CWI Soirée

een bijzondere avond
met voordrachten door

Bjarne Stroustrup
(AT&T Labs)
en
Edmund M. Clarke
(Carnegie Mellon University)

woensdag 24 maart 1999
19.15 - 22.00 uur

Koninklijk Instituut voor de Tropen
Mauritskade 63, Amsterdam

PROGRAMMA

19.15

ontvangst

19.45

welkomstwoord

G. van Oortmerssen, directeur CWI

19.50

introductie van de sprekers

J.W. de Bakker, CWI

20.00

Bjarne Stroustrup

What is an object and what isn't?

21.00

Edmund M. Clarke

Symbolic model checking without BDDs

22.00

receptie

BJARNE STROUSTRUP

Bjarne Stroustrup was born in Denmark and studied mathematics and computer science at the University of Aarhus. He received a Ph.D. degree in Computer Science from Cambridge University (UK) in 1979. Immediately afterwards he joined the Computer Science Research Center of Bell Telephone Laboratories. He now heads the Large-scale Programming Research department of AT&T Labs. His research interests include distributed systems, operating systems, simulation, design, and programming. Stroustrup is best known for his design and implementation of C++, now the most widely used language supporting object-oriented programming. He then used C++ as a tool for applying object-oriented and generic programming techniques to areas like general systems programming, switching, simulation, graphics, user interfaces, embedded systems, and scientific computation. His five books include „The C++ Programming Language" (Addison-Wesley 1985, 1991, and 1997), translated into eleven languages. He also took an active role in the creation of the ANSI/ISO standard for C++. Stroustrup is an AT&T Bell Laboratories Fellow and an AT&T Fellow. He received the 1993 ACM Grace Murray Hopper Award for his early work on C++ and is an ACM fellow. BYTE magazine named him in 1995 as one of „the twenty most influential people in the computer industry in the last twenty years".

Samenvatting

What is an object and what isn't?

Object-oriented programming is among our most effective techniques for managing complexity. In places, this has led to the belief that everything is best thought of as an object and best represented in a program as an object. However, not everything is an object and not every object is best represented using a single language mechanism. I will give a few examples of what can usefully be considered objects and what are best thought of as non-objects. I will discuss a few aspects that can make an object useful in a system. I will illustrate these ideas using C++ examples.

EDMUND M. CLARKE

Edmund M. Clarke studied mathematics at the University of Virginia and at Duke University, and received a Ph.D. degree in Computer Science from Cornell University in 1976. After positions at Duke University and Harvard University he joined in 1982 the Computer Science Department of Carnegie-Mellon University, where he was appointed Full Professor in 1989. In 1995 he became there the first recipient of the FORE Systems Professorship. Clarke's research interests focus on the automatic verification of computer hardware and software. His research group has developed a verification method called 'temporal logic model checking'.
The method is applied to, e.g., detecting logical errors in sequential circuit designs and communication protocols. Specifications are expressed in a propositional temporal logic, while circuits and protocols are modeled as state-transition systems. The group succeeded in checking some examples that would have required 10^{20} states with the original algorithm, and even cases with up to 10^{100} states have been successfully tackled. In combination with various abstraction techniques even larger systems could be verified, for example the cache coherence protocol in the IEEE Futurebus+Standard, in which several previously undetected errors were unveiled -- the first time that with formal methods nontrivial errors were detected in an IEEE standard.
Clarke was awarded the 1998 ACM Kanellakis Award for Theory and Practice for the joint invention with Randal E. Bryant, E. Allen Emerson, and Kenneth L. McMillan of symbolic model checking.

Samenvatting

Symbolic Model Checking without BDDs

Symbolic Model Checking has proven to be a powerful technique for the verification of reactive systems. BDDs have traditionally been used as a symbolic representation of the system. We show how boolean decision procedures, like Stålmarck's Method or the Davis & Putnam Procedure, can replace BDDs. This new technique avoids the space blow up of BDDs, generates counterexamples much faster, and sometimes speeds up the verification. In addition, it produces counterexamples of minimal length. We introduce a ***bounded model checking*** procedure for LTL which reduces model checking to propositional satisfiability. We show that bounded LTL model checking can be done without a tableau construction. We have implemented a model checker ***BMC***, based on bounded model checking, and preliminary results will be presented.