

Practical course

Systems Programming

(SoSe 2025)

Preliminary meeting

Systems Research Group

<https://dse.in.tum.de/>

IMPORTANT

This course is not offered to *IN2397* (CSE course)

About us

Systems Research Group

Prof. Pramod Bhatotia <https://dse.in.tum.de/>

| | | |
|---------------------|--|----------------------------|
| Anatole Lefort | | anatole.lefort@tum.de |
| David Schall | | david.schall@tum.de |
| Theofilos Augoustis | | theofilos.augoustis@tum.de |
| Jiyang Chen | | jiyang.chen@tum.de |
| Francisco Romão | | francisco.romao@tum.de |

Systems programming applications

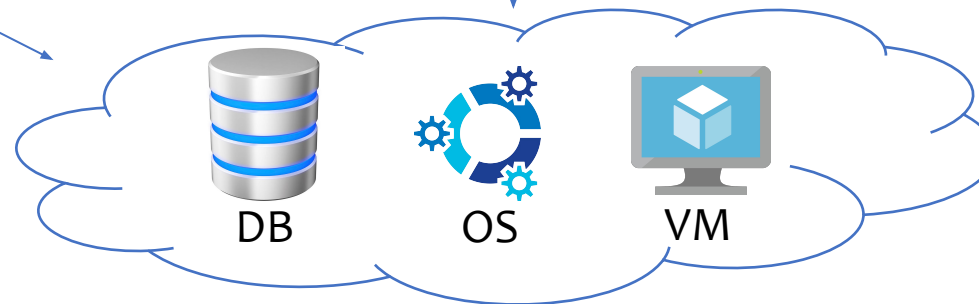
Internet of things



Transportation



Healthcare



Low level systems programming is an essential building block for high level applications

Software core properties

Performance



Reliability



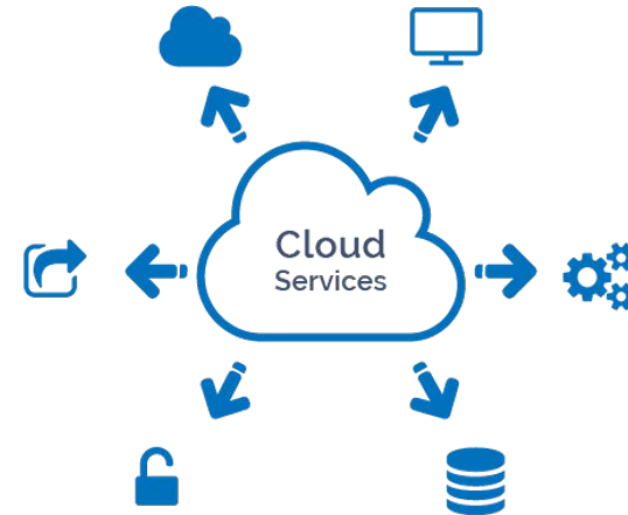
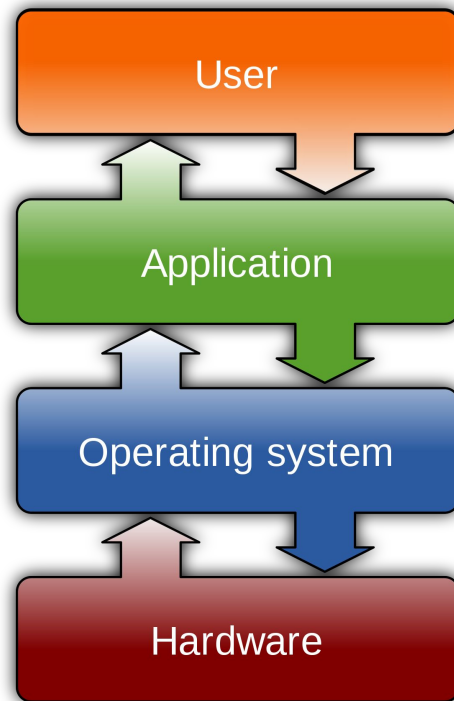
Security



