HelenOS in the Year of the Fire Monkey

http://www.helenos.org/

CHARLES UNIVERSITY
Faculty of Mathematics and Physics

HelenOS

Jakub Jermář

jakub@jermar.eu

Martin Děcký

decky@d3s.mff.cuni.cz

HelenOS in a Nutshell

www.helenos.org

HelenOS

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

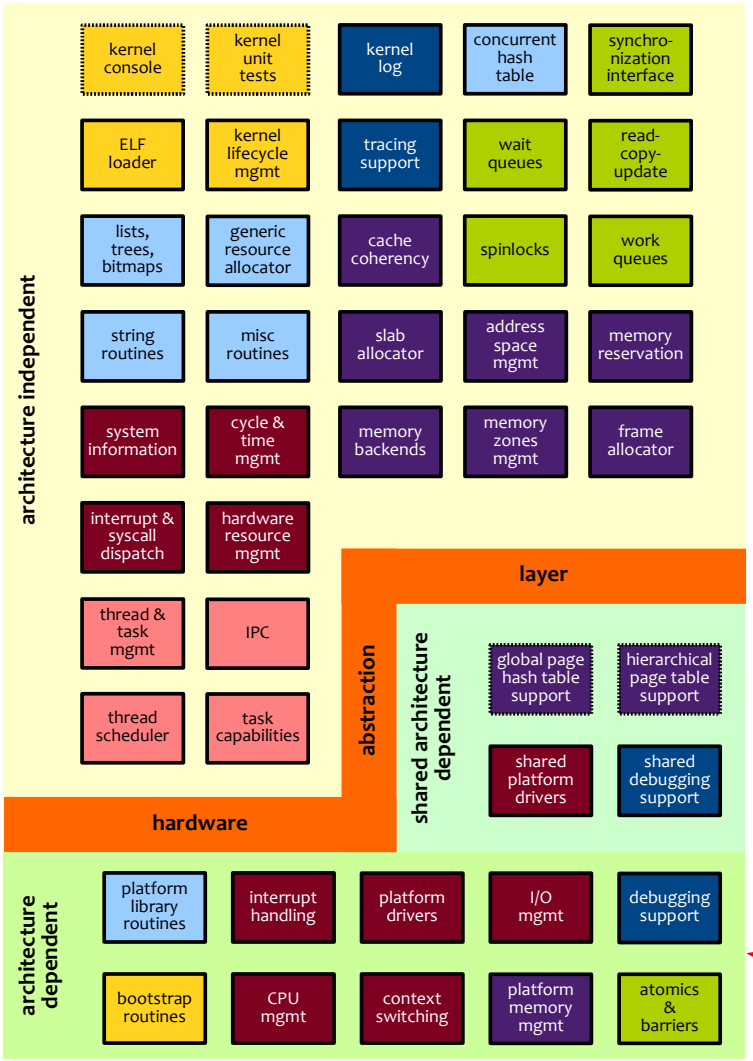
HelenOS in a Nutshell (2)

- **Open source**
  - Our own code is BSD, some 3rd party components are GPL
- **General-purpose**
  - Not biased towards any single deployment
- **Multiplatform**
  - amd64, arm32, ia32, ia64, mips32, ppc32, sparc64
- **Microkernel**
  - Obviously :)
- **Multiserver**
  - User space built from fine-grained components (microservices)

