Science and Engineering in Software Development
A Recognition of Harlan D. Mills' Legacy

Los Angeles, California
May 18, 1999

Sponsored by
IEEE Computer Society
ACM/SIGSOFT
Proceedings

Science and Engineering for Software Development: A Recognition of Harlan D. Mills’ Legacy
Proceedings

Science and Engineering for Software Development: A Recognition of Harlan D. Mills' Legacy

Los Angeles, California
May 18, 1999

Sponsored by
IEEE Computer Society
ACM/SIGSOFT

Co-Located With
ICSE'99

IEEE COMPUTER SOCIETY
Los Alamitos, California
Washington • Brussels • Tokyo
# Table of Contents

**Science and Engineering for Software Development: A Recognition of Harlan D. Mills' Legacy (SESD'99)**

Message from the Organizers ................................................................. vi
Program Committee Members ............................................................... vii
The Legacy of Harlan D. Mills ............................................................... viii
Guest Editor's Introduction ................................................................. x
1999 Harlan D. Mills “Practical Visionary” Prize .................................... xii
A Note on the Cover Graphic ............................................................... xiv

**Invited Talk**

Building on Rock: From Mills' Program Functions to Tabular Notation ................................................................. 2
*Professor David Lorge Parnas, McMaster University*

**Accepted Papers**

Coupling and Strength, a la Harlan D. Mills ...................................... 4
* N. Chapin

Developing Black Box Specifications through Sequence Enumeration ........................................................................... 14
* S.J. Prowell

Partition Testing with Usage Models .................................................. 24
* K. Sayre and J.H. Poore

Applying a Generalization of a Theorem of Mills to Generalized Looping Structures ...................................................... 31
* F. Tchier and J. Deharnais

**Special Presentation**

Application of Statistical Science to Testing and Evaluating Software Intensive Systems ................................................... 40
* J.H. Poore and C.J. Trammell
MESSAGE FROM THE ORGANIZERS


Those of us who were lucky enough to know Harlan have come away from the experience impressed with his vast contributions and their importance to our field, his great humanity and, as well, his characteristic humility. It is this third factor that makes it a bit complicated to celebrate the other two. In order to create a tribute that Harlan would have found pleasing, we have decided to concentrate not on his own accomplishment, but rather on the work of those who have applied and extended it. Hence this Colloquium. It focuses on a new generation of potential Harlans, on their work and on the insights that work brings to the discipline of software engineering.

In establishing this Colloquium, we are indebted for the support of the IEEE, the IEEE Computer Society, as well as our sponsors:

- Cleanroom Software Engineering, Inc., Boulder, CO
- Fraunhofer Center – Maryland, College Park, MD
- Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Germany
- Institute for Software Research, Fairmont, WV
- Q-Labs Inc.

We are particularly indebted to Q-Labs Inc. for underwriting the Harlan D. Mills “Practical Visionary” Prize. Q-Labs has been kind enough to fund the prize for its first year and to contribute the first half of the required funds to endow the Mills Prize in perpetuity.

Our thanks as well to all those who contributed to the Colloquium as authors, referees, prize nominators and prize committee members, and to our Proceedings Manager, Regina Sipple and our Guest Editor, Mike Dyer.

On Behalf of the Program Committee:

Vic Basili
College Park, Maryland

Tom DeMarco
Camden, Maine

Ali Mili
Fairmont, West Virginia
PROGRAM COMMITTEE MEMBERS

- V. Basili, University of Maryland
- S. Becker, American University
- T. DeMarco, The Atlantic Systems Guild
- M. Deck, Cleanroom Software Engineering
- J. Desharnais, Laval University, Canada
- M. Dyer, Dycon Systems
- S. Gerhart, Research Outlet and Integration
- P. Hausler, IBM
- A. Hevner, University of South Florida
- G. Hoffnagle, IBM
- R. Linger, Software Engineering Institute
- A. Mili, Institute for Software Research
- D. Parnas, McMaster University, Canada
- J. Poore, University of Tennessee
- S. Prowell, Q-Labs, Inc.
- D. Rombach, Fraunhofer Institute for Experimental Software Engineering, Germany
- C. Trammell, University of Tennessee
- J. Whittaker, Florida Institute of Technology
- M. Zelkowitz, University of Maryland
THE LEGACY OF HARLAN D. MILLS

This colloquium is organized in honor of Dr. Harlan D. Mills (1919-1996), and is intended to celebrate his legacy in the theory and practice of software engineering.

