O
O S E
E

Concept

Subject Matter

introduction .... a first problem analysis

fundamentals .... domain analysis, program families, and object orientation

case study .... design and development of TAL, a *threads abstraction layer*

discussion .... things nice to have and not to have

revision .... aspect orientation

conclusion .... lessons learned

Educational Objectives

- an excursus on modern operating-system design *and* implementation
  - focusing on system-software flexibility, portability, and scalability
- the use of software-engineering techniques in system-software design
  - feature modeling [3]
  - program families [5]
  - object orientation [8]
  - aspect-oriented programming [4]
- an inauguration into the secrets and a rationale of PURE [1]

Prerequisites

- structured computer organization, **operating systems**
- C/C++, assembler
- enjoy system-level programming
- no fear of stuff hard to digest
- some sort of **staying power**

Syllabus

- one **lecture** per week, two hours each .... 2 SWS
  - subject presentation
- one **seminar** per week, two hours each .... 2 SWS
  - subject consolidation
  - practice discussion
- computer **practice** N hours per week, $0 < N \leq 164$ .... 0 SWS

Achievement Control

- **practice** or
  - *pass* .... in case of successful elaboration of all exercises
  - *consultation* .... in case of unsuccessful elaboration of one exercise
  - *fail* .... otherwise, or enjoy . . .
- **examination** on lecture *and* seminar stuff

Academic Staff

- Wolfgang Schröder-Preikschat .... professor
  - `http://www4.informatik.uni-erlangen.de/~wosch`

Suggested Reading

[1] D. Beuche, A. Guerrouat, H. Papajewski, W. Schröder-Preikschat, O. Spinczyk, and U. Spinczyk. The PURE Family of Object-Oriented Operating Systems for Deeply Embedded Systems. In *Proceedings of the 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'99)*, St Malo, France, May 1999.

[2] J. O. Coplien. *Multi-Paradigm Design for C++*. Addison-Wesley, 1999. ISBN 0-201-82467-1.

[3] K. Czarnecki and U. W. Eisenecker. *Generative Programming—Methods, Tools, and Applications*. Addison-Wesley, 2000. ISBN 0-201-30977-7.

[4] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J.-M. Loingtier, and J. Irwin. Aspect-Oriented Programming. In M. Aksit and S. Matsuoka, editors, *Proceedings of the 11th European Conference on Object-Oriented Programming (ECOOP '97)*, volume 1241 of *Lecture Notes in Computer Science*, pages 220–242. Springer-Verlag, June 1997.

[5] D. L. Parnas. Designing Software for Ease of Extension and Contraction. *IEEE Transactions on Software Engineering*, SE-5(2):128–138, 1979.

[6] W. Schröder-Preikschat. *The Logical Design of Parallel Operating Systems*. Prentice Hall International, 1994. ISBN 0-13-183369-3.

[7] W. Schröder-Preikschat. Operating-System Engineering. http://www4.informatik.uni-erlangen.de, 2002.

[8] P. Wegner. Classification in Object-Oriented Systems. *ACM, SIGPLAN Notices*, 21(10):173–182, 1986.