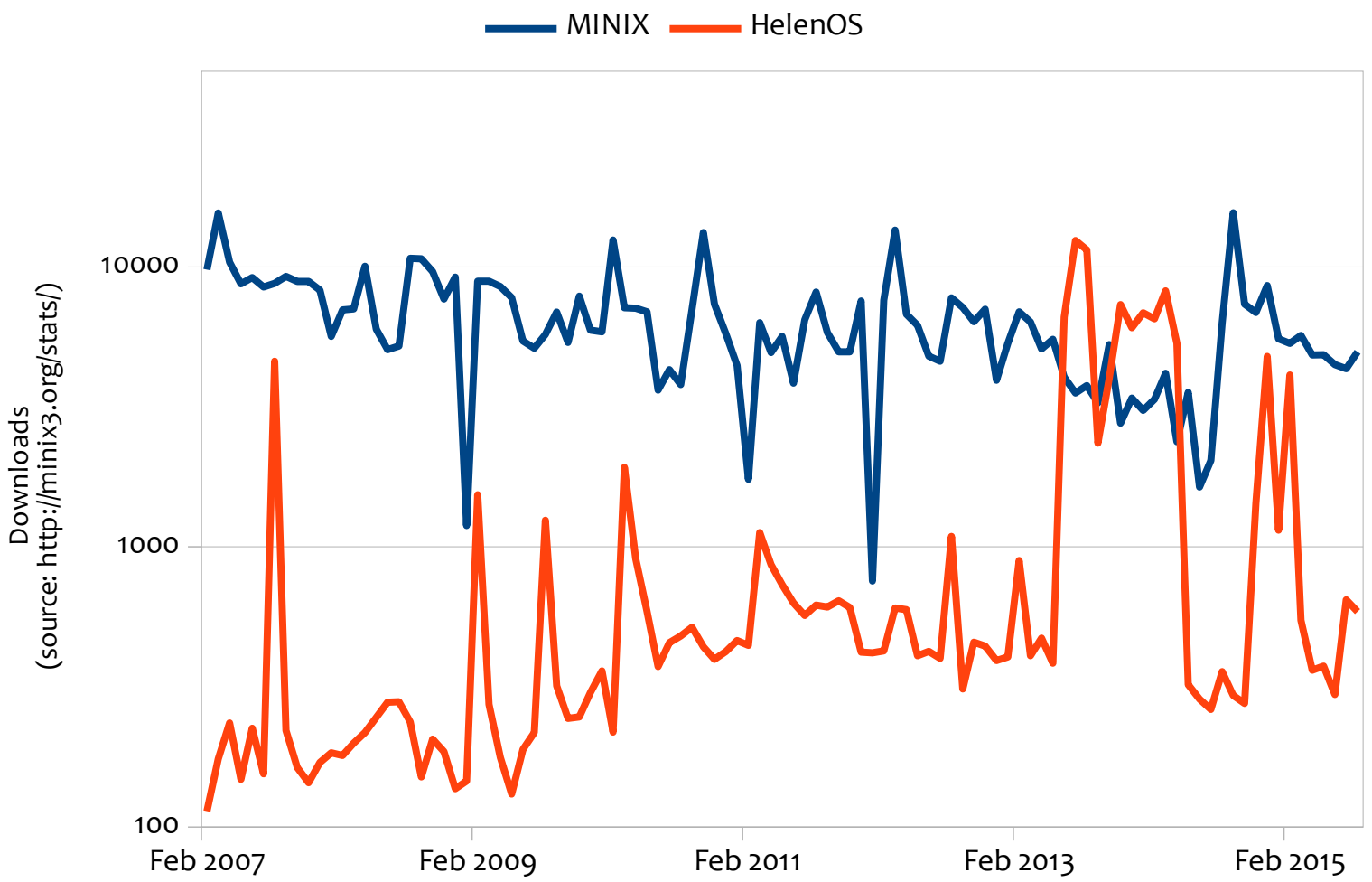
Development Process



Commits


8000
7000
6000
5000
4000
3000
2000
1000
0
Commits
Feb 2005
Feb 2007
Feb 2009
Feb 2011
Feb 2013
Feb 2015

Downloads



19.66 %

www.microkernel.info

µ–kernel.info

Microkernels are operating systems that outsource the traditional operating system functionality to ordinary user processes while providing them with mechanisms requisite for implementing it. Microkernel-based operating systems come in many different flavours, each having a distinctive set of goals, features and approaches. Some of the most often cited reasons for structuring the system as a microkernel is flexibility, security and fault tolerance. Many microkernels can take on the role of a hypervisor too. Microkernels and their user environments are most often implemented in the C or C++ programming languages with a little bit of assembly, but other implementation languages are possible too. In fact, each component of a microkernel-based system can be implemented in a different programming language.

