

*Office of Policy and Administration
ACM Headquarters
1515 Broadway
New York, NY 10036*

1 October 2002

*NOMINATION OF
DR. PETER NAUR*

*Professor Emeritus
Begoniavej 20
DK-2820 Gentofte
Denmark*

FOR THE ACM TURING AWARD

Submitted by

*Professor Per Brinch Hansen
2-175 Center for Science and Technology
Syracuse University
Syracuse, NY 13244
pbh@apollo.ecs.syr.edu (315) 446-4813*

ENCLOSURES

*Nomination Letter to the ACM Turing Award Committee Chair
Short Biography of Peter Naur
Scientific Contributions of Peter Naur
Supporting Letters for the Nomination of Peter Naur*

Professor Per Brinch Hansen
2-175 Center for Science and Technology
Syracuse University
Syracuse, New York 13244

1 October 2002

Robert Kahn
CNRI
1895 Preston White Drive, #100
Reston, VA 22091

Re: Nomination of Dr. Peter Naur for the ACM Turing Award

I am delighted to nominate Peter Naur for the ACM Turing Award,

for fundamental contributions to programming language definition, formal syntax notation, compiler construction, program assertions, structured programming, and computer education exemplified by pioneering work on the Algol 60 Report, the Backus-Naur Form, the Gier Algol compiler, "general snapshots," goto-less programming, "action clusters," and the Concise Survey of Computer Methods.

The nomination is strongly supported by letters from 27 leading computer scientists, including 13 Turing Award Winners and 10 Members of the National Academy of Engineering.

In support of the nomination I enclose the following documents:

- Short Biography of Peter Naur.
- Scientific Contributions of Peter Naur.
- Supporting Letters for the Nomination of Peter Naur.

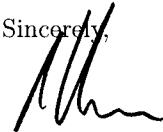
Peter Naur's contributions are now so familiar to everyone that it is difficult to appreciate just how revolutionary they were at the time. He was the driving intellectual force behind the definition of the programming language Algol 60. This has been called the "birth of computing science" because it "showed the first ways in which automatic computing could and should and did become a topic of academic concern" (Dijkstra 1999).

Eight computer pioneers, who participated in the development of Algol 60, or one its successors, Algol W, Simula 67, and Pascal, have already received the Turing Award:

1966	Alan Perlis	1980	Tony Hoare
1971	John McCarthy	1984	Niklaus Wirth
1972	Edsger Dijkstra	2001	Ole-Johan Dahl
1977	John Backus	2001	Kristen Nygaard

In my view, a Turing Award is long overdue for Peter Naur's towering contributions to computing.

Sincerely,



Per Brinch Hansen
Distinguished Professor

SHORT BIOGRAPHY OF PETER NAUR

Peter Naur was born in Frederiksberg near Copenhagen, Denmark, on October 25, 1928. While still in high school, he developed an interest in astronomy, and with the guidance of the staff of the Copenhagen University Observatory worked on calculations of the orbits of comets and minor planets. He began his academic studies at Copenhagen University in 1947, getting his mag. scient. degree in astronomy in 1949.

He spent the year 1950–51 as a research student at King's College, Cambridge, England, working on the design of a program for the EDSAC for calculating the perturbed motions of minor planets. The years 1952–53 he spent in the United States, with visits to a number of astronomical observatories and computer development laboratories, followed by a second stay at Cambridge, England.

From 1953 to 1959 he was a scientific assistant at Copenhagen Observatory, and also served as consultant in the areas of assembly language and debugging aids to the independent computer laboratory, Regnecentralen, on the development of the first Danish computer, DASK. He received his doctorate in astronomy in 1957.

In 1959 Dr. Naur joined the staff of Regnecentralen specializing in the area of high level languages, and thus became heavily involved in the international development of Algol 60. He organized the Algol Bulletin and became one of the thirteen members of the international team that produced the final design of Algol 60 in 1960, serving as editor of the team's report on the language, *Report on the Algorithmic Language Algol 60*.

As a member of the compiler design group of Regnecentralen, Dr. Naur contributed to the design of compilers for Algol 60 and Cobol, characterized by their effective use of multipass techniques. This experience led to an interest in the basic principles of data processing, resulting in a book, *Concise Survey of Computer Methods*, published in 1974.

As the first Danish professor of datalogy at the University of Copenhagen, from 1969 to 1998, Peter Naur has studied program development as a human activity.

He has been co-editor of *BIT*, the Nordic Journal for Numerical Mathematics and Computer Science, since its start in 1960. Dr. Naur served as president of The Danish Society of Datalogy until 1982, since its founding in 1966, and was co-editor of the report on the first conference on Software Engineering in 1968.

