Department of Computer Sciences
The University of Texas at Austin

presents

In Pursuit of Simplicity

A Symposium Honoring
Professor Edsger Wybe Dijkstra

May 12–13, 2000

A

De ingeschrevene is verplicht deze kaart bij zich te hebben wanneer hij(zij) een college, inrichting of verzameling bezoekt en hem op verzoek van een controlerend ambtenaar te tonen.

RIJKSUNIVERSITEIT TE LEIDEN

STUDIEJAAR 1954—1955

No. 0459

De RECTOR MAGNIFICUS verklaart, dat

mej. / de heer E. W. Dijkstra

als STUDENT is ingeschreven en mitsdien gedurende bovenvermeld studiejaar toegang heeft tot alle lessen der Rijksuniversiteiten, alsmede tot de daarbij behorende inrichtingen en verzamelingen.

Leiden, 23 Sep. 1954

[Handtekening van de ingeschrevene (ne varietur)]

E. W. Dijkstra

J. A. Bakhuizen van den Brink

11826 - '54 (12264)

This certificate shows that the student E. W. Dijkstra paid his university tuition for the academic year 1954-1955 and hence was entitled to follow lectures, visit the library, etc. Each student was required to be in possession of their certificate when in any university building and officials would sometimes spot check.

Contents

In Pursuit of Simplicity

A Symposium Honoring Professor Edsger Wybe Dijkstra

May 12-13, 2000

Department of Computer Sciences
Taylor Hall, Room 2.124
The University of Texas at Austin
Austin, TX 78712-118
(512) 471-7316
Fax (512) 471-8885
www.cs.utexas.edu

Symposium Committee:
Vicki L. Almstrum
Stephen W. Keckler, Chair
Jayadev Misra
Suzanne Kain Rhoads
Hamilton Richards

The Program

Friday, May 12th, Taylor Hall 2.106

8:00-8:30 *Registration*

8:30-8:45 *Welcome*
William Livingston, Professor Emeritus, Department of Government, College of Liberal Arts, The University of Texas at Austin
Benjamin J. Kuipers, Chairman, Department of Computer Sciences, The University of Texas at Austin

Session 1 **Chair: Vicki Almstrum, The University of Texas at Austin**

8:45-9:15 *A Real-time Application of Oberon-SA: Flight Control of Model Helicopters*
Niklaus Wirth, ETH Zurich

9:15-9:45 *The Music of Streams*
M. Douglas McIlroy, Darmouth College

9:45-10:15 *Panel*
Anish Arora, Ohio State University at Columbus and Microsoft Corporation
Kenneth L. Calvert, University of Kentucky
Joshyula R. Rao, IBM T. J. Watson Research Center
Ambuj Singh, University of California at Santa Barbara

10:15-10:45 *Break*, Taylor Hall, Room 3.128

Session 2 **Chair: Mohamad Gouda, The University of Texas at Austin**

10:45-11:15 *Model-Based Specification*
Lex Bijlsma, Utrecht University

11:15-11:45 *The Alma Project or How First-Order Logic Can Help Us in Imperative Programming*
Krzysztof R. Apt, CWI Amsterdam and University of Amsterdam

11:45-12:15 *Panel*
Jim Anderson, University of North Carolina at Chapel Hill
K. Rustan M. Leino, Compaq Systems Research Center
Natarajan Shankar, SRI Computer Science Laboratory, Menlo Park

12:15-2:00 *Lunch*, The University Texas Club

Session 3 **Chair: Jayadev Misra, The University of Texas at Austin**

2:00-2:30 *A Theory of Composition Motivated by wp*
K. Mani Chandy, California Institute of Technology

2:30-3:00 *Are there Systems Principles or only Systems Principals?*
Fred B. Schneider, Cornell University

3:00-3:30 *Panel*
Rajeev Joshi, Compaq Systems Research Center
Markus Kaltenbach, Siemens Information and Communication Network, Inc., Munich
Jacob Kornerup, National Instruments Corporation, Austin
Panagiotis Manolios, The University of Texas at Austin

3:30-4:00 *Break*, Taylor Hall Room 3.128

Session 4 **Chair: Stephen Keckler, The University of Texas at Austin**

4:00-4:30 *Formality Works*
Rob R. Hoogerwoord, Eindhoven University of Technology

4:30-5:00 *Frolicking with Formulae*
W. H. J. Feijen, Eindhoven University of Technology

Banquet **Lila B. Etter Alumni Center**

7:00-10:00 *Banquet Speakers*
David Gries, University of Georgia at Athens
Hamilton Richards, The University of Texas at Austin
W. M. Turski, Warsaw University

Saturday, May 13th, Taylor Hall 2.106

Session 1 **Chair: J Strother Moore, The University of Texas at Austin**

9:00-9:30 *Structured Programming and Literate Programming*
Donald E. Knuth, Stanford University

9:30-10:15 *Panel*
Don Braben, Venture Research International and University College London
Anne Kaldewaij, University of Amsterdam
Martin Rem, Eindhoven University of Technology
Netty van Gasteren, Eindhoven University of Technology

10:15-10:45 *Break*, Taylor Hall, Room 3.128

Session 2 **Chair: Hamilton Richards, The University of Texas at Austin**

10:45-11:15 *Drawing Lots for Sinterklaas*
Rutger M. Dijkstra, Eye-to-Eye, Groningen

11:15-11:45 *What I learned from VLSI: An Imaginary Return to the Tuesday Afternoon Club*
Alain J. Martin, California Institute of Technology

11:45-12:15 *Panel*
Ted Herman, University of Iowa
Charanjit S. Jutla, IBM T. J. Watson Research Center
David A. Naumann, Stevens Institute of Technology, New Jersey

12:15-1:30 *Lunch*, The University Texas Club

Session 3 **Chair: Robert Boyer, The University of Texas at Austin**

1:30-2:00 *Legacy*
C. A. R. Hoare, Microsoft Corporation, Cambridge

2:00-2:30 *Juno-2, a Constraint-Based Graphics System*
Greg Nelson, Compaq Systems Research Center

2:30-3:00 *Break*, Taylor Hall Room 3.128

3:00-4:00 *Farewell Lecture*
Edsger W. Dijkstra

4:00 *Closing Remarks*
Jayadev Misra

E. W. Dijkstra

Education

Gymnasium Erasmianum in Rotterdam
(with highest possible marks for mathematics, physics, chemistry, and biology), 1948

University of Leyden
Candidaats Examen, Mathematics and Physics, 1951
Doctoraal Examen, Theoretical Physics, 1956

University of Amsterdam
Ph.D. (Cum Laude), 1959

Professional Experience

Mathematical Centre, Amsterdam
Staff member, 1952-1962

Eindhoven University of Technology
Full Professor of Mathematics, 1962-1973
Burroughs Research Fellow / Professor Extraordinarius, 1973-1984

The University of Texas at Austin
Professor and Schlumberger Centennial Chair in Computer Sciences, 1984-1999

Honors and Awards

- Member of the Bataafsch Genootschap de Proefondervindelijke Wijsbegeerte, Rotterdam, 1964
- Member of the Royal Netherlands Academy of Arts and Sciences, 1971
- Distinguished Fellow of the British Computer Society, 1971
- Programming Systems and Languages Paper Award, 1971
- ACM Turing Award, 1972
- AFIPS Harry Goode Memorial Award, 1974
- Foreign Honorary Member of the American Academy of Arts and Sciences, 1975
- Doctor of Science Honoris Causa, The Queen's University of Belfast, 1976
- Computer Pioneer Award, IEEE Computer Society, 1982
- ACM/SIGCSE Award for Outstanding Contributions to Computer Science Education, 1989
- ACM Fellow, 1994

Written Works

According to his Curriculum Vitae, E. W. Dijkstra is the author or co-author of nine books, twelve book chapters, forty journal articles, thirty-four contributions to conference proceedings, and twenty-two other publications. However, these statistics only scratch the surface of his writings. The EWD series comprises over 1000 manuscripts (with EWD 1294 the highest in the series at this moment) that include technical papers, trip reports, and essays. While many of these are included among the published works mentioned above, many more are not. The spread of these manuscripts represents a unique form of "publication" via an informal distribution tree. Each EWD manuscript goes to a small group of people, each of whom acts as a new distribution point to send copies to additional people — all over the world. At least one paper returned to him six generations later. The Department of Computer Sciences is currently organizing these writings into a web site, which will be located at "Edsger W. Dijkstra archives" under Publications on the UT CS home page at www.cs.utexas.edu.

Selected Citations

Citation from the ACM Turing Award

The working vocabulary of programmers is studded with words originated or forcefully promulgated by E. W. Dijkstra: display, deadly embrace, semaphore, go-to-less programming, structured programming. But his influence on programming is more pervasive than any glossary can possibly indicate. The precious gift that this Turing Award acknowledges is Dijkstra's style: his approach to programming as a high, intellectual challenge; his eloquent insistence and practical demonstration that programs should be composed correctly, not just debugged into correctness; and his illuminating perception of problems at the foundations of program design. He has published about a dozen papers, both technical and reflective, among which are especially to be noted his philosophical address at IFIP, his already classic papers on cooperating sequential processes, and his memorable indictment of the go-to statement. An influential series of letters by Dijkstra have recently surfaced as a polished monograph on the art of composing programs. We have come to value good programs in much the same way as we value good literature. And at the center of this movement, creating and reflecting patterns no less beautiful than useful, stands E. W. Dijkstra.