Here is a list of active free, open source microkernel projects. If your project is missing, please let us know!

Escape

A UNIX-like microkernel operating system, that runs on x86, x86_64, ECO32 and MMIX. It is implemented from scratch and uses nearly no third-party components. To fit nicely into the UNIX philosophy, Escape uses a virtual file system to provide drivers and services. Both can present themselves as a file system or file to the user. (*github.com/Nils-TUD/Escape*)

M³

A microkernel-based system for heterogeneous manycores, that is developed as a hardware/OS co-design at the TU Dresden. It aims to support arbitrary cores (general purpose cores, DSPs, FPGAs, ASICs, ...) as first-class citizens. This is achieved by abstracting the heterogeneity of the cores via a new hardware component per core, called data transfer unit. (*github.com/TUD-OS/M3*)

F9

An experimental microkernel used to construct flexible real-time and embedded systems for ARM Cortex-M series microprocessors with power efficiency and security in mind. (*github.com/f9micro*)

MINIX 3

A free, open-source, operating system designed to be highly reliable, flexible, and secure. It is based on a tiny microkernel running in kernel mode with the rest of the operating system running as a number of isolated, protected, processes in user mode. (*minix3.org*)

Genode

A tool kit for building highly secure special-purpose operating systems. It scales from embedded systems with as little as 4 MB of memory to highly dynamic general-purpose workloads. (*genode.org*)

The Muen Separation Kernel

The world's first Open Source microkernel that has been formally proven to contain no runtime errors at the source code level. It is developed in Switzerland by the Institute for Internet Technologies and Applications (ITA) at the University of Applied Sciences Rapperswil (HSR). (*muen.sk*)

HelenOS

About Uncertainty

- **Aleatory (irreducible) uncertainty**
  - Uncertainty **in** the world
    - Unknown unknowns
    - It is impossible to predict exactly how many heads will occur in 100 trials of tossing a coin
    - Frequentist interpretation of probability needed
- **Epistemic (reducible) uncertainty**
  - Uncertainty about the world
    - Know unknows
    - Given a coin, I do not know the probability of tossing a head
      - But I can estimate it by doing experiments, learning about the physical properties of the coin, studying historical records about similar coins, etc.

About Uncertainty (2)

- **Definition: Perfect software**
  - Software that will never experience a failure in operation, no matter how much operational exposure it has
- **Definition: Possibly perfect software**
  - Software that might be perfect (we don't know it) with some probability
    - Probability of perfection

About Uncertainty (3)

- **How the probability of perfection relates to reliability?**
  - Using the theorem of total probability
    - *P*(SW fails [on a randomly selected demand]) =
      P(SW fails | SW is perfect) * *P*(SW is perfect) +
      P(SW fails | SW is imperfect) * *P*(SW is imperfect)
      - The first term is zero, because the software does not fail if it is perfect
      - Hence, define
        - p_{np} .. probability the software is imperfect
        - p_{fnp} .. probability that software fails if it is imperfect
      - Then $P(\text{SW fails}) \leq p_{fnp} * p_{np}$

About Uncertainty (4)

- **Interpreting P(SW fails) ≤ p_{fnp} * p_{np}**
  - p_{fnp} and p_{np} are subjective probabilities (degree of belief)
    - Conservative approximation
      - Assume software always fails if it is imperfect (p_{fnp} = 1)
      - Then P(SW fails) ≤ P(SW is imperfect)
- **Conclusion**
  - Even the most crude validation / verification improves lowers the probability that the software is imperfect
    - This is the upper bound of the probability that the software fails
    - The confidence of the improvement can be even quantified
    - This relieves formal verification tools from the burden of absolute perfection

Image References

- Therac-25 photo & schematics, Troy Gallagher, included under the fair use doctrine
- Ariane 5, Ignis, Creative Commons
- Compartments and watertight subdivision, Andy Dingley, public domain
- Gears, susannp4, public domain