He was honored with the G. A. Hagemann Medal in 1963, the Jens Rosenkjær Prize in 1966, and the Computer Pioneer Award of the IEEE Computer Society in 1986.

Adapted from the IEEE brochure describing
Peter Naur's 1986 Computer Pioneer Award

SCIENTIFIC CONTRIBUTIONS OF PETER NAUR

This summary of Peter Naur's work relies primarily on quotations from the published papers of Turing Award winners, who participated in the development of Algol 60 and its successors.

The Algol 60 Language

Looking back on half a century of computer science, *Edsger Dijkstra* (1999) remarked that "The 60s started with an absolute miracle, viz. Algol 60."

The programming language Algol 60 introduced recursive procedures, block structure, scope rules, and type declarations in imperative programming. It was developed by an international committee that included John Backus, John McCarthy, Peter Naur, and Alan Perlis.

According to *Peter Wegner* (1976):

In programming language development, Algol 60 is probably the most important conceptual milestone. Its defining document, known as the Algol 60 report (Naur 1960), presents a method of language definition which is an enormous advantage over previous definition techniques and allows us for the first time to think of a language as an object of study rather than as a tool in problem solution.

Tony Hoare (1974) agreed:

The more I ponder the principles of language design, and the techniques that put them into practice, the more is my amazement at and admiration of Algol 60. Here is a language so far ahead of its time that it was not only an improvement of its predecessors but also of nearly all its successors.

The Algol 60 Report

In 1959, at the initiative of Peter Naur, the *Algol Bulletin* was issued which served as an international forum for discussing the development of the language. In January 1960, Naur prepared a draft organized in the style of the final Algol report.

Dijkstra (1999): "Several friends of mine, when asked to suggest a birth for Computing Science, came up with January 1960, precisely because it was Algol 60 that showed the first ways in which automatic computing could and should and did become a topic of academic concern."

John Backus (1981) acknowledged Naur as the driving intellectual force behind the definition of the programming language Algol 60:

Peter Naur's conduct of the Algol Bulletin and his incredible preparation for that [January 1960] meeting in which Algol was all written down already in his notebook—he changed it a little bit in accordance with the wishes of the committee, but it was that stuff that really made Algol 60 the language that it is, and it wouldn't have even come about, I don't think, had he not done that.

On account of this work, the committee asked Naur to be the editor of the *Algol 60 report* (Naur 1960).

Dijkstra (1999): “I remember, when I received the final document, the awe with which I observed how in the block structure the textual concept of scope had been related to the nested life time of incarnations. It was a beautiful synthesis, relating computation structure to program structure; as such it represents a quantum leap in our coming to grips with the programming task.”

In a summary of Naur’s report, *Tony Hoare* (1974) praised “the simplicity and clarity of its description, rarely equaled and never surpassed.”

The Backus-Naur Form

The syntax notation introduced by John Backus (1959) and refined by Peter Naur (1960) was a major influence on the Algol 60 Report. As *Backus* recalled (1980):

I had been exposed to the work of the logician Emil Post (1943) and his notion of a “production.” I hastily adapted [Post productions] for use in describing the syntax of IAL [Algol 58]. The resulting paper (Backus 1959) was received with a silence that made it seem that precise syntax description was an idea whose time had not yet come. As far as I know that paper had only one reader, Peter Naur. . .He improved the notation. . .and then used it to describe the syntax of Algol 60 in the definitive paper on that language.

Donald Knuth (1964) agreed:

Naur’s additions [to the original Backus Normal form] are quite important. Furthermore, if it had not been for Naur’s work in recognizing the potential of Backus’s ideas. . . Backus’s work would have become virtually lost; and much of the knowledge we have today about languages and compilers would not have been acquired.

Knuth proposed “that henceforth we say Backus Naur Form instead of Backus Normal form, when referring to such a syntax.”

The evidence is overwhelming that Peter Naur’s Algol 60 report is a landmark in computer science:

Donald Knuth (1967): “The most notable feature of the first Algol 60 Report was the new standard it set for language definition, based on an almost completely systematic use of syntactic rules that prescribed the structure of programs; this innovation made it possible to know exactly what the language Algol 60 was, to a much greater extent than had ever been achieved previously.”

Edsger Dijkstra (1999): “The introduction and use of BNF (Backus Naur Form) to formalize the syntax was a very courageous novelty. . .It turned out to have all the properties of a helpful formalism, viz. compact, unambiguous and amenable to mechanical manipulation: before the end of the decade the construction of parsers had been mechanized, an achievement that 10 years earlier would have baffled the imagination.”

Bob Floyd (1964): “The most representative and fruitful example of the use of a formal grammar in defining a programming language is the use of a phrase structure grammar to specify most of the syntactic rules of Algol 60.”