Harlan Mills left a huge vacuum when he died in 1996. Those of us who were touched in any way by this gentle man’s intellect, humor and philosophy, are painfully aware of that vacuum. Our industry has lost one of its iconic figures.

To create a fitting memorial to Harlan, we turn our attention (as he would wish) to some of the themes that occupied him during his life. Even though it is a commemorative event, the colloquium is not just a reminiscence on one man’s work. Rather, it is focused on the extension of that work to today’s state of the art. It will underscore the relevance of his ideas to modern challenges in research and practice.

Some of the most recurrently noted themes in the work of Harlan Mills are the following:

1. Intellectual Control: Harlan recognized that much of the problem of software development stems from lack of intellectual control over the software development process. We cannot achieve quality in software unless we learn to develop software under perfect intellectual control. While conventional wisdom provides that we learn to develop software on small examples so as to graduate to larger examples, the premise of intellectual control provides that we learn on small examples in order ... not to do large examples.

2. Relentless Pursuit of Simplicity: Harlan recognized the importance of simplicity as the key feature of a design; many of his prescriptions are geared towards encouraging simple designs. Software design can in fact be viewed as the identification of simple (but not necessarily intuitive) principles behind seemingly complex behaviors.

3. Emphasis on Quality: In an industry best known for delivering faulty products, Harlan advocated a set of prescriptions dedicated to the design and implementation of products that are provably free of defect. The ways of writing correct programs and knowing are at the heart of his approach. Practice shows that one can achieve quality and reduce overall lifecycle costs at the same time; what seems to happen with quality focused processes is that developers spend more effort on design and much less on testing and certification.

4. Multiplicity of Means: Perhaps the most important success factor in Harlan’s approach to software development is the deployment of a multiplicity of means. This is dictated by the multi-faceted nature of software products and software processes. The Mills multiplicity of means is applied across a wide range of issues, including: combining technical with managerial standards in developing software products; combining static and dynamic methods in achieving product quality; combining scientific principles with engineering
methods to define software processes; combining formal semantics and understandability in dealing with program correctness.

The premises laid out above have been applied to the elemental activities of software development to produce great gains in product quality and process productivity.

- **Specification**: Specifications are represented by black box structures, which map stimulus histories to outputs [3].
- **Design**: Design is carried out by mapping black boxes into state boxes then clear boxes, which are increasingly detailed design representations [3].
- **Verification**: Methods of correctness verification are deployed at each step of the transformation process. Also, using Mills' Clean Room methods [2, 1], verification-based inspections substitute for traditional unit testing.
- **Testing**: Under the Clean Room technique, testing is used exclusively for certification and reliability estimation. Functional (black box) testing is applied, where input data is generated randomly to simulate the statistical usage patterns of the software product.
- **Re-engineering**: With such contributions as structure theorem transformations and design abstraction and documentation, Harlan provided the basis for a systematic approach to software reengineering.
- **Team Organizations**: Harlan's concepts of Chief Programmer Teams and Clean Room organizations offered original solutions to the problem of organizing communication channels among members of a programming team. His team organizations are characterized by low communication overhead, narrow communication bandwidth, and clearly distributed responsibilities and prerogatives.
- **Education**: To illustrate that his ideas on software engineering education are effective even though they are counter-intuitive, Harlan used two recurrent examples: learning how to swim and learning how to type. These two endeavors are examples of how the most natural way to learn is a very poor option.
- **Professional Standards**: Harlan attempted to define professional standards for the software industry, to parallel the rigor and the maturity of more traditional engineering disciplines.

**REFERENCES**


This volume contains the proceedings of the colloquium, "Science and Engineering in Software Development: a Recognition of Harlan D. Mills' Legacy", held in Los Angeles, California on May 18, 1999. The one-day colloquium acknowledged the contributions of Dr. Harlan Mills to the theory and practice of software engineering and their widespread application to the emerging challenges in software engineering.

Dr. Mills' contributions to software engineering were numerous, but a few should receive special note. The most pervasive theme of his work was to devise and promulgate methodology that would support the intellectual control of software product development. Within the industrial environment, this was realized through the deployment of formal methods, the blending of best practice into the total software lifecycle, and the introduction of technical management standards focused on zero-defect product quality. To ensure that the software engineer would be equipped for this challenge, he devoted his energy to advancing education in software engineering, both in academia and industry. The proceedings attempt to capture these contributions in a short article recounting the legacy of Harlan Mills.