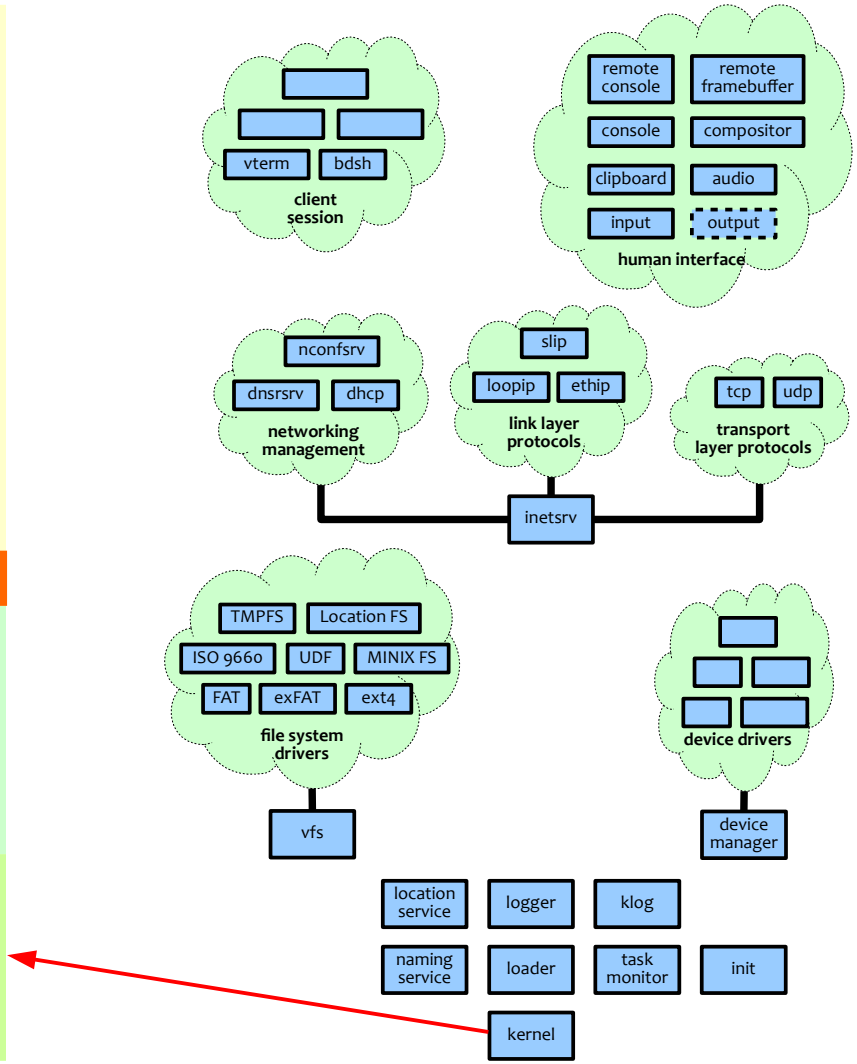
HelenOS in a Nutshell (3)

- **Designed and implemented from scratch**
  - Architecture based on a set of guiding design principles
  - Native API designed to reflect the architecture
    - Custom asynchronous IPC combining kernel preemptive and user space cooperative threads
      - Implementing concepts akin to futures and promises
  - Compatibility with legacy APIs a non-goal
    - However, providing API adaptation layer
      - Features implemented on-demand
      - binutils, fdlibm, GCC, Jainja, libgmp, libiconv, libisl, libmpc, libmpfr, libpng, MSIM, PCC, Python 2, zlib

HelenOS in a Nutshell (4)