Efficient low level systems programming is critical to ensure these properties

System stack

Systems programming spans in multiple system levels and application domains

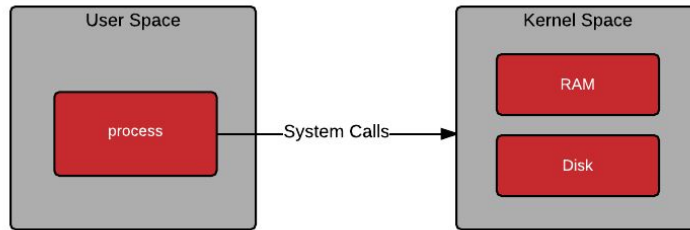


Time to get hands-on experience!

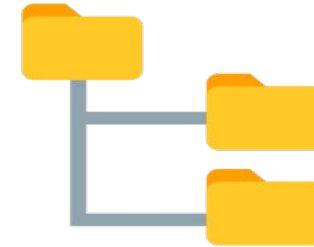
Course topics

- This course covers some of the most important aspects of systems programming:

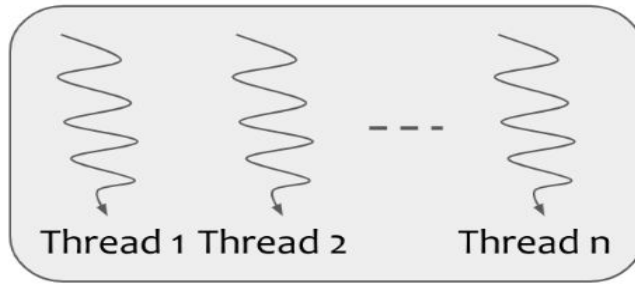
Kernel and system calls



File I/O



Concurrency and synchronization



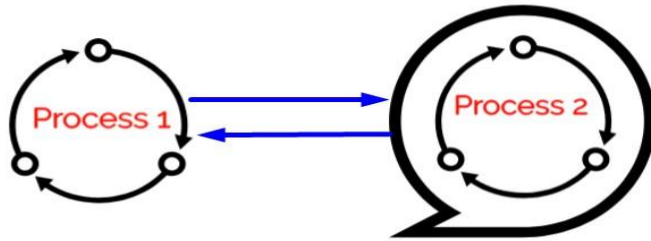
Memory management



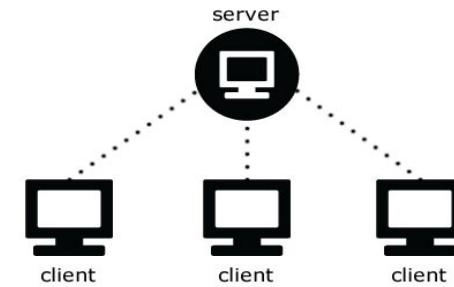
Course topics

- This course covers some of the most important aspects of systems programming:

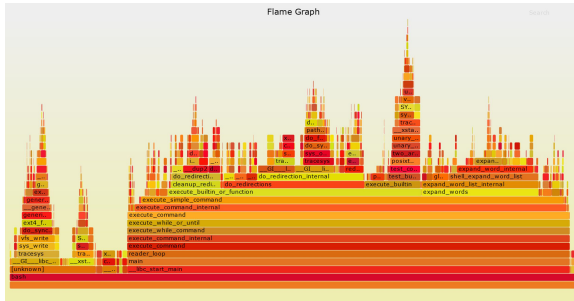
Processes



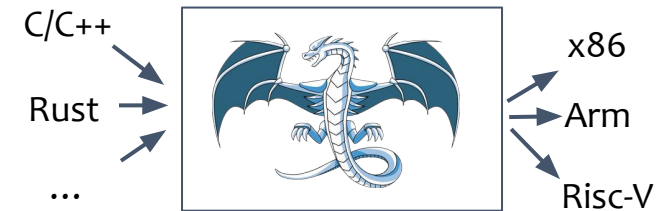
Networking



Performance profiling



Compilers / LLVM



Lab format

- Lab assignments
 - **8 practical programming exercises**
 - Deadline of 1 - 2 weeks depending on the difficulty/workload
 - Online submission
- Weekly meeting
 - Video with theoretical background coverage
 - Question and answer session to explain and discuss each assignment
 - Slack channel for questions
- Assessment:
 - **8 programming assignments (100%)** with public & private unit tests
 - No further exam / quiz / projects