From citation read by M. D. McIlroy, chairman of ACM Turing Award Committee, on August 14, 1972, at ACM's annual conference in Boston, MA, when Edsger presented his talk "The Humble Programmer" and received the 1972 ACM A. M. Turing Award.

Citation from the ACM/Fellows Award

Edsger Dijkstra was a principal contributor in the late 1950's to the development of the ALGOL, a high level programming language which has become a model of clarity and mathematical rigor. He is one of the principal exponents of the science and art of programming languages in general, and has greatly contributed to our understanding of their structure, representation, and implementation. His fifteen [sic] years of publications extend from theoretical articles on graph theory to basic manuals, expository texts, and philosophical contemplations in the field of programming languages.

Appears at http://www.acm.org/awards/fellows_citations/dijkstra.html

The 25th Anniversary Issue of *Communications of the ACM*

In planning the content of this issue, it was decided to let *Communications* speak for itself, and keep pontification, opinion, etc. to a minimum ... Accordingly, the body of the issue consists of reprints from the *Communications* of earlier days. All are papers (or, in some cases, letters) which seem to have set forth ideas which later proved significant and seminal for the computing field (indicated in many cases by high citation frequency). ... The final set consists of 21 papers which appeared from 1960 to 1978. ... The accompanying paragraphs were generated by the efforts of future Editor-in-Chief P. J. Denning and his colleagues at Purdue University ...

CACM, January 1983, Vol 26, No. 1 [The last *CACM* cover with the black background accented by lettering and sketches in white and blue ...]

Introduction to "Solution of a Problem in Concurrent Programming Control" in 25th *CACM* issue

This short paper marks the beginning of a long era of interest in expressing the synchronization of concurrent processors using ordinary programming languages. This paper considers the problem of implementing an indivisible operation by mutual exclusion of critical sections of code. Later papers by Dijkstra introduced the concepts of semaphores; one of the motivations of semaphores was reducing the complexity of the programming required to implement mutual exclusion. (A letter to the editor by Don Knuth, published in May 1966, points out that Dijkstra's solution is susceptible to "starvation" — meaning that a particular processor can fortuitously be forever blocked from entering the critical section. Knuth's starvation-free solution is even more complex than the solution here.) — P. J. D. [Reprinted from *CACM*, September 1965, Vol 8, No. 9, p. 569]

Introduction to "The Structure of 'THE'-Multiprogramming System" in 25th *CACM* issue

This was another of the papers presented at the first symposium on operating systems principles (SOSP-1) in 1967. Dijkstra reported on his approach to structuring the multiprogramming system at the Technische Hoogeschool Eindhoven (THE) as a hierarchy of nested abstract machines. The hierarchy was implemented as a series of layers of software: each extended the instruction set of the machines below it and hid the details of its resource management from the levels above it. This project initiated a long line of research in multilevel systems architecture — a line that continues to the present day because hierarchical modularity is a powerful approach to organizing large systems. (At the suggestion of the editor, Dijkstra added a short appendix about the P and V operations on semaphores, another of his concepts that was then in the formative stages.) — P. J. D. [Reprinted from *CACM*, May 1968, Vol 11, No. 5, pp. 341-346]

ACM Monthly Classics

The ACM web site includes the results of a project during 1965 and 1996 to select classic articles among those published by the ACM. While little information about this project is available, the web site does list eight monthly classics, covering articles published in *Communications of the ACM* in the years 1968 to 1984. The classics with the earliest publication dates are both by E. W. Dijkstra.

ACM Classic of the month, October 1995:

> Go To Statement Considered Harmful, Reprinted from *Communications of the ACM*, Vol. 11, No. 3, March 1968, pp. 147-148. Copyright (c) 1968, Association for Computing Machinery, Inc. On the Internet: http://info.acm.org/classics/oct95/

ACM Classic of the month, March 1996:

> The Structure of "THE"-Multiprogramming System Appendix, Reprinted from *Communications of the ACM*, Vol. 11, No. 5, May 1968, pp. 345-346. Copyright (c) 1968, Association for Computing Machinery Inc. On the Internet: http://info.acm.org/classics/mar96/

ISI Citation Classics

The ISI (Institute for Scientific Information) is a database publisher with a focus on Web-based products that offer scholarly research information in the sciences, social sciences, and arts and humanities. Among their products are the Science Citation Index and the Social Sciences Citation Index. As ISI collects citation statistics, they periodically identify works that have become "citation classics." The choice of a citation classic is context sensitive with respect to academic discipline, taking into consideration different thresholds of citation frequency for different fields. Once a paper or book is chosen as a citation classic, the author of the work is invited to create a 500-word commentary that describes the author's research, its genesis, and circumstances that affected its progress and publication. The author is encouraged "to include the type of personal details that are rarely found in formal scientific publication, such as obstacles encountered and by-ways taken. We also asked them that they mention the contributions of co-authors, any awards or honors they received for their research, and any new terminology arising from their work. Finally, we asked them to speculate on the reasons for their paper or book having been cited so often." [from Preface by E. Garfield, ISI *Current Contents*]

One article and one book by E. W. Dijkstra have reached the status of citation classics, each with more than 900 citations in the Web of Science database during the period 1945 to 2000. Following are excerpts from the two essays about these works.

An essay by E. W. Dijkstra, dated December 8, 1982, reflecting on the citation classic "A note on two problems in connexion with graphs" [*Numr. Math.* 1:269-71, 1959] in *Current Contents*, "This Week's Citation Classic," CC/Number 7, February 14, 1983 [EWD 841A].

> At the time, I was fully aware of having found two gems: at the moment of discovery I was duly excited and I remember having presented them in a lecture. In retrospect, it seems strange that I waited another two years before I submitted them for publication. The main reason was the absence of journals dedicated to automatic computing, something the Foundation of Numerische Mathematik sought to remedy. I wrote and submitted my little article — my second one — trying to assist the fledgling. Furthermore, the publish-or-perish syndrome had not reached the Netherlands yet.
>
> This paper has been cited for the following reason. The solutions, as elegant as effective, now belong to the intellectual baggage of any well-educated computer scientist, and the first one to publish them gets the credit and "collects" the references. I am pleased to see that by now the computing community is beginning to regard them as "common property" and to refer to them without mentioning my name: their birth has become history (as it should be).

An essay by E. W. Dijkstra, dated May 8, 1988, reflecting on the citation classic *A Discipline of Programming* [Englewood Cliffs, NJ: Prentice-Hall, 1976] in *Current Contents*, "This Week's Citation Classic," CC/Number 40, October 3, 1988.

> In the early 1970s I knew I had to forge programming into an effective mathematical discipline and got my first glimpse of how to do that. This vision clashed with that of my Department of Mathematics at Eindhoven University of Technology, which I left in 1973 to become a Burroughs Research Fellow (with the generous charter "to do my own thing"). Then things moved quickly.
>
> At the Blanchland meeting of the International Federation for Information Processing Working Group 2.3 on "Programming methodology," during the night of October 23-24, 1973, all the pieces fell into place. That evening I could not sleep and found myself preparing my lecture; with my brain burning, I left my bed at 2:30 a.m. and wrote for more than an hour.
>
> It was not only my former department that was not ready for it. When, eight months later, I submitted a paper under the title "Guarded commands, nondeterminacy and a calculus for the derivation of programs" to the *Communications of the ACM*, one referee objected so strongly that I had to remove "a calculus" from the title [changing it to "Guarded commands, nondeterminacy and formal derivation of programs"]. This paper [*CACM* 18:453-8. 1975, cited 85 times)] presented the bare ingredients; the monograph *A Discipline of Programming* developed the methodology and applied it to many programming problems.

Insights to the Life and Times of Edsger W. Dijkstra

Having elected to study theoretical physics, Dijkstra observed that many problems in the field required extensive calculation, so he decided to learn to program. Thanks to their wartime code-breaking work, the British led the development of European computing in the 1940s and 1950s. In 1951, Dijkstra attended a summer school in programming at Cambridge University. In March 1952 he got a part-time job at the Mathematical Centre in Amsterdam, where he became progressively more involved in computer programming. ... The hardware designers ... "would never include something in the machine unless I thought it was okay. I was to write down the functional specification that was the machine's reference manual. They referred to it as "The Appalling Prose" — it was as rigorous as a legal document."

Dijkstra had still not committed himself to a career in programming — a field that was virtually unknown in the Netherlands. "I finished my studies at Leiden as quickly as possible. Physics was a very respectable intellectual discipline. I explained to [Aad] van Wijngaarden [his advisor and an early Dutch computer pioneer] my hesitation about being a programmer. I told him I missed the underlying intellectual discipline of physics. He agreed that until that moment there was not much of a programming discipline, but then he went on to explain that automatic computers were here to stay, that we were just at the beginning and could not I be one of the persons called to make programming a respectable discipline in the years to come?" Dijkstra's doubts reflected the widespread ignorance about programming. When Dijkstra applied for a license to marry M. C. Debets, a colleague who was also a programmer, the bureaucrats did not recognize programmer as a profession, so he reluctantly labeled himself a "theoretical physicist."