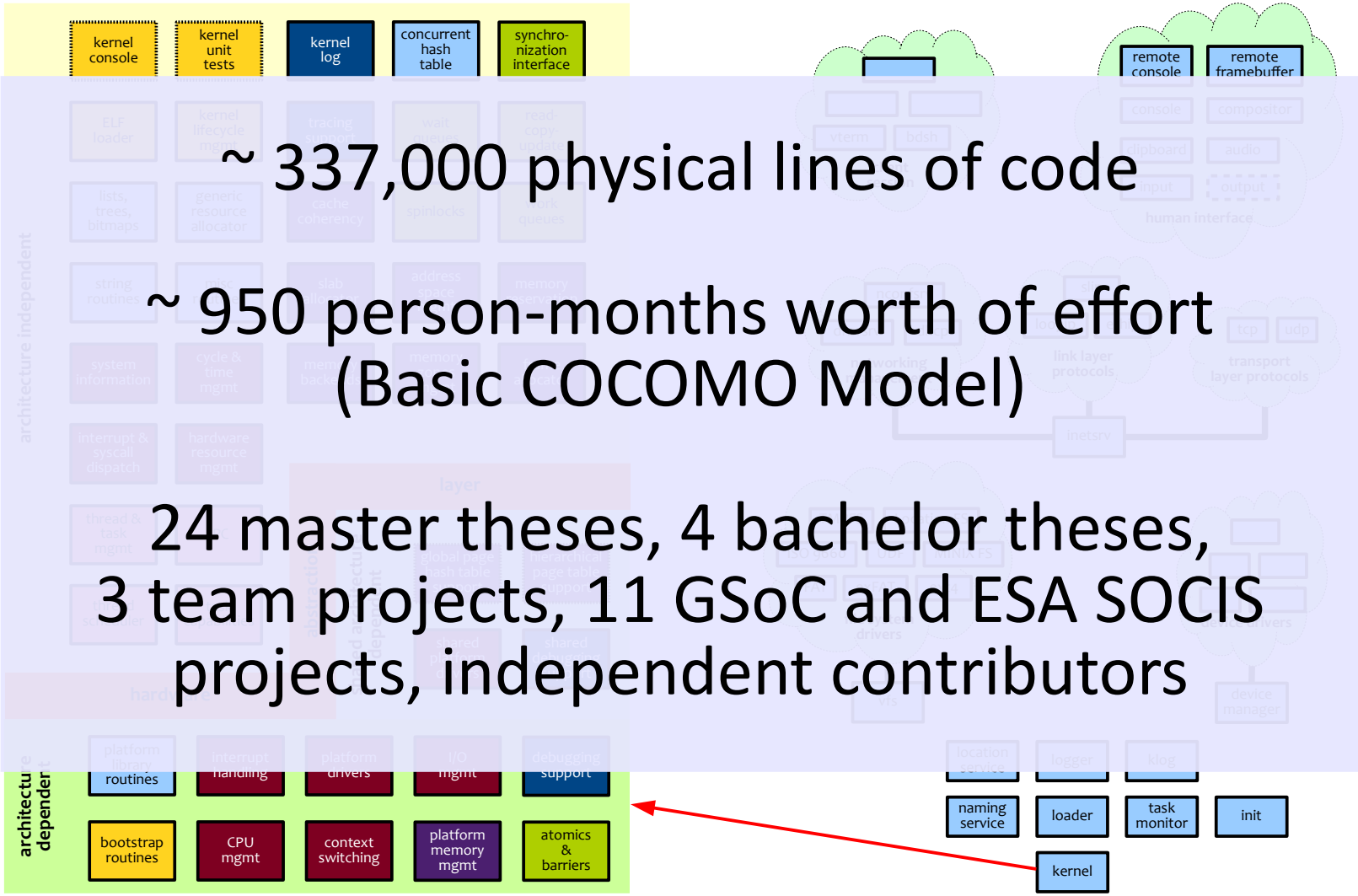


microkernel subsystems



components

HelenOS in a Nutshell (4)



~ 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

HelenOS Contributors

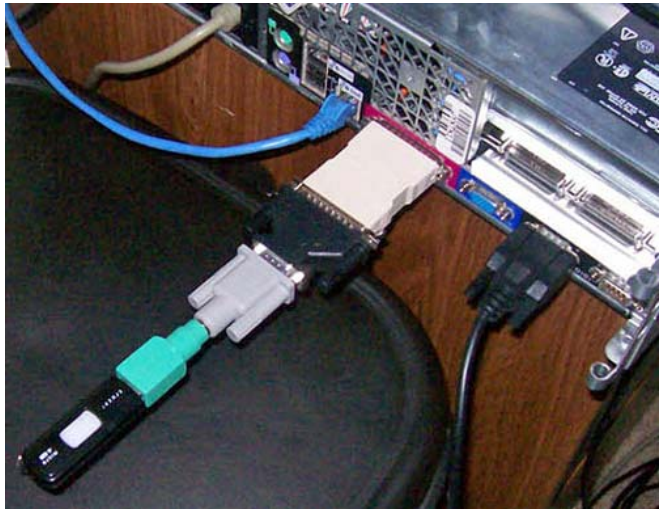
Sean Bartell
Tomáš Benhák
Dmitry Bolkhovityanov
Sergey Bondari
Tobias Börtitz
Zdeněk Bouška
Tomáš Brambora
Jan Buchar
Lubomír Bulej
Tomáš Bureš
Josef Čejka
Aurelio Colosimo
Manuele Conti
Martin Děcký
Matúš Dekánek
Jan Dolejš
Andrey Erokhin
Matteo Facchinetti
Beniamino Galvani
Matthieu Gueguen
Zbigniew 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

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
  - No glue code, mandatory adaptation layers, franken-components

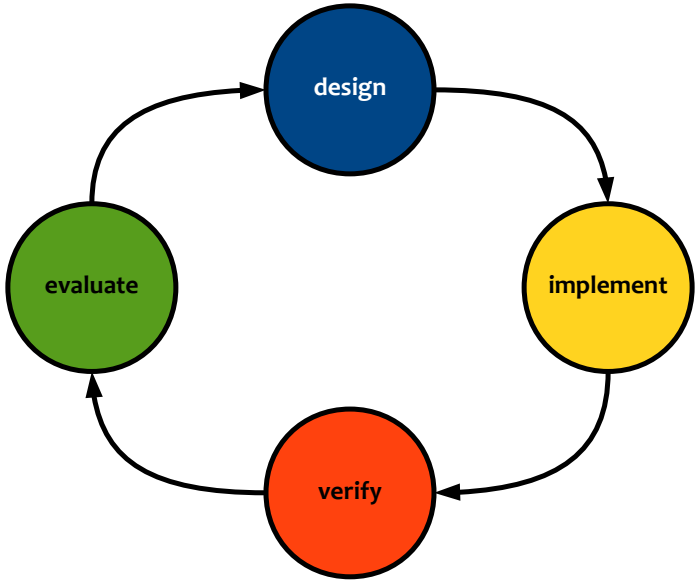
The Case for "Reinventing the Wheel"

