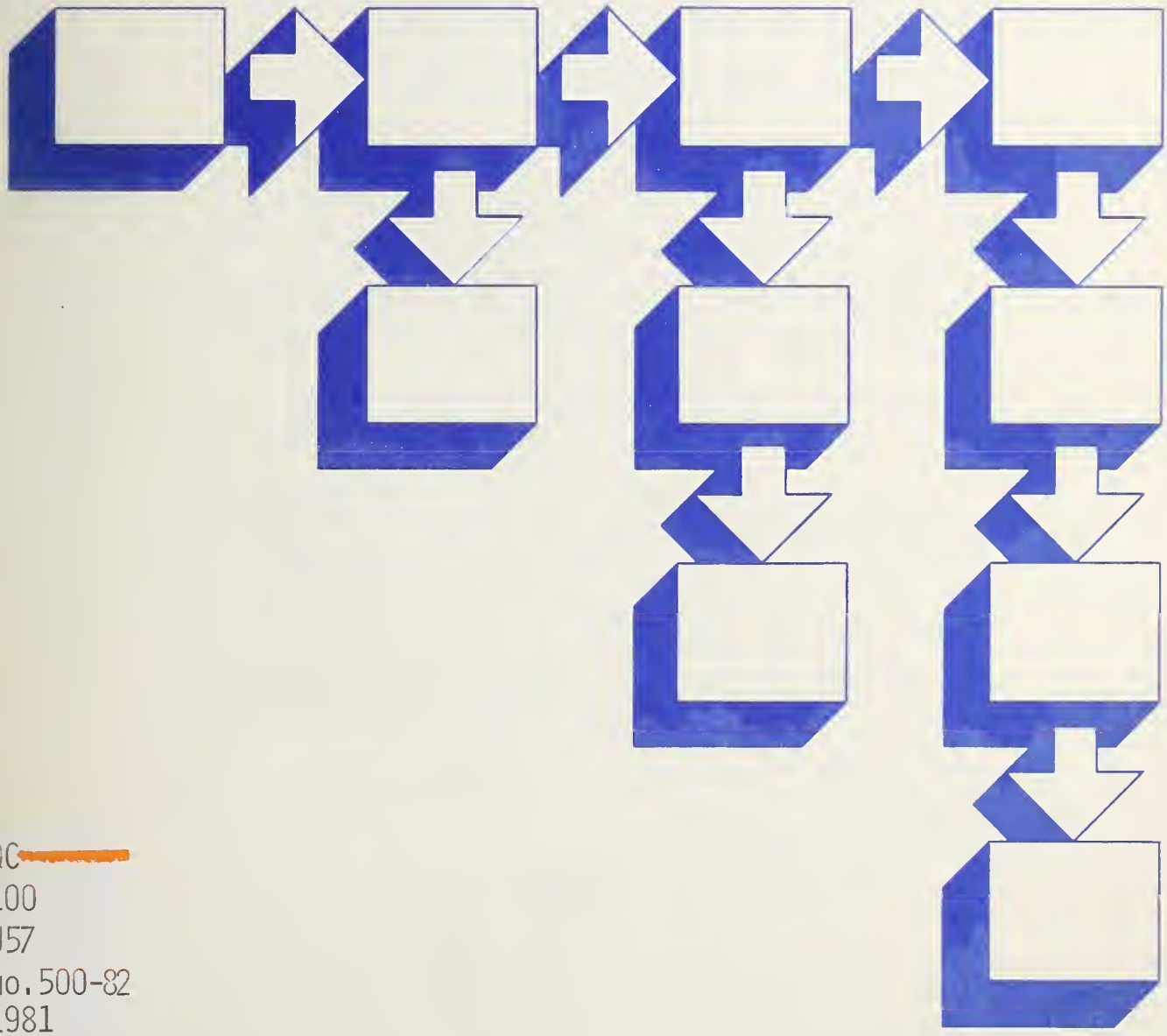
U.S. Department
of Commerce

National Bureau
of Standards

Computer Science and Technology

NBS Special Publication 500-82

Final Report: A Survey of Software Tools Usage



NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards[1] was established by an act of Congress on March 3, 1901. The Bureau's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the Nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, and the Institute for Computer Sciences and Technology.

THE NATIONAL MEASUREMENT LABORATORY provides the national system of physical and chemical and materials measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; conducts materials research leading to improved methods of measurement, standards, and data on the properties of materials needed by industry, commerce, educational institutions, and Government; provides advisory and research services to other Government agencies; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

Absolute Physical Quantities[2] — Radiation Research — Thermodynamics and Molecular Science — Analytical Chemistry — Materials Science.