Source: "Edsger W. Dijkstra: Appalling Prose and the Shortest Path", in *Out of their Minds, The Lives and Discoveries of 15 Great Computer Scientists*, D. Shasha and C. Lazere, Copernicus, 1995.

When looking back on any kind of movement or revolution, one always likes to point to a beginning: "It began right there — it all started with so-and-so's famous speech...." If structured programming can be thought of as a revolution, then surely Dijkstra's landmark paper, "Programming Considered as a Human Activity," published in 1965, marks its beginning. Virtually the entire gospel of structured programming is contained in a few short pages: the arguments against goto statements, the notion of top-down design, the emphasis on program correctness and quality (or "elegance," as Dijkstra prefers to call it), and the strong argument against programs that modify themselves.

In addition to these fundamental concepts, there are some rather classic phrases that first appeared in this paper, and that have popped up in dozens of subsequent articles and books. For example, when discussing the "divide and conquer" approach characterizing top-down design, Dijkstra admits, "I have only a very small head, and must live with it." What seems obvious to us today was a rather novel idea in 1965: the idea that while computers were — and still are — getting faster and more powerful, human beings weren't. This theme is repeated again and again, and is essentially the entire subject matter of Dijkstra's 1972 speech, "The Humble Programmer."

... [O]nce you do read it, you can see why Dijkstra has the reputation he does. His writing is succinct and yet convincing. Reading Dijkstra, someone said, has been compared to eating marzipan — it's best to take very small bites, chew slowly, and digest the mouthful before moving on to the next bite.

Source: *Classics in Software Engineering*, edited by E. Y. Yourdin, Yourdin Press, 1979.

In 1967, when he had been at Eindhoven five years, Dijkstra's professional life was falling apart. His colleagues in the mathematics department, still contemptuous of computing science, had essentially rejected the thesis of his first Ph.D. student. With his operating system completed, he sank into a depression, and the depression worsened. "I realized that my previous projects had only been agility exercises. I now had to confront complexity itself and try to find out the best way to do difficult things. But it took me a long time to gather the courage to do that. Alan Turing committed suicide; Kurt Gödel was on and off in a mental hospital. I was terribly frightened.

"I greatly admire my wife for the way in which she guided me through that period. I was essentially incommunicado, hardly spoke, did not work. I would sit all evening silently staring at the white walls in our living room. Finally, one night at half past two, my wife collected me weeping on the carpet in that room.

"From that moment I realized that something had to be done. I started writing 'Notes on Structured Programming' for therapeutic reasons. When that text was written, I knew what I had to do, and I knew how I was going to attack it.

"The history of that text is also very amazing. I sent a copy to all of my colleagues in the department, and the reaction was silence or

scathing remarks. But by that time I had recovered, and their remarks did not hurt me. So I mailed 20 copies to colleagues, half in Europe, half in the United States. Now, thanks to the Xerox machine, that text has been copied by thousands and thousands. It has gone over the world like wildfire. Years later, I still encounter people who say that they have a fourth or fifth generation copy, barely readable."

Source: "The Sage of Software," S. Olson, *Science*, January/February 1984.

Among the 50 or so participants [at the Working Conference on Software Engineering, held October 7 to October 10, 1968, in Garmisch, Germany], E. W. Dijkstra was dominant. He actually made not only cynical remarks like "the dissemination of error-loaded software is frightening" and "it is not clear that the people who manufacture software are to be blamed. I think manufacturers deserve better, more understanding users." He also said already at this early date, "Whether the correctness of a piece of software can be guaranteed or not depends greatly on the structure of the thing made," and he had very fittingly named his paper "Complexity Controlled by Hierarchical Ordering of Function and Variability," introducing a theme that followed his life the next 20 years. Some of his words have become proverbs in computing like "testing is a very inefficient way of convincing oneself of the correctness of a program."

Source: Friedrich L. Bauer's foreword to *Software Engineering — A European Perspective*, edited by R. H. Thayer and A. D. McGettrick, p. vi, IEEE, 1993.

Since the summer of 1973, when I became a Burroughs Research Fellow, my life has been very different from what it had been before. The daily routine changed; instead of going to the University each day, where I used to spend most of my time in the company of others, I now went there only one day a week and was most of the time — that is, when not travelling! — alone in my study. In my solitude, mail and the written word in general became more and more important. The circumstance that my employer and I had the Atlantic Ocean between us was a further incentive to keep a fairly complete record of what I was doing. The public part of that output found its place in what became known as "the EWD series", which can be viewed as a form of scientific correspondence, possible since the advent of the copier. (That same copier makes it hard to estimate its actual distribution; I myself make about two dozen copies of my texts, but their recipients were welcome to act as further nodes of the distribution tree.) ... *Nuenen, 19 July 1981 Edsger W. Dijkstra*

Source: Preface to *Selected Writings on Computing: A Personal Perspective*, E. W. Dijkstra, Springer-Verlag, 1982.

Officials of The University of Texas at Austin believe that they have scored a significant coup in persuading Edsger W. Dijkstra to [be the first to occupy an] endowed chair in computer science. A winner of the Association of Computer [sic] Machinery's prestigious A. M. Turing Award in 1972 for his work in programming languages and structure, Mr. Dijkstra has, for the last 11 years, been the Burroughs Corporation's only research fellow — his job has been to think.

Source: *The Chronicle of Higher Education*, J. A. T., April 4, 1984, p. 12.

Edsger is, with all that the word implies, a perfectionist. His programming and his mathematics are strongly guided by his concern for clarity of notation and exposition, and indeed for what he quite justifiably terms beauty. Thus his descriptions of problems and solutions, both in his lectures and published papers, and in his EWD series of documents, distributed via a network of friends and colleagues in a fashion reminiscent of Russian "samizdat" literature, are often vivid and compelling.

Source: Brian Randell, foreword to *Beauty is Our Business*, edited by W. H. J. Feijen, A. J. M. Van Gasteren, D. Gries, and J. Misra, Springer-Verlag, 1990.

Dijkstra's influence can be attributed to a penetrating mind; a rare intellectual honesty, which does not allow him to compromise his principles; and a way with words that few computer scientists can match. These factors make him appear caustic, at times, as he says what he believes but not what we want to hear. (A colleague once said, "Dijkstra's right, but you don't say those things.")

Source: Historical Note 10.1, *A Logical Approach to Discrete Math*, D. Gries and F. B. Schneider, Springer-Verlag, 1993.

In Edsger's Words

A selection of favorite quotes collected from Vicki Almstrum, Tony Hoare, Niklaus Wirth, Wim Feijen, and Rajeev Joshi. Bracketed supplemental information after many quotes indicate dates when the statement was spoken or published, translations, or other clarifying information known to the person submitting the quote.

Advice to a promising researcher: "Only do what only you can do."

"The problems of the real world are those that remain when you ignore their known solutions."

"Always design your program as a member of a whole family of programs, including those that are likely to succeed it."

"Avoid operational reasoning like the plague."

"Separate concerns."

"The prisoner falls in love with his chains."

"A programming language is a tool that has a profound influence on our thinking habits."

"I pray daily that more of my fellow programmers may find the means of freeing themselves from the curse of compatibility." [1972, Turing award lecture]

"The program and the correctness proof grow hand in hand." [1972, Turing award lecture]

"Brainpower is by far our scarcest resource." [1972, Turing award lecture]

"In their capacity as a tool, computers will be but a ripple on the surface of our culture. In their capacity as intellectual challenge, they are without precedent in the cultural history of mankind." [1972, Turing award lecture]

"The competent programmer is fully aware of the strictly limited size of his own skull; therefore he approaches the programming task in full humility, and among other things he avoids clever tricks like the plague." [1972, Turing award lecture]

"For the absence of a bibliography I offer neither explanation nor apology." [foreword to *A Discipline of Programming*, Prentice-Hall, 1976]

"Progress is possible only if we train ourselves to think about programs without thinking of them as pieces of executable code." [August 23, 1979]

"This solution is gloriously non-deterministic." [August 23, 1979]

"My daughter and I taking a shower with equal frequency is a frightening thought for both of us." [February 21,1984; commenting on a mutual exclusion algorithm in which the two components are alternately granted access to the critical section.]

"De plichten van een docent zijn divers, die van het gehoor ook." ["The duties of a teacher are several, but those of the audience as well."]

"Ik hou van wiskunde, maar spaar me de mathematen." ["I love mathematics, it's the mathematicians I cannot stand."]

"If somewhere you read 'in depth', ignore it."

"Nothing is as expensive as making mistakes."

"Program testing can at best show the presence of errors, but never their absence."

"Software Engineering is Programming when you can't."

"We must give industry not what it wants, but what it needs."

