Literatur zu Software-Engineering

Stand: Wintersemester 2017/2018

Dr.-Ing. Clemens Reichmann

Clemens.Reichmann@vector.com

Institut für Technik der Informationsverarbeitung
Fakultät für Elektrotechnik & Informationstechnik
Karlsruher Institut für Technologie (KIT)

Literaturempfehlungen

Übergreifende Literatur für Software-Engineering

- **Software Engineering.** (Ian Sommerville, 9. Auflage, *Addison-Wesley*, 2012
- **Lehrbuch der Softwaretechnik.**
  Teil 1: Software-Entwicklung
  Teil 2: Software-Management, Qualitätssicherung, Unternehmensmodellierung.
  (Helmut Balzert, 2. Auflage, *Spektrum Akademischer Verlag*, Feb. 2008.)
- **Designing Object-Oriented Software.** (Wirfs-Brock, Wiklerson, Wiener, Prent. Hall, 1990)
- **Object-Oriented Analysis and Design with Applications.** (Grady Booch, Addison-Wesley Professional, 1993)
- **Das UML-Benutzerhandbuch.** (Ivar Jacobson et. al., Addison-Wesley)
  (ISBN: 3827314860)

Entwurfsmuster

Nachschlagewerke zu Entwurfsmustern (Kapitel 6)

Bücher

- **The Design Patterns.** (James W. Cooper, Addison-Wesley)
- **Entwurfsmuster.** (Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides; Addison-Wesley, 1.Auflage 1996) (**ISBN: 3-8273-1862-9)**
  *Anmerkung: Auch bekannt als „**GoF Buch**“*
- **Objektorientierte Softwaretechnik mit UML, Entwurfsmustern und Java** (B. Brügge, A.H. Dutoit; Pearson 2004)
- **Patternorientierte Softwarearchitektur** (Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerland, Michael Stal; Addison-Wesley 1998)

Internet

- http://www.cmcrossroads.com/bradapp/docs/patterns-nutshell.html
- http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/patterns
- http://www.javaworld.com/javaworld/jw-05-2003/jw-0530-designpatterns.html
- http://java.sun.com/blueprints/patterns

Refactoring

Nachschlagewerke zu Refactoring (Kapitel 8)

Bücher

- **Refactoring. Wie Sie das Design vorhandener Software verbessern**. (Martin Fowler; Addison-Wesley Verlag) **(ISBN 3-8273-1630-8)**
- **Refactoring Workbook** (William C. Wake; Addison-Wesley) **(ISBN 0-321-10929-5)**
- **Refactoring To Patterns** (Joshua Kerievsky; Addison-Wesley) (ISBN 0-321-21335-1)

Entwurfsmuster

Nachschlagewerke zu Wiederverwendung (Kapitel 9)

Bücher

- **Reuse-based Software Engineering.** (H. Mili, A. Mili, S. Yacoub, E. Addy; John Wiley & Sons, Inc. 2002)
- **Software Reuse: Architecture, Process and Organization for Business Success.** (I. Jacobson, M. Griss, P. Jonsson; ACM Press 1997)
- **Software Engineering with Reusable Components.** (J. Sametinger; Springer-Verlag 1997) **(ISBN 3-540-62695-6)**
- **Software Reuse: A Holistic Approach**. (E. Karlsson; John Wiley and Sons Ltd, 1995)
- **Software Reusability.** (W. Schaefer, R. Prieto-Diaz, M. Matsumoto; Ellis Horwood, 1994)
- **Software Reusability.**
  Volume I: Concepts and Models. (ISBN-10: 0201080176)
  Volume II: Applications and Experience. (ISBN-10: 0201500183)
  (T.J. Biggerstaff, A.J. Perlis; ACM Press, 1989)

Weitere Literatur

Spezifische Themen des SE zur Vertiefung:

- **Erfolgreich im Team**, Christoph V. Hang, dtv Verlag, 1998
- **Projekte zum Erfolg führen**, Heinz Schelle, dtv Verlag, 1999

- **The Push to Make Software Engineering Respectable**. Describes the essentials of professionalism for software engineers. (G. Pour, M. L. Griss and M. Lutz, IEEE Computer, 33 (5), May 2000)

- **Special Issue on Software Process Diversity**. This special issue includes a number of interesting and useful articles on different types of process for developing software. It also includes articles covering process maturity and the CMM model (Chapter 25). (IEEE Software, 17 (4), July 2000)

- **eXtreme programming - Development through Dialog**. A short introduction that addresses common concerns about extreme programming. (R. C. Martin, IEEE Software, July 2000)

- **A Software Development Process for Small Projects**. Suggests a software process that has been adapted for small companies. (M. L. Russ and J. D. McGregor, IEEE Software, 17 (4), September 2000)

- **Wisdom: A Software Engineering Method for Small Software Development Companies**. A discussion of a 'lightweight' method that brings some elements of systematic software engineering to small organisations who cannot afford complex methods. (N. J. Nunes and J. F. Cunha, IEEE Software, 17 (4), September 2000)

- **Weaving Together Requirements and Architectures**. A short discussion of how requirements engineering and architectural design can be integrated in an evolutionary development process. (B. Nuseibeh, IEEE Computer, March 2001)

- **Get Ready for Agile Methods, with Care**. A good discussion of the pros and cons of agile methods such as extreme programming by a leading software engineering practitioner and researcher. (B. Boehm, IEEE Computer, 35 (4), January 2002)