Assignment examples (Actual tasks may change)

- Implement some functionalities of a filesystem
- Conversion of a single-thread program to a multithreaded version using locks
- Write your own memory allocator for better performance
- Implement your own client/server applications
- Use LLVM to write simple compiler passes

Grading system

- Github classroom (<https://classroom.github.com/>)
 - Template repository for each task with detailed instructions & test cases
- Automated tests
 - Points are distributed among the exercises based on the estimated workload (30-60 points)
 - Specially designed test cases with gradually increasing difficulty
 - Hidden tests to detect & prevent gaming the grading system
- Grading scheme :

| From | To | Grade | From | To | Grade |
|------|-----|-------|------|-----|-------|
| 0 | 100 | 5.0 | 194 | 205 | 2.7 |
| 101 | 112 | 4.7 | 206 | 217 | 2.3 |
| 113 | 124 | 4.3 | 218 | 229 | 2.0 |
| 125 | 148 | 4.0 | 230 | 238 | 1.7 |
| 149 | 163 | 3.7 | 239 | 245 | 1.3 |
| 164 | 178 | 3.3 | 246 | 250 | 1.0 |
| 179 | 193 | 3.0 | | | |

Languages / OS

- Languages
 - Choice between **C, C++ and Rust**
 - Limited choice of allowed libraries (different per language)
 - Can be switched for each task
 - **Some tasks require a specific language**
- OS Environment information
 - All executables must run on **Linux, x86_64**
 - Use virtual machines if you run a different OS (i.e. Hyper-V on Windows)
 - You can also use the remote desktop (lxe.halle.in.tum.de)

Learning goals

- Acquire fundamental knowledge to build robust systems
- Familiarize yourself with end-to-end system design
- Learn techniques for profiling, debugging and optimization of low-level code
- Get a good understanding of memory- and resource management
- Improve hands-on experience through a variety of programming tasks

- Importantly, have fun!

Prerequisites

- We don't have any compulsory pre-requisites, but we prefer
 - **Knowledge equivalent to the lectures**
 - Fundamentals of Programming (IN0002)
 - Introduction to Computer Architecture (IN0004)
 - Basic Principles: Operating Systems and System Software (IN0009)
 - **Programming knowledge**
 - Foundations of programming (C, C++ or Rust)
 - Work in a Linux environment
 - **Knowledge on Git and GitHub**

If the prerequisites are unclear/strict -- please check with the instructor!

Code of conduct

- University plagiarism policy
 - <https://www.in.tum.de/en/current-students/administrative-matters/student-code-of-conduct/>
- Decorum
 - Promote freedom of thoughts and open exchange of ideas
 - Cultivate dignity, understanding and mutual respect, and embrace diversity
 - Racism and bullying will not be tolerated
- **Please write your own code!**

Interested?



Matching platform

Welcome to the Matching platform matching.in.tum.de/

Dear students,

we changed the name of the course "Seminar: Recent advances in Computer Systems", for consistency reasons. The new name are "Seminar: Hot Topics in Computer Systems", now.

Login with your TUM identifier.

➔ TUM login

Login for exchange students
(without TUM identifier)

➔ Exchange student login

Any questions? Visit the FAQs!

📘 FAQs

Sign up on the TUM matching platform

Contacts

- Anatole Lefort
 - anatole.lefort@tum.de
- David Schall
 - david.schall@tum.de

We *strongly* prefer slack for all communications. Please join the slack channel.



Workspace: <http://ls1-courses-tum.slack.com/>

Website: <https://github.com/ls1-sys-prog-course/docs>

Channel: #ws-25-sys-prog

Join us with TUM email address (@tum.de)

IMPORTANT

This course is not offered to IN2397 (CSE course)