- **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
  - No glue code, mandatory adaptation layers, franken-components



The Case for "Reinventing the Wheel"

- **No maintenance burden due to a fork**
  - A forked component tends to become a software fossil
    - Security bugs, new features, diverging licenses, toolchain updates, dependencies
  - HelenOS mainline is always fresh
    - It does not age
    - Opposed to HelenOS coastline (a.k.a. the ports tree)



Intermezzo:
Cross-pollination

Microkernels - The

www.microkernel.info

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

Microkernel Devroom in GSoC

HelenOS

- **Proposal for the Microkernel devroom**
  - Apply for Google Summer of Code 2017 as an umbrella organization
    - HelenOS experience
      - Accepted in 2011, 2012, 2014
      - Not accepted in 2009, 2010, 2013, 2015, 2016
      - But still GSoC is extremely valuable to us
    - We don't know Google's criteria, but maybe too many operating system projects applying
    - An umbrella organization might be a solution
      - Positive feedback from ksys labs, Genode, MINIX3

Microkernel Devroom in GSoC (2)

- **Technicalities**
  - Application deadline February 9th 2017 16:00 UTC
  - Who should fill in the application?
    - HelenOS team volunteers (we have some experience)
      - Any input is always welcomed
  - Where should the ideas page be hosted?
    - At microkernel.info (a quasi-neutral location)
      - Don't hesitate to send us your project topics
  - Should we discourage individual projects from applying?

HelenOS Status Update

- **Since last FOSDEM**
  - February 1st 2016 – February 3rd 2017
    - Year of the fire monkey:
      February 8th 2016 – January 27th 2017
  - General observations
    - Less activity than in previous years
      - Contributors distracted by other projects
      - Less students (no GSoC, SOCIS, only two theses running)
      - No essential subsystem missing (plateauing)

HelenOS Camp 2016

- **A long tradition of hackatons**
  - Since 2005
    - OK, not so long as OpenBSD's, but an independent idea
- **August 27th – September 3rd 2016**
  - Třemošnice, CZ
  - ~ 3 developers
  - 45 change sets

Year in a Nutshell

- **0.6.1 release**
  - Almost ready (a FOSDEM release?)
- **Improvements of the sun4u (sparc64) support**
  - Jointly with QEMU improvements heading towards supporting HelenOS/sun4u in QEMU
  - Generic serial console in user space
- **Simple installer**
  - Non-interactive, for automated deployments
  - Currently targeting QEMU for ia32, amd64
  - Includes reproducible GRUB build

Year in a Nutshell (2)

- **USB 2.0 support**
  - Part of USB stack rewrite
    - Simplification of various component dependencies
    - Better suitable for further extension (USB 3.0, etc.)
- **IRQ notification handling optimizations**
  - Lower overhead of the processing of IPC message that represent an IRQ
    - Worker fibrils are no longer created and disposed for every single IRQ
- **Dynamic linking improvements**
  - Better support for TLS handling on all platforms
  - Dynamic linking now enabled by default on ia32

Year in a Nutshell (3)

- **Removal of SYS_TLS_SET**
  - Original dedicated syscall for configuring the thread local storage base address (selector) on ia32 and amd64
  - Use of GCC specific `-mno-tls-direct-seg-refs` option
    - The `%gs` and `%fs` segments are no longer used directly as TLS base addresses
    - Generic solution: WRFSBASE / WRGSBASE since Ivy Bridge, emulation on older CPUs and on ia32
- **Continuous maintenance**
  - Many bugs discovered and fixed thanks to verification tools (memory leaks, use-after-free, race conditions)
  - Continuous code refactoring
  - Support for the latest compiler toolchain
  - Detecting a regression in QEMU 2.7.0 on arm32

Year in a Nutshell (4)

- **RISC-V support**
  - Still unfinished (mostly due to lack of focus)
  - Shooting on a moving target
    - Some changes in the draft of the privileged specification and in the Spike reference simulator (a de facto reference platform) are not explicitly tracked and documented
  - HelenOS mainline contains a skeleton code
- **Service manager for HelenOS**
- **User space pagers**

Service Manager for HelenOS

- **Master thesis by Michal Koutný**
  - http://www.helenos.org/doc/theses/mk-thesis.pdf
  - Unification of system-level services (i.e. traditional microkernel servers) and logical services (e.g. network servers)
  - Remotely inspired by SMF, systemd, etc.