"Waiting is a very funny activity: you can't wait twice as fast." [February 28, 1984]

"It helps hand-eye coordination if, as you're doing your formulae, you gently sing the notation." [September 1, 1992]

"Do not try to change the world. Give the world the opportunity to change itself." [April 21, 2000]

"While current curricula extensively teach existing mathematics, they pay scant attention to the doing of mathematics, i.e., to the question of how to design and to present solutions. If any attention to these issues is paid at all, they are treated separately: design of solutions, i.e., "problem solving" or "mathematical invention", is viewed as a psychological issue, as a matter of mathematical intuition, while presentation is viewed as a matter of personal style or as an issue of education. Most mathematicians consider psychology and pedagogy as sciences too soft to be respectable, and consequently the subject of how to do mathematics has almost been tabooed." [foreword to *On the Shape of Mathematical Arguments*, A. J. M. Van Gasteren, Lecture notes in Computer Science, edited by G. Goos and J. Hartmanis, Springer-Verlag, 1987]

"Teaching to unsuspecting youngsters the effective use of formal methods is one of the joys of life because it is so extremely rewarding. Within a few months, they find their way into a new world with a justified degree of confidence that is radically novel for them; within a few months, their concept of intellectual culture has acquired a radically new dimension. To my taste and style, that is what education is about. Universities should not be afraid of teaching radical novelties; on the contrary, it is their calling to welcome the opportunity to do so." ["On the Cruelty of Really Teaching Computing Science," *CACM*, 32(12), December 1989, page 1404]

"So-called "natural language" is wonderful for the purposes it was created for, such as to be rude in, to tell jokes in, to cheat or to make love in (and Theorists of Literary Criticism can even be content-free in it), but it is hopelessly inadequate when we have to deal unambiguously with situations of great intricacy, situations which unavoidably arise in such activities as legislation, arbitration, mathematics or programming." [foreword to *Teaching and Learning Formal Methods*, edited by C. N. Dean and M. G. Hinchey, Academic Press, 1996]

"I mean, if 10 years from now, when you are doing something quick and dirty, you suddenly visualize that I am looking over your shoulders and say to yourself, "Dijkstra would not have liked this", well that would be enough immortality for me." ["Introducing a course on calculi" (EWD 1213), August 1995]

"Many mathematicians derive part of their self-esteem by feeling themselves the proud heirs of a long tradition of rational thinking; I am afraid they idealize their cultural ancestors." ["A stupid notation" (EWD 782)]

"The traditional mathematician recognizes and appreciates mathematical elegance when he sees it. I propose to go one step further, and to consider elegance an essential ingredient of mathematics: if it is clumsy, it is not mathematics." ["On the Economy of doing Mathematics" (EWD 1130), June 1992]

"Don't compete with me: firstly, I have more experience, and secondly, I have chosen the weapons." [During first lecture in "Capita Selecta," August 29, 1996]

"Aim for brevity while avoiding jargon." [Lecture in "Capita Selecta", September 5, 1996]

"Maintaining a large range of agilities —mental and physical— requires regular exercise [..]. That is why the capable are always busy." [Lecture, "Capita Selecta", October 10, 1996]

"Mathematicians are like managers — they want improvement without change." [During a meeting of the Austin Tuesday Afternoon Club, Fall 1996]

"And even now my first reaction to formulae, written by someone else, is one of repulsion —in particular when an unfamiliar notational convention is used — and when reading an article, my natural reaction is to skip the formulae." ["My hopes of computing science" (EWD 709)]

"Show any mathematician a really elegant argument that is new for him: at the moment it becomes his intellectual property, he starts to laugh!" ["My hopes of computing science" (EWD 709)]

... I had already come to the conclusion that in the practice of computing, where we have so much latitude for making a mess of it, mathematical elegance is not a dispensable luxury, but a matter of life and death." ["My hopes of computing science" (EWD 709)]

"For me, the first challenge for computing science is to discover how to maintain order in a finite, but very large, discrete universe that is intricately intertwined. And a second, but not less important challenge is how to mould what you have achieved in solving the first problem, into a teachable discipline: it does not suffice to hone your own intellect (that will join you in your grave), you must teach others how to hone theirs. The more you concentrate on these two challenges, the clearer you will see that they are only two sides of the same coin: teaching yourself is discovering what is teachable." ["My hopes of computing science" (EWD 709)]

Blackboard Commentary

Being a selection of quotes posted by EWD at the beginning of various lectures, as collected by Vicki Almstrum, Pete Manolios, Tony Hoare, Wim Feijen, Rajeev Joshi, and Rustan Leino. Bracketed dates are occasions when the associated quote was posted.

Arabic ("Algol") = "demon"
Astronomic ("Algol") = "intermittent star" [August 23, 1979]

More than anything else, mathematics is a method." *Morris Kline* [August 24, 1979]

It is always difficult, and no mean feat, to change the tires of a moving vehicle." *Kolodzei*

"academic: a term of opprobrium applied to those that do their job well by those who cannot."
Sir Ernest Gowers in The Complete Plain Words

"When one cannot invent, one must at least improve." *fortune cookie* [September 9, 1992]

"Don't try to teach a pig to sing. It wastes your time and annoys the pig." *Anonymous* [September 29,1994]

"Shut up and code" *Notice on the office wall of an industrial software manager, 1970* [October 6, 1994]

"elegant: [...] ingeniously simple and effective" *Concise Oxford Dictionary* [November 11, 1994]

"intelligent: 1[...]; 2[...]; 3: able to perform some of the functions of a computer [...]" *Webster's Collegiate Dictionary* [December 1, 1994]

"A body on which no forces are exerted persists in its state of rest or uniform motion." *After I. Newton* [February 21, 1984; Edsger expresses his 'abhorrence' of superfluous case-analysis.]

"De fluit herwint terrein." *Hubert Bahrwasser (1950)* [Literally: "The flute regains lost grounds"] [February 28, 1984]

"Oh, that would be a great idea!" *Mahatma Gandhi (when asked what he thought of Western civilization)* [March 20, 1984]

"It is always difficult to think and reason in a new language; and that difficulty discouraged all but men of energetic minds." C*harles Babbage* [March 27, 1984]

"I have a Bosendorfer too!" *Donald E. Knuth* [April 24, 1984]

"Ziezo, Die hangt weer." *Jo Vincent* [Roughly: "Okay, we hung Him again."] [1984.05.08; Jo Vincent, a soprano, once said this after her Nth performance (N>>>0) of Bach's St. Matthew Passion]

"Characteristica Universalis." *G. W. Leibniz* [October 5, 1982]

"But every genius is apt to be much of an ass at times." *Dame Rebecca West*

"I can't think why fancy religions should have such a ghastly effect on one's grammar." *Dorothy L. Sayers*

"Laat dan de linkerhand niet weten wat de rechter doet." ["Let then the left hand not know what the right hand does."] *Mattheus 6:3*

"His opera 'Palestrina' and his very elaborate romantic cantata 'Von Deutscher Seele' are works which move his admirers to reverence."
Percy A. Scholes re: Hans Pfitzner

"Omit needless words." *Strunk and White, Elements of Style*

"Conducting is generalship on the battlefield of music." *Percy A. Scholes*

"Many people would sooner die than think. In fact they do." *Bertrand Russell*

"noe 2 thynges can be moare equalle" *Robert Recorde, 1557, in 'Whetstone of Witte', defending symbol '=' for equality*

"Als het honden waren, zou je ze niet naar de tentoonstelling sturen." *Edsger's father* [Literally: "If they were dogs, you wouldn't send them to the exhibition."]

"Mathematical symbols cross linguistic borders less readily than do mathematical ideas [...]. In fact, no American mathematical symbol, except perhaps the dollar mark ($), has found acceptance in Europe." *Florian Cajori, 1929*

"I never read a book before reviewing it — it prejudices one so." *Sidney Smith*

"Zorg dat je formules kent en onthoud dat, als je meer dan vijf regels nodig hebt, je op de verkeerde weg bent." *Edsger's mother* ["See to it that you know your formulae, and remember that whenever you need more than five lines, you are on the wrong track."] [This advice was given to Edsger by his mother when he had his first lessons in trigonometry.]

"If words could cure the ills of our profession of programming, how healthy and highly respected a profession it now would be." *C.A.R. Hoare*

"She did her best, but her best was none too good." *Inscription on a grave*

"Brilliant trickery is no longer considered reputable mathematics." *E.T. Bell (in The Development of Mathematics*)

"Some problems are better evaded than solved." *C.A.R. Hoare*

"Vooruitgang pappa, daar helpt niets tegen." *Rutger M. Dijkstra* ["Progress, daddy, there is no cure for it."]

"Nostalgia ain't what it used to be." *Anonymous*

"Das Publikum ist wie ein Schwein: es frisst alles, sogar das Gute." *Arthur Schnabel* ["Your audience is like a pig; it swallows everything, even the good stuff."]