THE NATIONAL ENGINEERING LABORATORY provides technology and technical services to the public and private sectors to address national needs and to solve national problems; conducts research in engineering and applied science in support of these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the ultimate user. The Laboratory consists of the following centers:

Applied Mathematics — Electronics and Electrical Engineering[2] — Mechanical Engineering and Process Technology[2] — Building Technology — Fire Research — Consumer Product Technology — Field Methods.

THE INSTITUTE FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides scientific and technical services to aid Federal agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following centers:

Programming Science and Technology — Computer Systems Engineering.

[1]Headquarters and Laboratories at Gaithersburg, MD, unless otherwise noted; mailing address Washington, DC 20234.
[2]Some divisions within the center are located at Boulder, CO 80303.

Computer Science and Technology

NBS Special Publication 500-82

Final Report: A Survey of Software Tools Usage

Herbert Hecht

SoHaR Incorporated
1040 So. La Jolla Avenue
Los Angeles, California 90035

U.S. DEPARTMENT OF COMMERCE
Malcolm Baldrige, Secretary

National Bureau of Standards
Ernest Ambler, Director

Issued November 1981

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

National Bureau of Standards Special Publication 500-82
Nat. Bur. Stand. (U.S.), Spec. Publ. 500-82, 58 pages (Nov. 1981)
CODEN: XNBSAV

Library of Congress Catalog Card Number: 81-600112

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1981

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, DC 20402
Price $4.25
(Add 25 percent for other than U.S. mailing)

TABLE OF CONTENTS

TABLE OF CONTENTS (CONT.)

LIST OF TABLES

SECTION 1

EXECUTIVE SUMMARY

1.1 BACKGROUND

Tools have a great potential for improving the quality of software, for increasing productivity in software development and maintenance, and for facilitating more efficient utilization of computer resources. All of these benefits are of particular significance to software operations in Federal agencies because of the critical nature of much of the programming effort and because of the tremendous volume of software generated and maintained for the Government.

A study of software tool usage was contracted by the National Bureau of Standards, Institute of Computer Sciences and Technology (NBS/ICST), as part of an effort to develop methodologies and standards for software quality control, which is one of the areas of responsibility of NBS/ICST under the Brooks Act. This report describes a survey of software tool usage and the analysis of the survey findings in terms of present and projected use of software tools.

1.2 EXTENT OF TOOL USAGE

The survey population consisted of 23 sites of which 8 were in the private sector, 5 were Government-support organizations, and 10 were Government agencies. The sites represent a wide spectrum of programming environments and size of software staff. Seven of the organizations were tool developers. Participation in the survey was voluntary [Section 3]*. Interviews based on a questionnaire were conducted at each site concerning software practices, tool usage, and specific tools in use. When it was found that different software practices were followed by individual organizations at a given site, each distinct type of environment was separately interviewed. This caused the number of interviews to be 29.

The extent of tool usage was classified into three categories:

Minimal - only tools normally supplied as part of an operating system are used.

Intermediate - special purpose tools to support the mission of the organization are in use.

General purpose - tools for general purpose software development and test are used.

---

* In this summary, square brackets, [ ], are used to direct the reader to sections and paragraphs of the report that expand on the material that is here presented in a very abbreviated manner.

The 29 interviews identified 3 minimal tool users, 7 intermediate tool users, and 19 general purpose tool users. The seven tool developers were all general purpose tool users, and because these were considered a distinct population, they have been censored from the following analysis of environmental effects.

To obtain a better insight into the overall status of tool usage, it is convenient to assign numerical (ordinal) indices to tool usage. Arbitrarily, 0 is equated to minimal tool usage, 1 to intermediate tool usage, and 2 to general purpose tool usage. The extent of tool usage in various environments is represented by these indices in Table 1 - 1.

TABLE 1 - 1 NUMERICAL INDICES OF TOOL USAGE

Software Staff	Organization Type		
	Private	Gov't Support	Government
Small (<15)	1	1	0
Medium (15-39)	1	1	0.7
Large (40-99)	-	1	1.5
Very large (>99)	2	2	2

These data indicate that the size of the software staff has a significant effect on tool usage. The type of organization did not seem to affect the extent of tool usage.

In terms of major tool functions, the feature most frequently encountered was Transformation (22 times), which was followed closely by Static Analysis (21 times), while Dynamic Analysis was utilized less frequently (13 times). (A typical tool that uses transformation is a preprocessor, a tool that uses static analysis is a code auditor, and a tool that uses dynamic analysis is a test analyzer [5.2].)