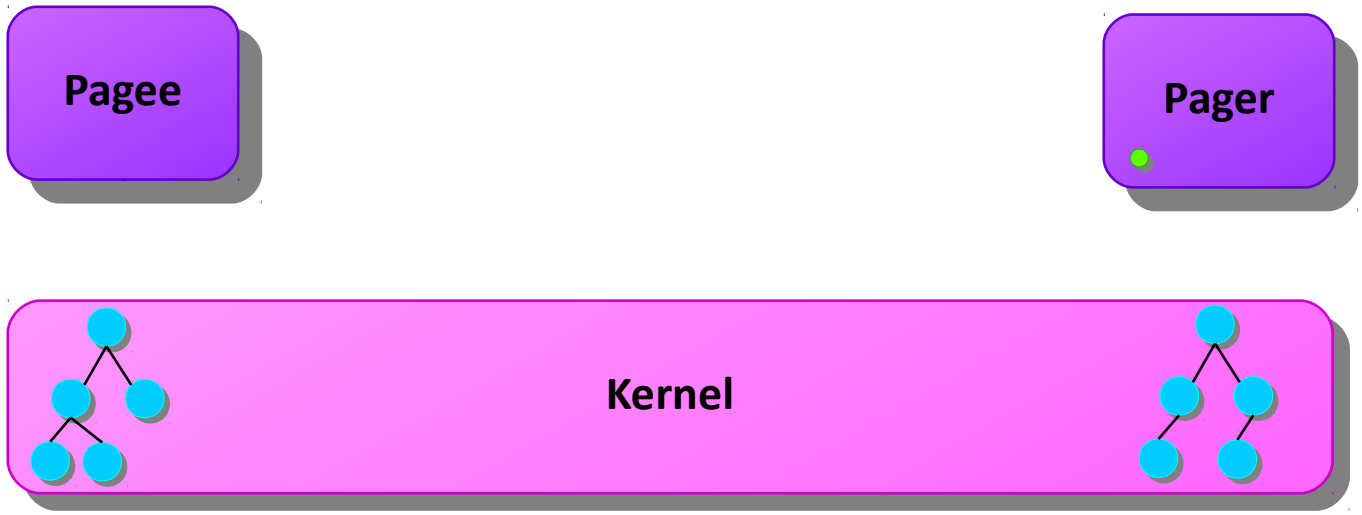
Service Manager for HelenOS (2)

- **sysman service**
  - Unit dependency resolver
    - Starts/stop services according to its dependencies
      - Explicit dependencies (configuration file)
      - Implicit dependencies (IPC broker requests)
    - Unit types: service, mountpoint, configuration, target
- **taskman service**
  - Task creation and monitoring API
  - Manages logical parent/child relations
  - Prospect of restarting failed services