"In a right triangle, the sum of the squares is the same" *Theorem of Pythagoras according to Anon.*

"Man aint really evil, he jest aint got any sense." *W. Faulkner*

"The use of anthropomorphic terminology when dealing with computing systems is a symptom of professional immaturity." *EWD 498*

"25 journalists killed in 26 countries" *Headline, International Herald Tribune*

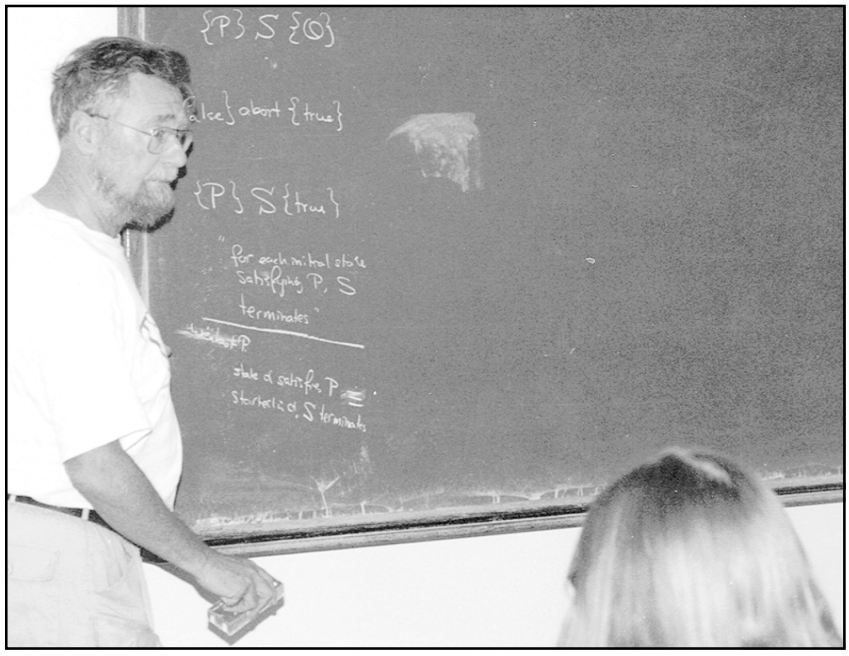
"And, while we are facing facts, we note the opinion of many observers that ever since the days of Aquinas science has been feared and secretly hated by nine human beings out of every ten who have sufficient animation to hate or fear anything." *Eric Temple Bell in The development of Mathematics*

"Machines do not think and therefore can be much more reliable than humans." *F. L. Bauer Marktoberdorf proceedings, 1996* [September 3, 1996]

"Praise the humanities, my boy. That'll make them think you are broadminded." *Winston Churchill's advice to R. V. Jones (who had just been appointed as Professor of Physics)* [September 19, 1996]

"A Conservative Government is an organized hypocrisy." *Benjamin Disraeli (1804-1881)* [November 7, 1996]

"Contributions must be in English or American." *IPL* [November 12, 1996]



"Capita Selecta," Fall 1994

The Speakers

Jim Anderson
The University of North Carolina at Chapel Hill, U. S. A.

Panelist Background: James H. Anderson received a B.S. degree in Computer Science from Michigan State University in 1982, a M. S. degree in Computer Science from Purdue University in 1983, and a Ph.D. degree in Computer Sciences from The University of Texas at Austin in 1990. Since 1993, he has been with the Department of Computer Science at the University of North Carolina at Chapel Hill, where he is currently an Associate Professor. From 1990 – 1993, he was with the Computer Science Department at the University of Maryland. In 1995, Anderson received the U.S. Army Research Office Young Investigator Award, and in 1996, he was named an Alfred P. Sloan Research Fellow. Anderson's main research interests lie within the areas of concurrent and distributed computing and real-time systems.

Krzysztof R. Apt
CWI Amsterdam and University of Amsterdam, the Netherlands
The Alma Project or How First-Order Logic Can Help Us in Imperative Programming

Abstract: A logical approach to computing has been found useful in the realm of functional, logic, and database programming. Imperative programming can also profit from a logical view. To this end, we describe an implemented small programming language called Alma-0 that augments the expressive power of imperative programming by a limited number of features inspired by the logic programming paradigm. These additions make it a more attractive vehicle for problems that involve search. The solutions offered in Alma-0 are substantially simpler than their counterparts written in the imperative or in the logic programming style and can be used for different purposes without any modification. In some cases, programs are equal to their specifications and are, therefore, obviously correct.

Speaker Background: Krzysztof R. Apt is a senior researcher at CWI, Amsterdam and Professor of Computer Science at the University of Amsterdam. From 1987 until 1990 he was the William B. Blakemore Professor in Computer Sciences at The University of Texas at Austin. His research interests include use of logic as a programming language, constraint programming, non-standard forms of reasoning, program correctness and semantics, and methodology of distributed programming. In 1999 Apt founded two new journals in order to create more opportunities to publish in journals owned by non-profit organizations. The journals are *ACM Transactions on Computational Logic* (TOCL) (http://www.acm.org/tocl) and, jointly, with his colleagues, *Theory and Practice of Logic Programming* (TPLP), Cambridge University Press, (http://www.cwi.nl/projects/alp/TPLP/).

Anish Arora
The Ohio State University, and Microsoft Research, U. S. A.

Panelist Background: Anish Arora is Associate Professor of Computer Science at The Ohio State University, Columbus. Many aspects of Arora's work on principles of dependability properties such as fault-tolerance, security and timeliness, methods for design, verification and implementation, and prototypes of distributed and network systems have been influenced by Edsger W. Dijkstra. Presently, he is serving as program chair of the Year 2000 Dagstuhl on Self-Stabilizing Systems—an area that Dijkstra started— and as program committee member of ICNP '00, SRDS '00, FTRTFTS '00 and PODC '00, while visiting Microsoft Research —a company that Dijkstra would likely not have started—on sabbatical.

Lex Bijlsma
Department of Computer Science, Utrecht University, the Netherlands
Model-Based Specification

Abstract: The classical specification formalism involving pre- and post-conditions expressed in program variables cannot directly be applied to the specification of classes, interfaces, components, and design patterns, as the program variables involved are mostly invisible outside of (dynamically selected) implementation code. A more useful specification formalism would operate at the problem domain level rather than at the implementation level. Moreover, it ought to be possible to decompose a complete system's specification into expressions involving only a few related entities. The talk sketches some of the problems that must be surmounted to achieve this and suggests ways of doing so.

Speaker Background: Lex Bijlsma studied number theory at Amsterdam (Ph.D. 1978), Eindhoven, and Paris. In 1983 his attention shifted to computing. He has since worked in the groups of M. Rem, A. Kaldewaij, and R. C. Backhouse (all at Eindhoven University of Technology), and S. D. Swierstra (Utrecht University). Bijlsma's research interest focuses on the interplay between programming language constructs and program construction style, in particular with respect to object-oriented design.

Don Braben
Venture Research International and University College London, U. K.

Panelist Background: Don Braben worked in elementary particle physics at Daresbury, Frascati, and CERN. Later he worked at the Cabinet Office in Whitehall, London. He went on to develop the Venture Research concept to fund, with BP's money, scientists in any discipline radically challenging convention. EWD was a participant in that program for over 10 years. He is a visiting professor in planetary sciences at University College London.

Kenneth L. Calvert
Department of Computer Science, University of Kentucky, U. S. A.

Panelist Background: Kenneth L. Calvert was born and raised in Kansas City, Missouri. He received the Bachelor of Science degree in Computer Science and Engineering from the Massachusetts Institute of Technology in 1979, the M.S. in Computer Science from Stanford University in 1980, and the Ph.D. from the Department of Computer Sciences at The University of Texas at Austin in 1991. From 1979 to 1984 Calvert was a Member of Technical Staff at Bell Telephone Laboratories in Holmdel, New Jersey. From 1991 to 1998 he was with the College of Computing at Georgia Institute of Technology. Calvert's research deals with the design and implementation of computer communication networks.

Mani Chandy
Computer Science Department, California Institute of Technology, U. S. A.
A Theory of Composition Motivated by wp

Abstract: (with Michel Charpentier) This talk explores theories that support system design by composition. The problem is well known: how can properties of a system be proved from properties of its components and properties of composition operators? We postulate properties of composition operators, such as associativity, and then derive theorems about composition from these properties. We study properties and composition operators in general, and we then focus on types of properties that are particularly useful when reasoning about action systems and fairness. We propose steps towards a theory of predicate-transformers for composition. This work is based on the significant body of work on predicate transformers developed by Edsger W. Dijkstra and Carel S. Scholten.

Speaker Background: Mani Chandy got his Ph.D. from MIT, taught at The University of Texas at Austin, and is now the Simon Ramo Professor and Executive Officer of Computer Science at the California Institute of Technology.

Rutger M. Dijkstra
Eye-to-Eye, Groningen
Drawing Lots for Sinterklaas

Abstract: The birthday of Sinterklaas (St. Nicholas, a.k.a. Santa Claus) is traditionally celebrated in the Netherlands with a feast that involves giving presents. Sinterklaas parties at schools are organized such that every child gives one present to one of his/her classmates. Who gives a present to whom is determined by drawing names out of a hat (or any other kind of receptacle). All the names of the children are put into a hat and, subsequently, every child draws the name of the classmate to whom a present will be given. Unfortunately, it occurs regularly that one or more of the children draw their own name, which means that the procedure must repeated. This raises the following question: What is the chance of a successful draw, i.e., what is the chance that no child draws his/her own name? The problem above will serve as a vehicle for illustrating some of E.W.D.'s contributions to the art and science of effective problem solving.

Speaker Background: Rutger M. Dijkstra was born and raised in/near Eindhoven, the Netherlands. In 1992 he got his masters degree in Mathematics from the University of Cologne, Germany. From 1992 to 1998 Dijkstra did research on formal methods at the Computing Science Department of the University of Groningen, the Netherlands. Since then, he has been developing web applications in industry.

