Prof. Bernhard Steffen
TU Dortmund

BIO

Bernhard Steffen graduated in Mathematics (1983) and obtained a PhD in Computer Science (1987) from the Christian-Albrechts Universität Kiel (D), then he was Research Fellow at the Laboratory for Foundations of Computer Science (LFCS) in Edinburgh and Researcher at the University of Aarhus (Denmark). 1990 he became Associate Professor for Distributed Systems at RWTH Aachen, and 1993 Full Professor for Programming Systems at the University of Passau. Since 1997 he holds the Chair of Programming Systems and Compiler Construction at the University of Dortmund, where he was Dean of Computer Science between 2002 and 2006 and member of the Senate 2006/07.

He is author of over 200 internationally refereed papers concerning various aspects of formal (verification) methods and tools for program analysis, compiler optimization, model generation, testing, and service-oriented software development (h-number 46). He has served on more than 10 0 Program Committees, over 15 times as chair, on numerous Steering Committees, and in the Editorial Boards of the *ACM Transactions on Programming Languages and Systems* (TOPLAS), Kluwer´s *Formal Methods in System Design*, as well as Springer´*s Software: Concepts and Tools* and *Innovations in Software and Systems engineering: A NASA Journal*. Since 2004 he is editor of LNCS (Lecture Notes in Computer Science) for the sub-libraries *`Theoretical Computer Science´*, *`Programming Techniques and Software Engineering´* and *`Advanced Research in Computing and Software Science´*,

He has broad experience in the use of formal methods to support state of the art industrial software development of distributed cooperative systems through major academic and industrial projects (where he won the European IT Award in 1996 together with Siemens-Nixdorf, and a start-up competition in 2001) and consulting (he supported e.g., BASF, IKEA, Siemens, Deutsche Telekom, ThyssenKrupp, and was founder and member of the Advisory Board of various start-up companies), as well as through his activity as an Advisory Board Member of ASTEC, a Swedish technology transfer initiative for *Advanced Software TEChnology*, and as a member of the International Scientific Advisory Board of the UK strategic research & training initiative in Large-Scale Complex IT Systems (LSCITS).

He is founder and Editor in Chief of *Software Tools for Technology Transfer* (STTT), Springer Verlag, and co-founder and Steering Committee Member of TACAS, the Int. Conference on Tools and Algorithms for the Construction and Analysis of Systems. In 2004 he co-founded ISoLA (Int. Symposium on Leveraging Applications of Formal Methods, Verification and Validation).

In 1989 he co-developed the Concurrency Workbench, one of the earliest formal analysis tools for distributed and parallel systems, 1991 he set the scene for Software Model Checking with his paper *Data Flow Analysis as Model Checking*, 1992 he presented the first functioning Model Checker for infinite-state systems with *Model Checking for Context-Free Processes*, and in 2002 he obtained the *Most Influential PLDI Paper Award* for *Lazy Code Motion*, which is given 10 years later in retrospective.

Currently, his research focuses on technology to continuously support business process modelling and service-oriented development of complex, heterogeneous, evolving systems. Conceptual backbone for the envisaged Continuous Model Driven Engineering (CMDE) is the eXtreme Model-Driven Development (XMDD) paradigm, which aims at continuously involving the customer/application expert throughout the whole systems' life cycle including software maintenance and evolution. Characteristic for XMDD, which combines service and aspect orientation, model driven design and ideas from eXtreme programming in order to achieve scalable Scrum-like agility, is its comprehensive accompanying model. This model, which serves all the involved parties via dedicated views as a unique reference led to the term *One*

Thing Approach (OTA). Continuous quality assurance is guaranteed via model-checking-based verification and the combination of model-based testing and automata learning. This combination has, in particular, proved to be very effective for enterprise-scale applications, where the test models required for model-based testing are not available.