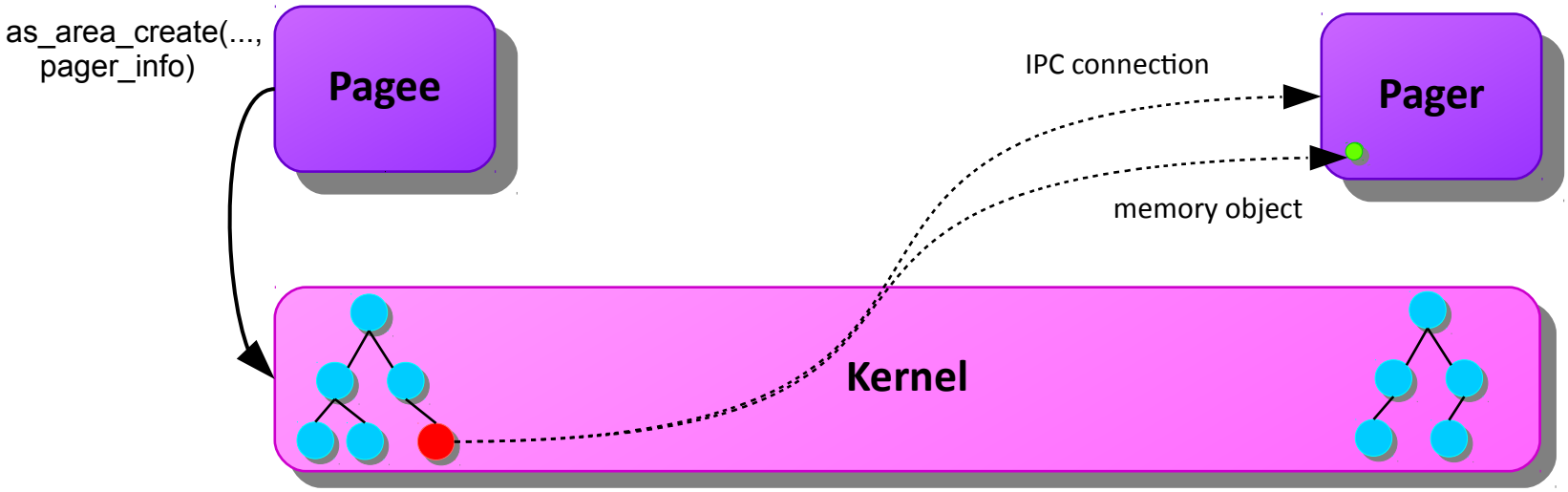
User Space Pagers

- **HelenOS memory management**
  - Traditional kernel architecture (frame allocator, kernel heap allocator, virtual address space areas manager)
    - Mostly mechanisms, but also unavoidable minimal policies (single point of failure)
    - Originally three address space area backends
      - `backend_phys`
      - `backend_anon`
      - `backend_elf`
    - New address space area backend
      - `backend_user`

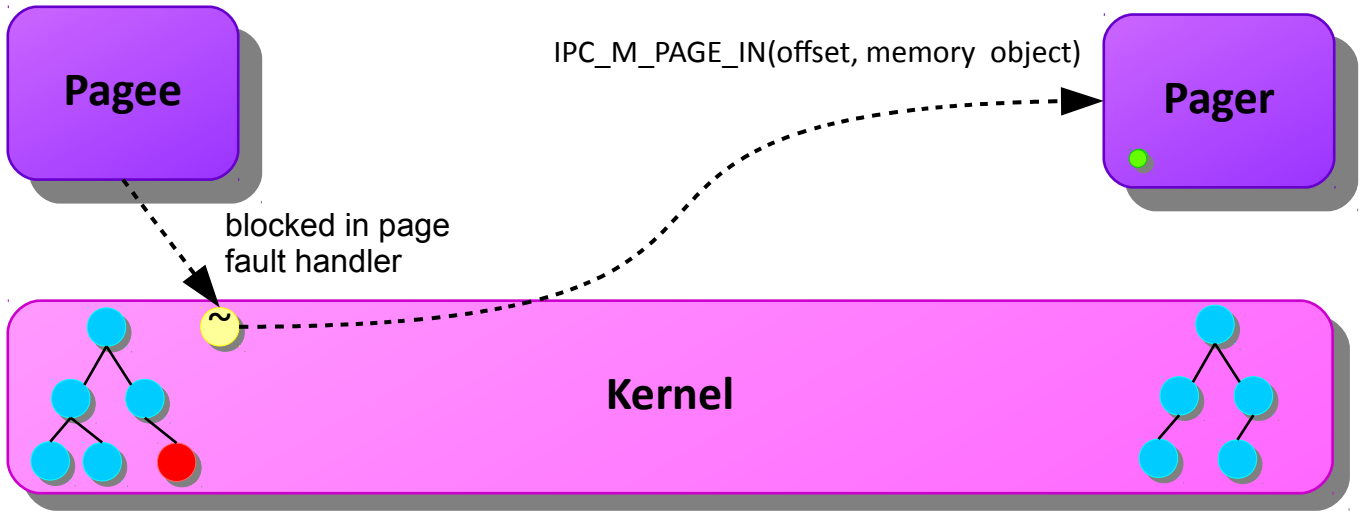
User Space Pagers (2)