W. H. J. Feijen
Department of Mathematics and Computing Science, Technical University Eindhoven, the Netherlands
Frolicking With Formulae

Abstract: We will present one, two, or three more or less (in)significant designs in which the formulae guide the way.

Speaker Background: After having obtained a master's degree in applied mathematics, Wim Feijen joined a small group supervised by Professor Edsger W. Dijkstra, where he participated in transforming the art of programming into a scientific discipline. During the 1980s Feijen became interested in calculational methods for the benefit of both computing science and mathematics. During this period, he also became actively engaged in the formal design of multiprograms—recently publishing a book with Netty van Gasteren. He has conducted the Eindhoven Tuesday Afternoon Club since 1984.

David Gries
Department of Computer Science, University of Georgia, U. S. A.

Banquet Speaker Background: A native of New York, David Gries received his bachelors degree from Queens College in 1960, his Masters degree from Illiniois in 1963, and his doctorate from the Munich Institute of Technology in 1966 (all in Mathematics, since there was no Computer Science at the time). He spent the period 1966–1969 as an Assistant Professor of Computer Science at Stanford. He moved to Cornell because Stanford had no weather. He was in Cornell's Computer Science Department from 1969 to 1999, serving as department chair in 1982–1987. In 1999, he finally escaped Cornell's weather and became a Franklin Professor of Computer Science at the University of Georgia. But he stayed Greek, moving from Ithaca, New York to Athens, Georgia. Gries is known for his research in compiling and in programming methodology and for his books *Compiler Construction for Digital Computers* (1971), *An Introduction to Programming*, a *Structured Approach* (1973, with Dick Conway), *The Science of Programming* (1981), and *A Logical Approach to Discrete Math* (1993, with F. B. Schneider). He is finishing a live text called ProgramLive: an Introduction to Programming using Java. Gries served as Chair of the Computing Research Association (then the Computer Science Board) during its formative years, when it opened an office in Washington and began in earnest to represent computing research interests. He also conducted the Taulbee Surveys in the period 1984–1991 and was responsible for making the surveys so useful by obtaining essentially complete responses from Computer Science departments granting Ph.D.'s. He received the Computing Research Association's 1991 Service Award. Gries has received a number of awards for his contributions to education: the AFIPS Education Award, ACM SIGCSE's education award, ACM's Karlstrom award, the IEEE Computer Society Education Award, and a Cornell Weiss Presidential Fellow Award.

Ted Herman
University of Iowa, U. S. A.

Panelist Background: Ted Herman was a graduate student at the University of Texas at Austin during the mid-eighties to early nineties. During this time, he attended the Tuesday Afternoon Club and began to do research in the field of self-stabilization, an invention credited to Edsger W. Dijkstra. From 1992 to 1994, Herman was a visitor to the Computer Science Department at the University of Utrecht. From 1994 to the present he is an Assistant Professor at the University of Iowa.

C. A. R. Hoare
Microsoft Research Laboratory, Cambridge, U. K.
Legacy

Abstract: Edsger has been the source and inspiration of all my research into the theory of programming. He is the origin of the constructive approach to the question of program correctness. He has advised the avoidance of operational reasoning like the plague. And the prospect of his reading of my work has been a constant incentive to sharpen the clarity of my thinking and my presentation. Following his lead still further, I have now taken an opportunity of a spell of work for industry. I find myself in an environment that is highly adept at operational modeling, and in a company whose major concern is not the construction of new code, but the maintenance and improvement of a vast body of legacy code. I hope to present some grounds for the belief that the lessons we have learned from Edsger provide the essential clue to tackling and solving these problems.

Speaker Background: Tony Hoare has been involved in programming methods and languages since the early 1960s, first at the Queen's University in Belfast, and later at Oxford University, as head of the Programming Research Group. He has contributed to proof techniques for programs and distributed computing. Hoare is a designer of the CSP system on which the OCCAM language and the transputer design are based. Currently at the Microsoft Research Laboratory, Cambridge, his research interests include linking theories of programming, and the application of these theories to the problems of legacy code.

Rob R. Hoogerwoord
Department of Mathematics and Computing Science, Eindhoven University of Technology, the Netherlands
Formality Works

Abstract: Formal calculations can contribute successfully to a better understanding of the structure of a proof, both in programming and in "classical" mathematics. This will be illustrated by means of a few examples.

Speaker Background: Rob Hoogerwoord earned an Master of Science in Mathematical Engineering in October 1979 at Eindhoven University of Technology under the supervision of Professor Edsger W. Dijkstra. He earned his Ph.D. in December 1989 at Eindhoven University of Technology in the area of functional programming. From 1980 to 1982 Hoogerwoord was a researcher in Computing Science at EUT; from 1982 to 1984 he worked as a software engineer at Philips Electronics, and from 1984 to the present he has been an Assistant Professor in Computing Science at EUT. Hoogerwoord's main research area is (formal) techniques for designing programs and implementations—be it sequential, functional, parallel, or distributed.

Rajeev Joshi
Compaq Systems Research Center, U. S. A.

Panelist Background: Rajeev Joshi graduated in 1990 with a B. Tech in Computer Science from the Indian Institute of Technology in Bombay, India. From 1990 to 1992, he worked with the Math Sciences Research Group at AT&T Bell Laboratories in Murray Hill, New Jersey. In September 1992, he enrolled in the Department of Computer Sciences at The University of Texas at Austin, graduating with a Ph.D. in December 1999. Joshi is currently working at Compaq System Research Center in Palo Alto, California. His main research interest is in the design of effective and efficient techniques to reason formally about computer systems. In particular, he is interested in formalisms that are attractive both computationally (for use in automated verification), and also calculationally (for use by humans in designing programs and proofs "hand in hand").

Charanjit Singh Jutla
IBM T. J. Watson Research Center, U. S. A.

Panelist Background: Charanjit Singh Jutla received his Ph.D. in Computer Science from The University of Texas at Austin in 1990. Since then, he has been a Research Staff Member at the IBM T. J. Watson Research Center. Jutla's research interests include cryptography, security, and algorithms.

Anne Kaldewaij
School of Informatics, University of Amsterdam, the Netherlands

Panelist Background: Anne Kaldewaij received a Master of Science degree in Mathematics from the University of Utrecht, the Netherlands) and a Ph.D. degree in Computing Science from Eindhoven University of Technology. From 1982 until 1995 he worked as Associate Professor in Computing Science at the E. U. T., doing research in parallel programming and the design of algorithms and data structures. There he was brought up in the EWD style. Currently, Kaldewaij is Director of the School of Informatics at the University of Amsterdam. He still enjoys teaching and has written a number of textbooks on mathematics and programming.

Markus Kaltenbach
Siemens Corporate Technology, Germany

Panelist Background: Markus Kaltenbach received a Ph.D. in Computer Sciences from The University of Texas at Austin in 1996 and a Diplom degree (M.A.) in Computer Sciences from the Technical University of Karlsruhe, Germany, in 1991. Between 1984 and 1996 he held various appointments as a teaching and research assistant and worked for McKinsey & Company, Siemens, Motorola, and the Fraunhofer Institute. During that time, he also received several scholarships including a Fulbright Scholarship and an IBM Fellowship. Kaltenbach's research interests are related to the challenges faced in making formal verification techniques accessible and successful in an industrial context. He joined Siemens Corporate Technology in 1996, where he is now project manager in the department of Design Automation. He is responsible for the technological development of software verification.

Jacob Kornerup
National Instruments, U. S. A.

Panelist Background: Jacob Kornerup received a Master of Science in Computer Science degree from the University of Aarhus in Denmark in 1988. He was a graduate student at The University of Texas at Austin from 1988 to 1997, under the supervision of Jayadev Misra, working on the UNITY and Powerlist formalisms. He was a member of the Austin Tuesday Afternoon Club from 1990–1997. Kornerup joined the faculty at Southern Methodist University in Dallas as an Assistant Professor, after successfully defending his dissertation titled *Data Structures for Parallel Recursion*. In the summer of 1999, he joined National Instruments in Austin, Texas, as a Group Manager working on distributed systems for industrial automation.

Donald E. Knuth
Professor Emeritus of The Art of Computer Programming, Stanford University, U. S. A.
Structured Programming and Literate Programming

Abstract: Professor Knuth will reflect on the profound influence of Edsger W. Dijkstra's writings on all the computer programs Knuth has written since about 1970.

Speaker Background: Professor Knuth met Professor Dijkstra in the 1960s when both were affiliated with Burroughs Corporation. Although he and Edsger have never written a joint paper, the literature contains at least one citation of an imagined work by a "Professor Knijstra." Over the years they have enjoyed many hours together discussing Computer Science and numerous other things; sometimes they even play four-hands piano music together, provided that nobody but Ria is listening.

K. Rustan M. Leino
Compaq Systems Research Center, U. S. A.

Panelist Background; Rustan Leino is a computer science researcher at Compaq Systems Research Center. Most of his work is in programming languages, systems, and semantics. He is currently leading the Extended Static Checking for the Java Project, which has produced a practical program checker based on formal methods. Leino received his Ph.D. (1995) and Masters (1993) in Computer Science from California Institute of Technology. Before entering the graduate program in 1991, he was a technical lead in the Windows/NT group at Microsoft. In 1989, he received his Bachelors with Special Honors in Computer Science from The University of Texas at Austin, which is where he first met Professor Edsger W. Dijkstra.

