



Aktuelle Trends in der System-Software

Seminar, SS 2025

Michael Schöttner

- Studierende können nach erfolgreichem Abschluss dieses Moduls
 - relevante Inhalte zu ihrem Thema recherchieren und ein praktisches Projekt darin umsetzen
 - ihr Projekt sowie dessen Kontext schriftlich und mündlich erläutern,
 - eine Präsentation planen und durchführen,
 - sinnvolle Fragen stellen, beantworten und konstruktive Kritik annehmen sowie geben

- Schwerpunkt in diesem Semester liegt auf Forschungsarbeiten im Bereich der Betriebssystem-Entwicklung mit der Programmiersprache Rust.

- Seminar für den Master-Studiengang
- Lehrpersonen: Michael Schöttner & Niklas Sombert
- Mittwochs 14:30 – 16:00 Uhr in Raum 25.12.01.51
- 5 ECTS
- Voraussetzungen für die Vergabe von Kreditpunkten
 - Ein eigenes praktisches Projekt (nach Absprache)
 - Schriftliche Ausarbeitung (4-6 Seiten, wahlweise Deutsch/Englisch)
 - Halten eines Vortrags (15-20min., wahlweise Deutsch/Englisch)

■ Lesbarkeit

- Sprache
- Didaktik

■ Inhalt

- Struktur / roter Faden
- Inhaltliches Niveau
- Fachbegriffe eingeführt und verwendet
- Gibt es einen tiefergehenden Schwerpunkt?
- Zusammenfassung

■ Quellen

- Qualität & Quantität

- **Quantität:**
 - Wie viel wurde gemacht?

- **Qualität:**
 - Wie anspruchsvoll war die Umsetzung?
 - Wie gut funktioniert die eigene Implementierung?

■ Vortrag

- Struktur
- Erklärungen
- Präsentationsstil
- Sprache
- Zeit

■ Folien

- Inhaltsangabe
- Inhaltliches Niveau
- Zusammenfassung
- Didaktik
- Gestaltung

■ Fragenbeantwortung

Bewertung: Gewichtung

- 70% Projekt
- 20% Vortrag
- 10% Ausarbeitung

Ablauf

Woche	Datum	Inhalt	Frist
1	09.04.25	- Entfall -	
2	16.04.25	Organisatorisches und Themenvorstellung	
3	23.04.25	"Ablauf: wissenschaftliche Publikationen" - Abstimmung wegen Themenauswahl	Themenwahl
4	30.04.25	"Schreiben der Ausarbeitung" und - Abstimmung zu den Projektvorhaben	Ideen zum geplanten Projekt
5	07.05.25	"Vorbereiten des Vortrags"	Stichpunkte zum geplanten Projekt
6	14.05.25	Coding vor Ort & Austausch	
7	21.05.25	Coding vor Ort & Austausch	
8	28.05.25	Coding vor Ort & Austausch	
9	04.06.25	Coding vor Ort & Austausch	
10	11.06.25	Coding vor Ort & Austausch	
11	18.06.25	Coding vor Ort & Austausch	
12	25.06.25	Coding vor Ort & Austausch	
13	02.07.25	Coding vor Ort & Austausch	
14	09.07.25	Vorträge 4x90min	
15	16.07.25	Vorträge 4x90min	
16	23.07.25	-	Abgabe der Slides & vorläufigen Projekte
	31.08.25		Abgabe der Ausarbeitung & finalen Projekte