- **Software System Engineering: A Tutorial**. There are few articles published on this important topic. This is a good overview and introduction. (R. H. Thayer, IEEE Computer, 35 (4), April 2002)

- **The Agile Methods Fray**. A follow-up to Boehm's January article presented as a discussion between Boehm and DeMarco. (T. DeMarco and B. Boehm, IEEE Computer, 35 (6), June 2002)

- **A Reference Model for Requirements and Specifications** (C. A. Gunter, E. L. Gunter, M. Jackson and P. Zave, IEEE Software, 17 (2), March 2000) Requirements Management: The search for Nirvana (D. J. Reifer, IEEE Software, 17 (3), May 2000)

- **Requirements that Handle IKIWISI, COTS and Rapid Change**. An interesting short article on how requirements engineering has to evolve to cope with different types of software development. (B. Boehm, IEEE Computer, 33 (7), July 2000)

- **Developing Groupware for Requirements Negotiation**: Lessons Learned. A description of a support systems based around Boehm's win-win approach for teams who are negotating system requirements (B. Boehm, P. Grunbacher, R. O. Briggs, IEEE Software, 18 (3), May/June 2001)

- **Where do Requirements Come From**. A short description of a new approach to requirements discovery using stimuli to creative thinking. (N. Maiden and A. Gizikis, IEEE Software, 18 (5), September/October 2001)

- **Correctness by Construction: Developing a Commercial Secure System**. Discusses the use of formal specification in developing a secure system (A. Hall and R. Chapman, IEEE Software, 19 (1), January/February 2002)

- **What do you mean by COTS**. The use of COTS systems, discussed in Chapter 14, is likely to be increasingly important. This is an interesting clarification of the sometimes-confusing terminology in this area. (D. Carney and F. Long, IEEE Software, 17 (2), March 2000) Redesigning Air Traffic Control: An exercise in software design (D. Jackson and J. Chapin, IEEE Software, 17 (3), May 2000)

- **Real-time Distributed Object Computing**: An Emerging Field. A special section discussing the use of distributed objects in real-time systems (E. Shokri and P. Sheu, IEEE Computer, 33 (6), June 2000)

- **Diversity in Reuse Processes**. Describes four different successful approaches to achieve software reuse. (M. Morisio, C. Tully and M. Ezran, IEEE Software, 17 (4), July 2000)

- **Component-based Enterprise Frameworks**. A special issue of the CACM with a range of papers on frameworks. (Comm ACM, 43 (10), October 2000)

- **Tracking Usability Issues**: To Bug or Not to Bug. A discussion of how usability problems that arise in a system can be tracked and integrated with other problem reporting systems. There are conflicting viewpoints on whether these issues should be classed as system bugs (C. Wilson and K. Coyne, ACM Interactions, May/June 2001, 8 (3))

- **Accelerating Development with Agent Components**. One of a number of articles on very rapid software development (software engineering on internet time) which proposes the use of agents for more rpaid delivery. The problems of verification and validation of agent-based systems are not covered (and these are big problems). (M. Griss and G. Pour, IEEE Computer, May 2001, 34 (5))

- **COTS-based Systems Top 10 List**. A list of hypotheses about COTS-based systems which highlights why this approach may not be as cheap nor as effective as enthusiasts suggest (V. R. Basili and B. Boehm. IEEE Computer, 34 (5), May 2001)

- **Middleware Challenges Ahead**. A good summary of the features that next-generation middleware will have to provide to support new types of distributed application such as ubiquitous computer systems. (K. Geihs, IEEE Computer, June 2001, 34 (6))

- **Comm. ACM. Special Issue on Aspect-oriented Programming**. This special issue includes an introduction to aspect-oriented programming - an approach to program design that may have a major impact on future software development (Comm. ACM, 44 (10), October 2001)

- **Will UML 2.0 be agile or awkward**. A plea for the revision of UML expected in 2002 to be lean rather than flabby. (COmm. ACM, 45 (1), January 2002)

- **The Case for Reflective Middleware**. Discusses a new approaches to the construction of middleware for distributed systems. (F. Kon, F. Costa, G. Blair and R. H. Campbell. Comm. ACM, 45 (6), June 2002)

- **Can you Trust Software Capability Evaluations**. Discusses practical problems with using the CMM model to assess contractor capability. (E. O' Connell and H. Saiedian, IEEE Computer, 33 (2), February 2000) Applying CMM Project Planning practices to Diverse Environments. Discusses how the Capability Maturity Model can be adapted to cope with development environments that are different from the large, long-lifetime project developments for which the model was designed. (D. L. Johnson and J. G. Brodman, IEEE Software, 17 (4), July 2000)

- **How Internet Companies Negotiate Software Quality**. An article discussing the difficulties of applying conventional approaches to software quality management when there are very severe schedule pressures and a need for rapid delivery. (R. Baskerville et al., IEEE Computer, May 2001, 34 (5))
  There have been major revisions of capability maturity models discussed in Chapter 25 with a new integrated model suite published by the Software Engineeirng Institute. Full information can be accessed at http://www.sei.cmu.edu/cmmi/ but I hope to find something more accessible soon.

- **Extreme Collaboration**. Discusses an approach used at NASA's Jet Propulsion Laboratory for rapid aerospace design. This involves a design team working together in a large room with lots of public displays. (G. Mark. Comm. ACM, 45 (6), June 2002)