Panagiotis Manolios
Department of Computer Sciences, The University of Texas at Austin, U. S. A.

Panelist Background; Panagiotis Manolios was born in Athens, Greece in 1967. He received a B.A. and an M.A. from Brooklyn College. Currently he is a Ph.D. student at The University of Texas at Austin. His interests include formal methods, with a focus on theorem-proving, model-checking, and abstraction.

Alain J. Martin
Computer Science Department, California Institute of Technology, U. S. A.
What I Learned From VLSI: An Imaginary Return to the Tuesday Afternoon Club

Abstract: In this talk, I imagine returning to the Tuesday Afternoon Club that I left 19 years ago, and describing to its members my long journey in the strange and faraway land of asynchronous VLSI. When I embarked on this new area of research, I decided to approach VLSI design as a concurrent programming problem. Many years and several million transistors later, what did I learn and how successful was this approach? I will briefly address a number of concurrency issues that the study of VLSI has either modified, like the notion of atomicity and the semantics of communications, or revealed, like slack elasticity and pipeline dynamics. I will suggest that we need a new model of computational complexity that includes both energy and time: the "E*t^2" measure may be a candidate. Finally, I will revisit our belief in correctness by construction after several "quasi real-world" experiments.

Speaker Background: Alain J. Martin is a Professor of Computer Science at the California Institute of Technology. He is a graduate of the Institut National Polytechnique de Grenoble and a founding member of the Tuesday Afternoon Club.

M. Douglas McIlroy
Dartmouth College, U. S. A.
The Music of Streams

Abstract: Data streams make preeminent instruments for playing certain classical themes from analysis. Complex networks of processes, effortlessly orchestrated by lazy evaluation, can enumerate terms of formal power series ad infinitum. Expressed in a language like Haskell, working programs for power-series operations are tiny gems, because the natural programming style for data streams fits the mathematics so well—often better than time-honored summation notation. Applications to Taylor series, generating functions, and solutions of differential equations are immediate and perspicuous.

Speaker Background: Doug McIlroy retired to blissful adjuncthood at Dartmouth College after a long career in programming research at Bell Labs. There he had headed the department in which UNIX was born, and promoted literacy via the UNIX program "spell." Over the years he has played with (a.k.a. researched) systems, languages, text processing, synthetic speech, computer graphics as number theory, documentation, security, cartography, and sundry computing byways. Stream processing has been a recurrent theme: UNIX pipes are his monument. McIlroy treasures the memory of presenting the ACM Turing award to the "humble programmer," Edsger W. Dijkstra**.**

David A. Naumann
Department of Computer Science, Stevens Institute of Technology, U. S. A.

Panelist Background: David Naumann learned through his experience as a software engineer that the scientific foundation of the discipline is both inadequate and inadequately used in practice. His goal is to bridge the gaps between theoretical and methodological studies and the languages and tools used in practice. After completing his doctoral study at The University of Texas at Austin, Naumann joined the faculty of nearby Southwestern University where formal methods were being integrated throughout the undergraduate curriculum. He is writing an undergraduate textbook on data structures and data refinement based on this experience. In his present position, his research focuses on verification of software components, including tools for automated checking of security properties.

Greg Nelson
Compaq Systems Research Center, U. S. A.
Juno-2, a Constraint-Based Graphics System

Abstract: Dijkstra's calculus of predicate transformers is usually applied to the predicates of a first-order theory of integers and integer arrays, but it can be applied to the predicates of any first-order theory. What kind of programming language would we get if we applied it to other first-order theories, say, the theory of Euclidean Geometry? We would get Juno, a powerful constraint-based geometric language that will be described in this talk. The talk also describes an interactive editor for producing illustrations using Juno-2, the second version of Juno, which was implemented by the speaker together with Allan Heydon. The talk will include a demonstration of the editor (weather permitting).

Speaker Background: Greg Nelson received his Ph.D. in Computer Science from Stanford University in 1980, where he worked on program verification and algorithms for mechanical theorem-proving. He received his B. A. degree in Mathematics from Harvard College in 1976. He was a member of the committee that designed the programming language Modula-3, and was the author of the Juno constraint-based graphics system at the Computer Science Laboratory, Xerox Palo Alto Research Center. Nelson has taught computing at Princeton University, and is currently a Principal Member of the Technical Staff at Compaq's Systems Research Center.

Josyula R. Rao
IBM Thomas J. Watson Research Center, U. S. A.

Panelist Background: J. R. Rao manages the Internet Security and Technology group at IBM's Thomas J. Watson Research Center. He obtained his doctoral degree from The University of Texas at Austin in 1992. Rao's research interests include programming methodology, distributed systems, network security, and cryptography. He is a member of IFIP's Working Group 2.3 on Programming Methodology.

Martin Rem
Eindhoven University of Technology, the Netherlands

Panelist Background: Martin Rem studied mathematics at the University of Amsterdam. In 1976 he obtained a Ph.D. degree in Computer Science from Eindhoven University of Technology. Professor Edsger W. Dijkstra acted as his thesis advisor. Thereafter he worked for two years as an Assistant Professor in Computer Science at the California Institute of Technology in Pasadena, U. S. A.. Since 1978 Rem has been a Professor of Computer Science at Eindhoven University of Technology. During the academic year 1985–1986 he was a part-time guest professor at Catholic University of Leuven, Belgium. Since 1985 he has been a consultant to Philips Research. His research areas are parallel computing and asynchronous VLSI circuits. Rem is a member of the European Science and Technology Assembly. From 1991 to 1994 he was Dean of the Faculty of Mathematics and Computer Science and, since September 1996, he has been the Rector of Eindhoven University of Technology.

Hamilton Richards
Department of Computer Sciences, The University of Texas at Austin, U. S. A.

Banquet Speaker Background: Hamilton Richards started programming in the 1960's as a mechanical engineer in the aerospace industry. After a two-year stint in the Peace Corps, he continued programming at Fairchild Semiconductor, where he constructed the first operating system software for the Symbol Computer. He continued work on the Symbol, earning his Ph.D. in 1976 at Iowa State University. For the next 11 years he worked on dataflow and functional programming projects at various Burroughs Corporation laboratories. When the Austin lab closed in 1986, Richards came to The University of Texas at Austin to coordinate the symposia and publications of the UT Year of Programming. Since the YoP ended, he has been teaching undergraduate Computer Science courses.

Fred B. Schneider
Cornell University, U. S. A.
Are There Systems Principles or Only Systems Principals?

Abstract: The tradition has been to teach several courses that each focus on one or another form of computing system: computer hardware, operating systems, database systems, distributed systems, networks, etc. Yet there are recurring themes throughout. In failing to identify these, we have balkanized the field. And, as a result, our specialists are doomed to reinvent the same wheel in different settings. What is "systems" all about if not what our systems principals are building? And what principles might one discuss in a course devoted to systems principles? Experiences with Cornell's "systems principles" course will be discussed.

Speaker Background: Fred B. Schneider is a Professor at Cornell's Computer Science Department. He is author of two texts on programming methodology: *On Concurrent Programming* and (with D. Gries) *A Logical Approach to Discrete Math*. He chaired the recent National Research Council's study committee on information systems trustworthiness and edited its final report, "Trust in Cyberspace." Schneider is a fellow of ACM and ABSTRACTS, and he has been a Professor-at-Large at the University of Tromsö (Norway) since 1996. He is managing editor of *Distributed Computing* and serves on a number of other editorial boards. He holds three patents in the area of fault-tolerant systems design.

Natarajan Shankar
SRI International Computer Science Laboratory, U. S. A.

Panelist Background: Natarajan Shankar is a staff scientist at the SRI International Computer Science Laboratory where he enjoys reasoning about programs that reason. He graduated with a Bachelor of Technology in Electrical Engineering from the Indian Institute of Technology, Chennai, and a Ph.D. in Computer Sciences from The University of Texas at Austin under the supervision of Professors Robert Boyer and J Strother Moore. Prior to joining SRI in 1989, he worked as a research associate at Stanford University. Shankar is chairman of IFIP Working Group 2.3 on programming methodology.

Ambuj K. Singh
Department of Computer Science, University of California at Santa Barbara, U. S. A.

Panelist Background: Ambuj K. Singh is an Associate Professor in the Department of Computer Science at the University of California, Santa Barbara and is currently serving as the vice-chair of the department. He received his Bachelor of Technology with Honors in Computer Science and Engineering from Indian Institute of Technology, Kharagpur in 1982, M.S. in Computer Science from Iowa State University in 1984, and Ph.D. in Computer Science from The University of Texas at Austin in 1989. His research interests are in the areas of formal specification and verification, distributed systems, multimedia databases, and digital libraries. Singh has served on a number of program committees including ACM Symposium on Principles of Distributed Computing (PODC), IEEE International Parallel processing Symposium (IPPS), IEEE International Conference on Distributed Computing and Systems (ICDCS), the Workshop on Distributed Algorithms (WDAG), and IEEE International Conference on Multimedia and Expo (ICME).