The goal of the 1999 colloquium was to bring together members of the software engineering community and share experiences on the application, adaptation, and extension of Mills' work in their professional careers. It is hoped that this will be the first of an annual event so that these contributions can continue to flourish for the betterment of the software industry. To that end, a special award, the Harlan D. Mills Practical Visionary Award, has been established to be presented to a person who has excelled at promoting and extending these ideas into the development workplace.

These proceedings contain the texts of five papers delivered at the colloquium. These papers were selected by referees from a larger body of submitted articles and were judged on their own qualities and for their relevance to Dr. Mills' ongoing contribution to software engineering.

The first paper, entitled "Coupling and Strength, a la Harlan Mills", discusses an underlying theme of Mill's work of emphasizing function over algorithm that naturally lead to the fostering of modularity in software. Building on this idea, the paper introduces the method of message tables for documenting the design of functionality and data flow among modules. The paper includes worked-out simple examples, noted that the approach has been successfully used from the design through testing of software, and discusses its applicability for defining improved measures of module coupling and strength.

The second paper, entitled "Developing Black Box Specification through Sequence Enumeration", discusses Mills' significant contribution to the management of requirements for software products, the box structure methodology. The "black box" specification can provide an arguably complete, consistent, and traceable specification for a software product that accurately represents the desired external system behavior. The paper provides a detailed discussion of sequence enumeration is a powerful technique for creating and managing such specifications.

The third paper, entitled "Partition Testing with Usage Models", discusses an extension of the software testing ideas promoted by Mills, namely the use of
statistical based testing methods that would allow the making of statistical inferences about software quality. The paper discusses the application to software testing of a fundamental strategy in statistics of improving sampling efficiency by partitioning the sampling population. Usage models that underlie much of the methodology in the statistical testing of software support many strategies for automated partitioning and for generating software test cases from a partitioned sampling population. Two specific strategies are discussed in this paper with a demonstration of the efficiency gains in sampling from each strategy.

The fourth paper, entitled “Applying a Generalization of a Theorem of Mills to Generalize Looping Structures”, describes some ongoing research into the mathematical underpinning of software structures that was key to Mills’ tenet that correct software could be routinely constructed. The research uses a generalization of a theorem by Mills, known as the while statement verification rule, and applies it to three forms of looping structures that have been proposed in the software literature. The paper reports on the results of the initial research that indicate some advantage to iterative forms of looping structures - an advantage that has been mostly ignored by the formal programming community.

The fifth and final paper, entitled “Application of Statistical Science to Testing and Evaluating Software Intensive Systems”, discusses how well statistical methods address the complexity of software testing and how they provide a mathematical underpinning to what was a heuristic based validation practice. The paper describes how statistics provides a structure for collecting data and transforming that data into information that can improve decision making under uncertainty. Statistical testing of software should not be viewed as just the use of randomly generated test cases, but rather as the comprehensive application of statistical science to the total problem of software testing. The paper discusses the roles that statistical methods can play in population characterization, sampling, inference, design of experiments, etc. and the benefits realized.

It is hoped that these proceedings will shed some light on the breadth and scope of the contribution that Harlan Mills made to the field of software engineering. It is also hoped that the proceedings reflect how Mills’ work continues to spawn further research and practice to establish software as an engineering science. As he was so often heard to remark, “when the science of geometry was just 25 years old, the ancient Egyptians hadn’t discovered the right triangle, so we have a ways to go in formulating a software science”.

In editing this volume, we have been helped by many others, and would like to take this opportunity to thank them all. Specifically, our thanks and appreciation go first to the contributors for their work in preparing and revising the manuscripts, then to the organizers of the colloquium, and finally to all who participated in the colloquium.

Mike Dyer
Dycon Systems
Bethesda, Maryland
THE HARLAN D. MILLS “PRACTICAL VISIONARY” PRIZE

The Harlan D. Mills “Practical Visionary” Prize is to be given each year to a contributor whose work best demonstrates and extends upon advances first made or proposed by Harlan Mills.