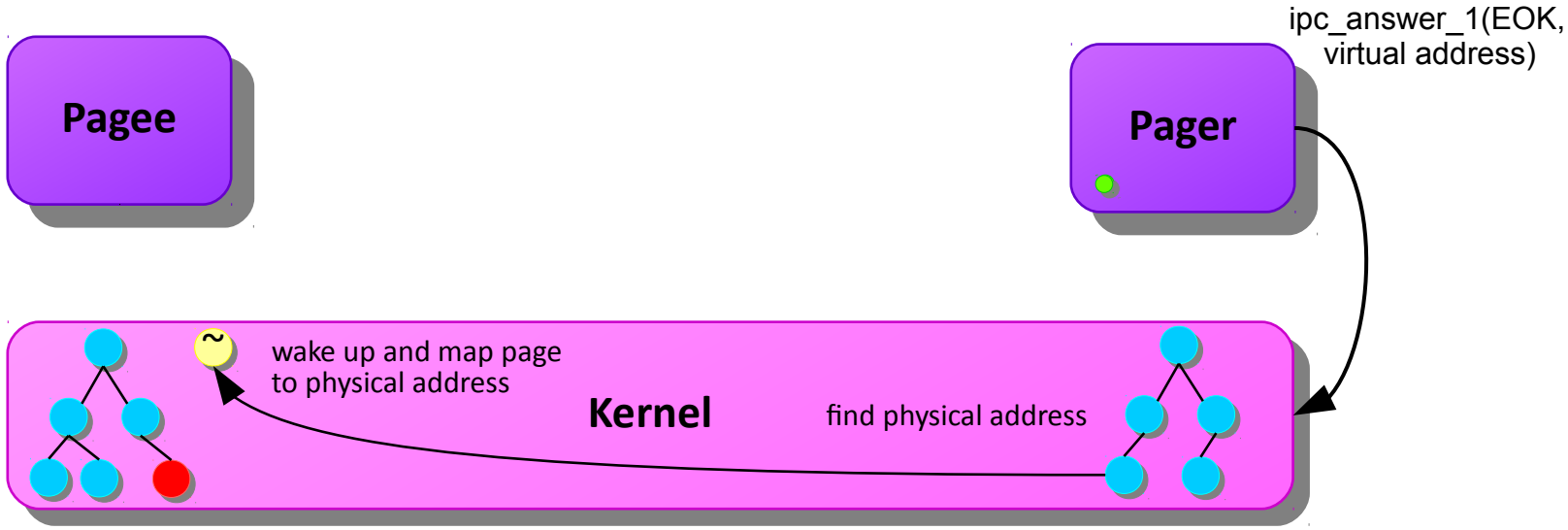
User Space Pagers (2)



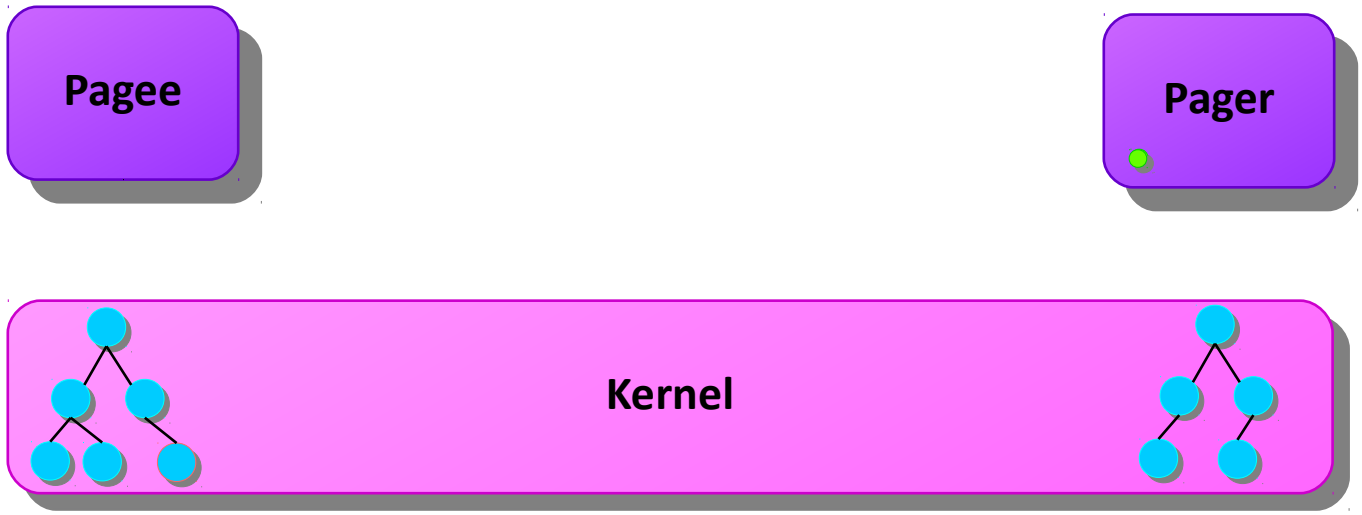
User Space Pagers (2)



User Space Pagers (2)



User Space Pagers (2)