Edsger Dijkstra (1972): “The famous Report on the Algorithmic Language Algol 60 is the fruit of a genuine effort to carry abstraction a vital step further and to define a programming language in an implementation-independent way. . .The report gloriously demonstrated the power of the formal method BNF, now fairly known as Backus-Naur-Form, and the power of carefully phrased English, at least when used by someone as brilliant as Peter Naur. I think that it is fair to say that only very few documents as short as this have had an equally profound influence on the computing community.”

Compiler Construction

The famous *Gier Algol compiler* of Jørn Jensen and Peter Naur implemented Algol 60 on the Gier computer with a memory of only 1K words and a drum of 12K words (Naur 1963b). Inspired by the Atlas computer (Fotheringham 1961), the run-time system implemented demand paging of compiled code—without any hardware support!

Gier Algol introduced several innovations in compiler technology. During compilation, syntactic and semantic errors were handled by the same kind of logic that was used for correct source programs. Type checking was performed by abstract evaluation of expressions operating on type descriptions rather than values of operands (Naur 1965). The compiler was checked by means of small Algol programs constructed specifically to ensure that each instruction of the compiler would be executed at least once. This systematic approach to testing made the compiler virtually error-free.

Dijkstra correctly called Gier Algol “a masterpiece” (Brinch Hansen 1976).

Program Correctness

Peter Naur (1966b) also pioneered the idea of using *assertions* to prove the correctness of programs. According to *Dijkstra* (1999):

Peter Naur from Copenhagen was around 1965 the first to contribute to the art of reasoning about programs in the little article in which he launched his “general snapshots” (which later would be called “assertions”). Due to the limited circulation of the journal BIT in which it was published it did not get the attention it deserved. He was followed by Robert W. Floyd (1967) from Stanford with the paper on assigning meanings to programs.

Structured Programming

In a short paper on *Go to statements and good Algol style*, Naur (1963c) wrote:

I would like to turn the attention to a feature which mars a large number of the published Algol programs: the incessant and unnecessary use of the goto statement. . .If you look carefully you will find that surprisingly often a “go to” statement which looks

back really is a concealed “for” statement. And you will be pleased to find how the clarity of the algorithm improves when you insert the “for” clause where it belongs.

As *Donald Knuth* (1974) pointed out: “Naur’s (1963c) comments were the first published remarks about harmful go to statements.” They appeared five years before the more well-known letter of *Dijkstra* (1968) on “Go to statements considered harmful.”

In a paper on *Programming by action clusters*, Naur (1969a) illustrates systematic construction of programs from smaller pieces that maintain global invariants.

Computer Science Education

In a brilliant paper, Naur (1966a) views compilation as a general data processing problem that involves more fundamental programming methods which should be taught as part of a core of computer science. At a time when compiler construction was still regarded as a fundamental subject in its own right, Naur’s insight was ahead of its time.

Naur (1968, 1974) proceeded to outline a complete core course on computer science based on fundamental principles of data representation, arithmetic, searching and sorting, finite state analysis of text strings, expression evaluation, linked lists, backing stores, and design of large data systems. Naur’s vision of computer science was published in the same year as *Donald Knuth’s* first volume on *The Art of Computer Programming* (1968).

The Software Crisis

In 1969 Peter Naur and Brian Randell edited a report on the first conference on Software Engineering. *Dijkstra* (1999) viewed this as a turning point in the history of computer programming:

It was there and then that the so-called “Software Crisis” was admitted and the condition was created under which programming as such could become a topic of academic interest. The latter, not surprisingly, turned programming from an intuitive activity into a formal one.

Human Aspects of Computing

Many of Peter Naur’s papers first appeared in the little known Scandinavian journal BIT. In 1992, ACM Press published a collection of Peter Naur’s most important papers. This volume also includes some of his later papers from the 1970s through the 1990s. Although not as influential as his groundbreaking work in the 1960s, his essays on *Computing: A Human Activity* show great wisdom and a refreshingly undogmatic view of the field.

References

- Backus, J.W. 1959. The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference. *International Conference on Information Processing*, (June), UNESCO, Paris, France, 125–132.
- Backus, J.W. 1980, Programming in America in the 1950s. In N. Metropolis, J. Howlett and G.-C. Rota (eds.), *A History of Computing in the Twentieth Century*, 125–135.