Dr. Mills identified several key technical and organizational barriers to improving software development: 1) too many (and often inexcusable) errors; 2) programmers' and managers' inability to predict and track technical progress due to lack of formal foundations; and 3) industrial and educational standards that were unnecessarily low, and practically and socially unacceptable

Mills — individually, and with colleagues present at this colloquium — devised specific and integrated responses to these problems. These responses include box structures, statistical testing, and team proof. And he wrapped them together into cleanroom framework that distinguished his blend of approaches and projected his difficult-to-reach, but still very practical, goals and standards. Each of these technical subjects has stimulated increasingly deep papers and identified many still open problems. The collective technical contribution of Harlan Mills can best be summed up as “awesome.”

Further, Dr. Mills demonstrated that his blend of approaches can and do work on serious software projects. As important, he inspired others outside his immediate organization, to try his methods, even when the difficulty of the challenge and his eclectic approach were scorned by some of the more narrowly focused leaders of the field.

As a person, Harlan Mills also went out of his way to help younger researchers and projected his courage and hard work as an example to do more than just publish and talk at conferences.

The "practical visionary" rubric for this award pulls together the key aspects of Mills work. He had the ability to see the technical barriers, the knowledge to come up with serious technical responses, the managerial savvy to design and execute technology transfer strategies, and the humanitarianism to reach out to individuals as people, whether supporters or detractors.

Recipients of this award, current and future, stand for the Mills goals of tackling the hard challenges of software production, applying personal forcefulness to project a vision of the nature of that challenge, and putting solid technical responses to disciplined field trials and experiments to demonstrate progress over years and decades.

The committee strongly appreciates the thoughtful nomination letters for this first award. We hope that the establishment of this award will focus attention on the important technical work of both Mills and the nominees/winners of the prize, and further develop and recognize those who are willing to shake up the software industry by leading through practical methods toward new visions.

Susan Gerhart
Applied Formal Methods Inc.
1999 Harlan D. Mills "Practical Visionary" Prize Award

The program committee of "Science and Engineering for Software: A Recognition of the Legacy of Harlan D. Mills" and the Mills Prize Award committee are pleased to announce that the winner of the 1999 Harlan D. Mills "Practical Visionary" Prize is:

David Lorge Parnas
McMaster University

David Lorge Parnas holds the NSERC/Bell Industrial Research Chair in Software Engineering in the McMaster University Faculty of Engineering's Computing and Software Department where he is Director of the Software Engineering Program.

He has been a Professor at the University of Victoria, the Technische Hochschule Darmstadt, the University of North Carolina at Chapel Hill, Carnegie Mellon University and the University of Maryland.

He has also held non-academic positions advising Philips Computer Industry (Apeldoorn), the United States Naval Research Laboratory in Washington, D.C. and the IBM Federal Systems Division. At NRL, he instigated the Software Cost Reduction (A-7) Project, which develops and applies software technology to aircraft weapon systems. He has advised the Atomic Energy Control Board of Canada on the use of safety-critical real-time software at the Darlington Nuclear Generation Station.

The author of more than 200 papers and reports, Dr. Parnas is interested in most aspects of computer system design.

In his teaching, as well as in his research, Dr. Parnas seeks to find a "middle road" between theory and practice, emphasizing theory that can be applied to improve the quality of our products.

Professor Parnas received his B.S., M.S. and Ph.D. in Electrical Engineering - Systems and Communications Sciences from Carnegie Mellon University, and honorary doctorates from the ETH in Zurich and the Catholic University of Louvain in Belgium. He won an ACM "Best Paper" Award in 1979, and two "Most Influential Paper" awards from the International Conference on Software Engineering. He was the 1998 winner of ACM SIGSOFT’s "Outstanding Research Award".

Dr. Parnas is a Fellow of the Royal Society of Canada and a Fellow of the Association for Computing Machinery (ACM). He is licensed as a Professional Engineer in the Province of Ontario.
A Note on the Cover Graphic

One of the most enduring lessons we learned from Harlan Mills is the notion that seemingly complex behaviors can be explained by simple first principles. An embodiment of this idea arises in software design: the best design is one that discovers and exploits the actual simplicity behind the seeming complexity. Many of the solutions that Dr. Mills proposed for software engineering problems were counter-intuitive but simple, by contrast with their more “common sense” solutions.

In mathematics, fractals epitomize complex designs that can be captured by simple mathematical formulas. Hence we chose to use a fractal graphic as the cover theme of for these proceedings.
Invited Talk