Thema 1: Projekt im eigenen Rust-OS

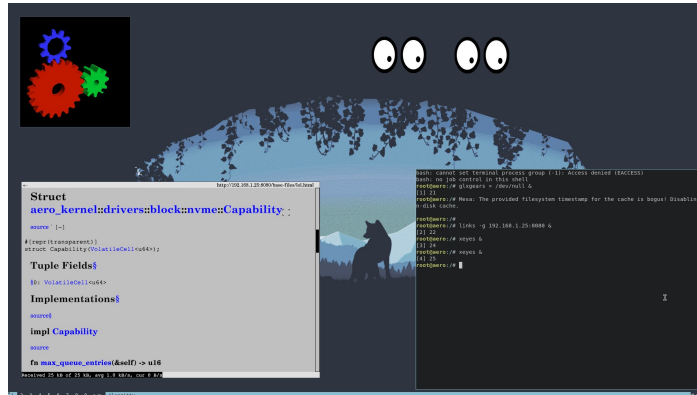
- Richtet sich an alle Personen die mindestens einen unserer Kurse “Betriebssystem-Entwicklung” und “Isolation und Schutz in Betriebssystemen” besucht haben

Thema 2: D3OS

- A new distributed operating system for data centers, developed by the operating systems group of the department of computer science at Heinrich Heine University Düsseldorf
- <https://github.com/hhu-bsinfo/D3OS>



- Aero is a new modern, experimental, unix-like operating system written in Rust. Aero follows the monolithic kernel design and it is inspired by the Linux Kernel. Aero supports modern PC features such as Long Mode, 5-level paging, and SMP (multicore), to name a few.
- <https://github.com/Andy-Python-Programmer/aero>



Thema 4: Moros

- MOROS is a hobby operating system written in Rust by Vincent Ollivier.
- It targets computers with a x86-64 architecture and a BIOS, so mostly from 2005 to 2020, but it also runs well on most emulators (Bochs, QEMU, and VirtualBox).
- <https://github.com/vinc/moros>

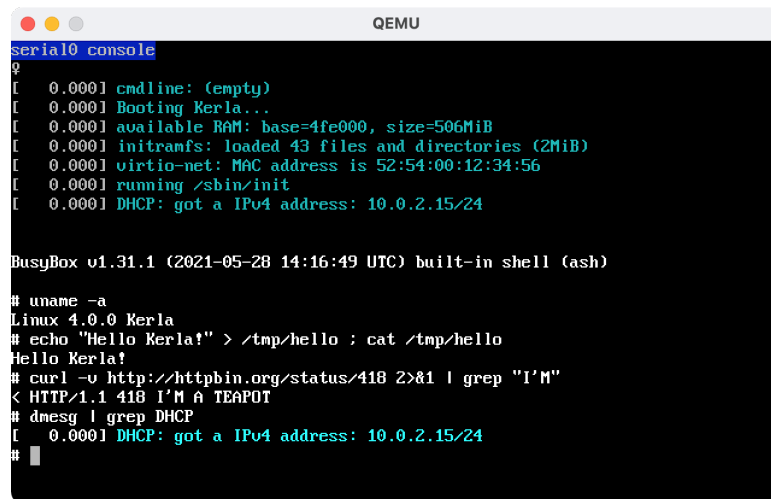
```
[0.410937] MFS Superblock found in ATA 0:0
[0.415937] RTC 2024-12-21 10:03:57 +0000

      (\_/)
      (o o)
-----oo0--(_)--0oo-----
+-----+
| .100 110. .1100. 111110. .1001. .01000. |
| 00'1001'11 .11 01. 00 '00 .10 00. 10' 11 |
| 01 00 10 10 00 001101' 01 00 '100. |
| 10 01 10 01 11 01'00 01 11 '100. |
| 00 01 01 '00 11' 10 '11. '00 11' 01 00 |
| 11 10 10 '1001' 00 01 '0110' '01101' |
|                                     |
|      MOROS: Obscure Rust Operating System      |
|                                     |
|      (v0.11.0)      |
|                                     |
+-----+

Username: vinc
Password:

~
> _
```

- Kerla is a monolithic operating system kernel written from scratch in Rust which aims to be compatible with the Linux ABI, that is, it runs Linux binaries without any modifications.
- <https://github.com/nuta/kerla>