- Backus, J.W. 1981. Question and answer session on Algol 60. In R.L. Wexelblat (ed.), *History of Programming Languages*. Academic Press, New York, 162.
- Brinch Hansen, P. 1976. The programmer as a young dog. English translation in *The Search for Simplicity: Essays in Parallel Programming*, P. Brinch Hansen (ed.), IEEE Computer Society Press, Los Alamitos, CA, 1996, 142–146.
- Dijkstra, E.W. 1968. Go to statements considered harmful. *Communications of the ACM* 11, 3 (March), 147–148.
- Dijkstra, E.W. 1972. The humble programmer. *Communications of the ACM* 15, 10 (October), 859–866.
- Dijkstra, E.W. 1999. Computing science: achievements and challenges, (March), *ACM Symposium on Applied Computing*, San Antonio, TX.
- Floyd, R.W. 1964. The syntax of programming languages—a survey. *IEEE Transactions on Electronic Computers* 13, (August), 346–353.
- Floyd, R.W. 1967. Assigning meaning to programs. Mathematical Aspects of Computer Science. In J.T. Schwartz (ed.), *Symposia in Applied Mathematics 19*, American Mathematical Society, Providence, RI, 19–32.
- Fotheringham, J. 1961. Dynamic storage in the Atlas computer, including an automatic use of a backing store. *Communications of the ACM* 4, 10 (October), 435–436.
- Hoare, C.A.R. 1974. Hints on programming language design. In *Computer Systems Reliability, State of the Art Report 20*, C. Bunyan (ed.), Pergamon.
- Knuth, D.E. 1964. Backus Normal Form versus Backus Naur Form. *Communications of the ACM* 7, 12 (December), 735–736.
- Knuth, D.E. 1967. The remaining trouble spots in Algol 60. *Communications of the ACM* 10, 10 (October), 611–618.
- Knuth, D.E. 1968. *The Art of Computer Programming. Vol. 1. Fundamental Algorithms*. Addison-Wesley, Reading, MA.
- Knuth, D.E. 1974. Structured programming with go to statements. *ACM Computing Surveys* 6, 4 (December), 261–301.
- Naur, P. (ed.), Backus, J.W., Bauer, F.L., Green, J., Katz, C., McCarthy, J., Perlis, A.J., Rutishauer, H., Samelson, K., Vauquois, B., Wegstein, J.H., van Wijngaarden, A., and Woodger, M. 1960. Report on the algorithmic language Algol 60. *Communications of the ACM* 3, 5 (May), 299–314.
- Naur, P. (ed.), Backus, J.W., Bauer, F.L., Green, J., Katz, C., McCarthy, J., Perlis, A.J., Rutishauer, H., Samelson, K., Vauquois, B., Wegstein, J.H., van Wijngaarden, A., and Woodger, M. 1963a. Revised report on the algorithmic language Algol 60. *Communications of the ACM* 6, 1 (January), 1–17.
- Naur, P. 1963b. The design of the Gier Algol compiler. *BIT* 3, 2–3, 124–140 & 145–66. Also in *Annual Review in Automatic Programming* 4, Pergamon Press, New York, 1964, 49–85.
- Naur, P. 1963c. Go to statements and good Algol style. *BIT* 3, 3, 204–208.
- Naur, P. 1965. Checking of operand types in Algol compilers. *BIT* 5, 3, 151–163.
- Naur, P. 1966a. Program translation viewed as a general data processing problem. *Communications of the ACM* 9, 176–179.
- Naur, P. 1966b. Proof of algorithms by general snapshots. *BIT* 6, 4, 310–316.
- Naur, P. 1968. Datalogy, the science of data and data processes and its place in education. *IFIP Congress 68*, vol. II, 1383–1387.
- Naur, P. 1969a. Programming by action clusters. *BIT* 9, 3, 250–258.
- Naur, P., and Randell, B. (eds.) 1969b. *Software Engineering*. Nato Science Committee, (October), Brussels, Belgium.
- Naur, P. 1974. *Concise Survey of Computer Methods*. Petrocelli/Charter, New York.
- Naur, P. 1978. The European side of the last phase of the development of Algol 60. ACM History of Programming Languages Conference, *SIGPLAN Notices* 13, 8 (August), 15–44. Also in *History of Programming Languages*, R.L. Wexelblat (ed.), Academic Press, New York, 1981, 92–139.
- Naur, P. 1992. *Computing: A Human Activity*. ACM Press, New York.
- Perlis, A.J. 1978. The American side of the development of Algol 60. ACM History of Programming Languages Conference, *SIGPLAN Notices* 13, 8 (August), 3–14. Also in *History of Programming Languages*, R.L. Wexelblat (ed.), Academic Press, New York, 1981, 75–91.
- Post, E.L. 1943. Formal reductions of the general combinatorial decision problem. *American Journal of Mathematics* 65, 197–215.
- Sveinsdottir, E., and Frøkjær, E. (eds.) 1988. *Bibliography of Peter Naur*. Institute of Datalogy, University of Copenhagen, Denmark.
- Wegner, P. 1976. Programming languages—the first 25 years. *IEEE Transactions on Computers*, (December), 1207–1225.