Plans for the Year of the Fire Rooster

HelenOS

- **Finish research papers in the pipeline**
- **Finish the RISC-V port**
- **Support for Turris Omnia**
- **Switch to bi-annual release cycle**
- **More IPC optimizations**
  - Wait-free notifications
  - Adaptive handling of synchronous cases

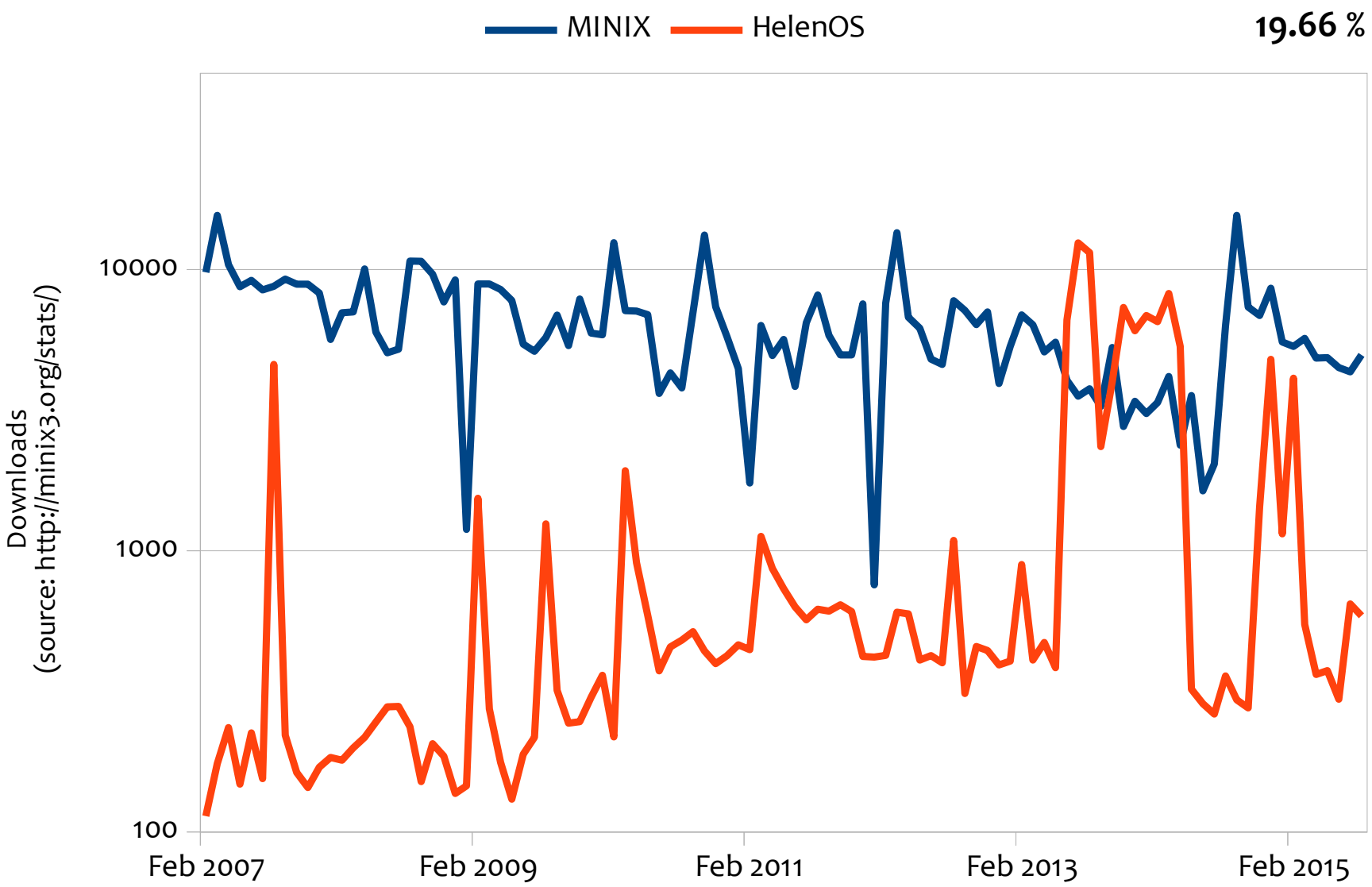
cz.nic

Q&A

www.helenos.org

Backup slides

Downloads


MINIX
HelenOS
19.66 %
Downloads
(source: http://minix3.org/stats/)
10000
1000
100
Feb 2007
Feb 2009
Feb 2011
Feb 2013
Feb 2015