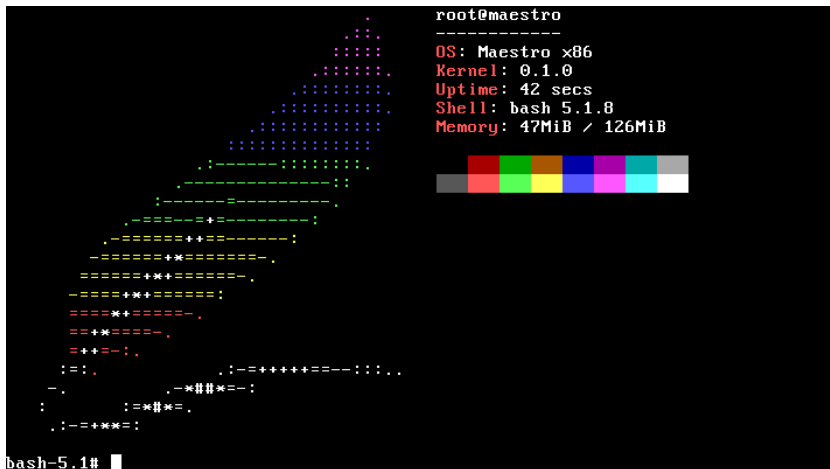
```
QEMU
serial0 console
[ 0.000] cmdline: (empty)
[ 0.000] Booting Kerla...
[ 0.000] available RAM: base=4fe000, size=506MiB
[ 0.000] initramfs: loaded 43 files and directories (2MiB)
[ 0.000] virtio-net: MAC address is 52:54:00:12:34:56
[ 0.000] running /sbin/init
[ 0.000] DHCP: got a IPv4 address: 10.0.2.15/24

BusyBox v1.31.1 (2021-05-28 14:16:49 UTC) built-in shell (ash)
# uname -a
Linux 4.0.0 Kerla
# echo "Hello Kerla!" > /tmp/hello ; cat /tmp/hello
Hello Kerla!
# curl -v http://httpbin.org/status/418 2>&1 | grep "I'M"
< HTTP/1.1 418 I'M A TEAPOT
# dmesg | grep DHCP
[ 0.000] DHCP: got a IPv4 address: 10.0.2.15/24
#
```

- An embedded operating system designed for running multiple concurrent, mutually distrustful applications on low-memory and low-power microcontrollers.
- <https://tockos.org/>
- <https://github.com/tock/tock>

- Theseus is a new OS written from scratch in Rust to experiment with novel OS structure, better state management, and how to leverage intralingual design principles to shift OS responsibilities like resource management into the compiler.
- <https://github.com/theseus-os/Theseus>

- <https://github.com/maestro-os/maestro>



- rxv64 is a pedagogical operating system written in Rust that targets multiprocessor x86_64 machines. It is a reimplementation of the xv6 operating system from MIT.
- As a pedagogical system, it supports very little hardware other than the text-mode CGA device, serial port, PS/2 keyboard controller, and PCIe AHCI SATA storage devices.
- <https://github.com/dancrossnyc/rxv64>

Thema 10: Octox

- Octox is a Unix-like operating system inspired by xv6-riscv.
- Octox loosely follows the structure and style of xv6, but is implemented in pure Rust.
- <https://github.com/o8vm/octox>

```
ls          File      4 1333200
init        File      5 1300632
rm          File      6 1259400
echo        File      7 1259864
initcode    File      8 1254496
sh          File      9 1356232
grep        File     10 1339424
head        File     11 1312080
kill        File     12 1272784
wc          File     13 1271968
mkdir       File     14 1264800
cat         File     15 1263448
console     Device    16 0
null        Device    17 0
$ echo Hello World
Hello World
$ cat README.org | head | grep octox
#+title: octox
octox is a Unix-like operating system inspired by xv6-riscv.
octox loosely follows the structure and style of xv6, but is implement
ed in pure Rust.
$
```

- This is an experiment in writing an OS Kernel in rust (<http://rust-lang.org>).
- Mostly the architecture is being designed as I go along, but it will be written to be architecture independent (the current version is x86_64/amd64).
- https://github.com/thepowersgang/rust_os