W. M. Turski
Institute of Informatics, Warsaw University, Poland

Banquet Speaker Background: Professor Dr. W. M. Turski earned a Ph.D. in 1962, habilitation in 1966, and the scientific title of a Professor in 1972. He was the Director of the Institute of Informatics from 1977 until 1982 and held a joint position (1983–1987) at Warsaw University and Imperial College in London, Great Britain. He served as the Dean of the Faculty of Mathematics, Informatics and Mechanics from 1996–1999. Turski is a Foreign Member of the Royal Academy of Engineering, United Kingdom, and a Distinguished Fellow of British Computer Society. He is an Honorary Member of the Polish Informatics Society, of which he was the President (1981–1987) and Vice-President (1987–1993). He was the Programme Chairman of the IFIP World Computer Congress (1977) in Toronto, Canada. He is the Managing Editor of *Information Processing Letters* and an Editor of *Acta Informatica*. His main research interests include software engineering (specification methods, software process and its support technology), novel approaches to real-time systems and parallel computing, and programming for behaviors.

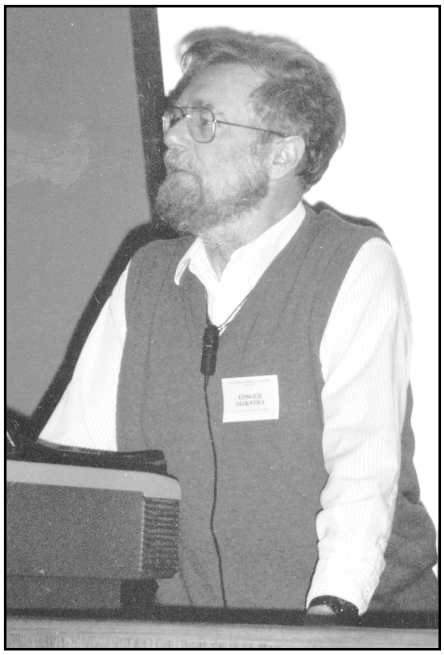
A. J. M. van Gasteren
Department of Mathematics and Computing Science, Eindhoven University of Technology, the Netherlands

Panelist Background: Netty van Gasteren has known Edsger W. Dijkstra since 1976, when she began studying for her master's degree in mathematics with him. In 1981 she joined Dijkstra's group as a BP Venture Research Fellow, investigating the orderly presentation and design of programs and proofs. She received her Ph.D. from Eindhoven University. Since then van Gasteren has taught and investigated programming methodology in various computing science departments in the Netherlands, getting more and more interested and involved in the use of calculational methods. A main driving force of her work is the quest for ways and methods to achieve clarity of thought and economy of expression.

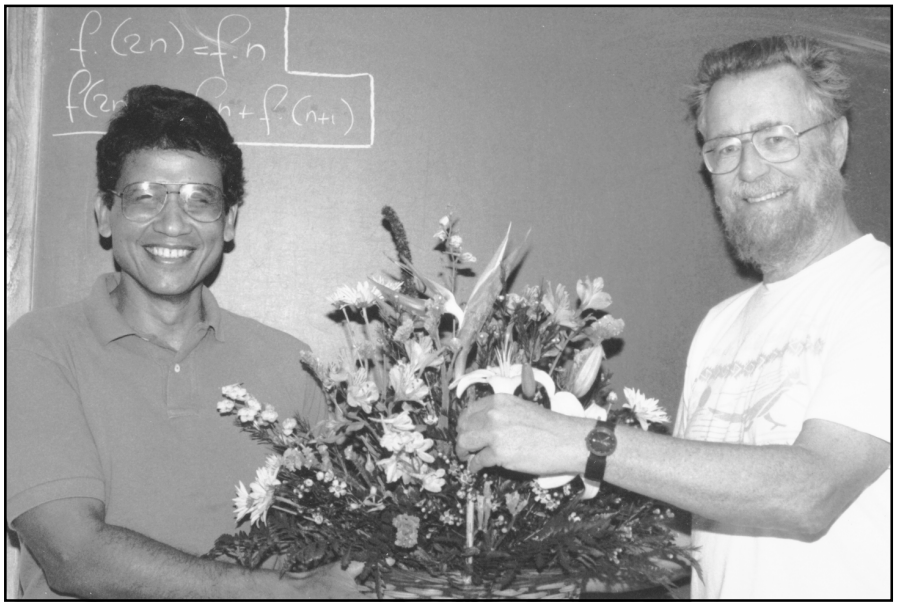
Niklaus Wirth
ETH Zentrum, Zurich, Switzerland
A Real-Time Application of Oberon-SA: Flight-Control of Model Helicopters

Abstract: Oberon is a small, structured, procedural programming language. Its conceptual simplicity and perspicuity result in compact implementation and efficient code. It is, therefore, particularly appropriate for real-time applications. As an example, we present the design of an airborne system for flight control of model helicopters, and we show how simplicity and compactness can be achieved in spite of (or because of) the high-level language. Oberon-SA is a large subset of Oberon implemented for the Strong-ARM processor.

Speaker Background: Niklaus Wirth has been a professor at the ETH Zurich from 1968 to 1999. He is the author of the programming languages Algol W, Pascal, Modula, and Oberon. He also codesigned the operating system Oberon and the personal computers Lilith and Ceres.



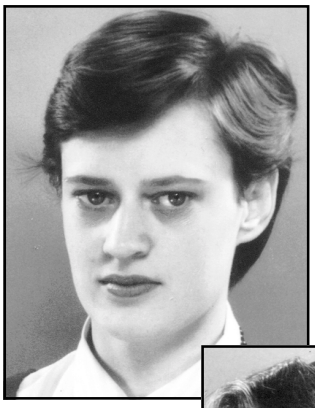
February 1994



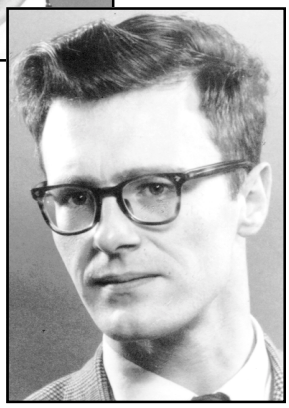
During the 1993–1994 academic year, the Edsger W. Dijkstra Scholarship was established at Eindhoven University of Technology. Professor Mohamed Gouda (left) and Professor Edsger W. Dijkstra (right).

1998

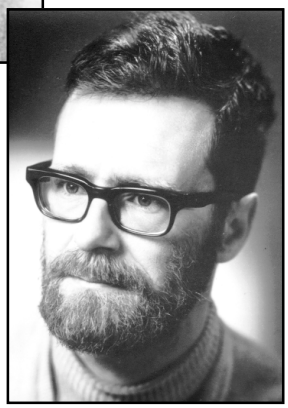
Fall 1997



1955



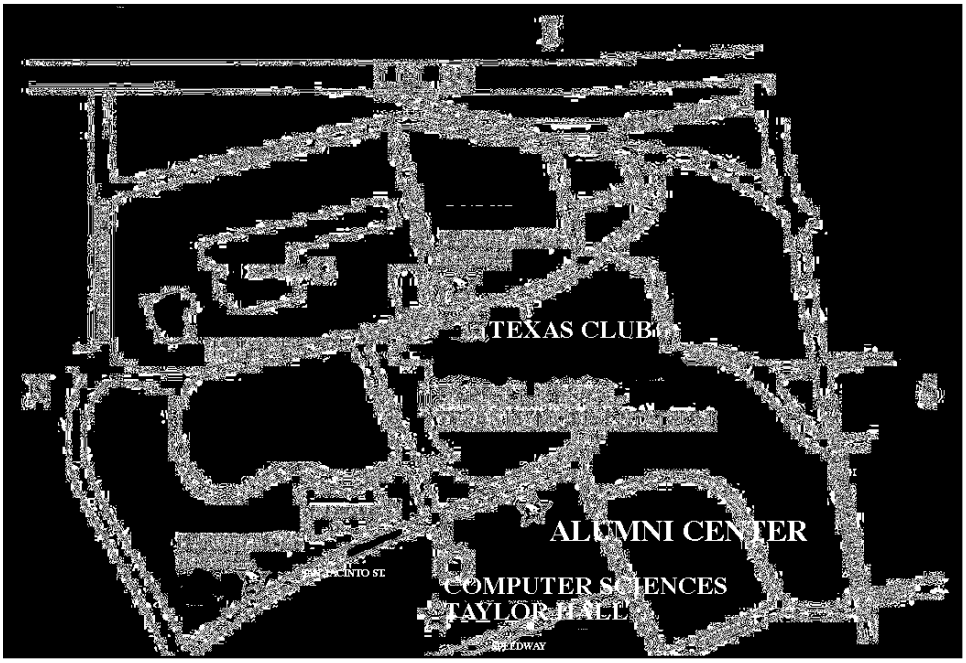
1970



M. C. Dijkstra-Debets

M. C. Dijkstra-Debets studied at R. K. Lyceum voor Meisjes, Reinier Vinkleskade, Amsterdam. She worked at the Mathematical Center in Amsterdam from 1949-1958. Edsger reports "That's where I found her." She began as a calculator with a desk machine and ended her tenure there as a programmer (in machine code). She and Edsger have now been happily married for 40 years.

May 1998



Map showing The University Texas Club, Taylor Hall, and the Lila B. Etter Alumni Center.