1.3 USER REACTIONS TO TOOLS

Most tools were regarded as useful and as eliminating tedious and repetitive operations. The tools that elicited the most favorable comments from the immediate users were editors, pre-processors, and (general purpose) development tools. Because most of them address a restricted area of software development and have few options, they are regarded as easy to use [5.3.1]. From the management point of view, libraries, data base managers, and performance analyzers received the most favorable comments. The reaction of programmers and first level supervision to this group was mixed. The use of these tools involved some effort on their part and the payoff was not always apparent to them [5.3.2].

For several tools used primarily by large and very large software organizations, difficulties were cited repeatedly by both the immediate user and management; e. g., the run-time expansion and restrictions on program size or structure inherent in the use of some test analyzers caused problems. They were still regarded as useful because the information which they furnished was impossible to obtain by other means. Requirements analyzers were sometimes characterized as difficult to use, but again the potential benefits were recognized [5.3.3].

A local manager made the decisions regarding tool acquisition in two-thirds of the organizations interviewed. In most of the others a headquarters manager was the final authority [6.1.2]. In only two organizations, both private, a deliberate cost-benefit study was made prior to tool procurement or in-house development. In many cases the procurement or development of a tool is based on a clearly identified need, and economic factors are not explicitly considered. Examples are pre-processors for structured programming, simulations, and management tools that generate reports required by a customer. In others the tool is perceived to present a better way of accomplishing a necessary task, and technical management authorizes its procurement out of discretionary funds. Program design languages and performance monitoring or optimization tools may be acquired out of this motivation [6.1.3].

The reliability of tools was generally rated high. Where there were any adverse comments they were related to failure to furnish adequate diagnostics on improper input [6.2.2]. Most of the organizations used a formal training program when tools were introduced. About one-quarter relied on informal training (reading of manuals, on-the-job training, etc.) exclusively; small organizations were heavily represented in this group [6.2.3]. Tool documentation was, in most cases, rated as good or adequate, but improvements were desired by several survey participants. One adverse comment related to failure to adequately document "all the special conditions" [6.2.3]. A large majority of the organizations contacted expected to acquire additional tools in the near future. The tools that were to be procured generally had greater capability than those currently in use [6.3.2].

1.4 PROJECTED TOOLS USAGE

The use of tools can be expected to increase due to two factors: current tools will become accepted in a broader sector of the software community (particularly if a "friendly interface" and better documentation can be provided) and new software engineering practices will create a demand for new tools. The evaluation of software quality metrics (which are being accepted as contractual requirements) can best be carried out by means of tools. Cost modeling and reliability modeling are other areas in which large amounts of data on software characteristics are required in order to validate or apply current research. Software tools will be required to support the extensive data collection. Further impetus for tool usage may arise from the adoption of naming conventions within large programs and data bases. Adherence to conventions and verification of uniqueness of names can best be carried out by software tools.

Increased interest at high levels of project and organizational management in gaining visibility and control of software development is expected to speed the

introduction of tools. The survey participants frequently commented that present software practices were fragmented, but that there was perceptible pressure, or already existing directives, toward greater uniformity [7.1.2]

Some obstacles to tool usage must also be recognized. Among these are lack of portability (many tools are tied to specific computers and operating systems) and the amount of training required for efficient use of some tools. A significant organizational obstacle is that many tools provide benefits that are outside the sphere of responsibility of the immediate user. Examples are requirements analyzers (the primary benefits of which are felt at project stages that are far in the future), test tools (which are applied by the development organization but benefit operations), and management report generators (which are of greatest use to levels of management that are not directly involved with the software operations). In all of these areas it is important to establish an appropriate motivation for the tool user [7.2].

SECTION 2

INTRODUCTION

This report is the second document generated for the National Bureau of Standards, Institute of Computer Sciences and Technology (NBS/ICST), under contract NB79SBCA0273 by SoHaR Incorporated. This report covers Work Assignment No. 2, Survey of the Development and Use of Software Tools in Industry. The overall scope of the Work Assignment is identified as follows:

> "Determine how different programming environments affect the use of and direct the development of tools for quality software. Report the current state of the art regarding the development and use of tools in each of the programming environments."

In Work Assignment No. 1 of this contract a Software Tool Taxonomy was generated [1], and major portions of this were incorporated in an NBS Special Publication [2]. This material was then utilized in the effort reported here to classify features of software tools encountered in the user environment. The nomenclature for software tool features throughout the present report corresponds to that in the NBS Special Publication. As a sequel to the survey of software tools, guidelines for the introduction of software tools into a programming environment will be generated under Work Assignment No. 3 of this contract.

The Software Tools Studies contract is one of a series of efforts undertaken by NBS/ICST in connection with its responsibilities under the Brooks Act (PL 89-306)which aims to aid Government agencies in improving the cost-effectiveness in the selection, acquisition, and utilization of automatic data processing resources. NBS efforts to satisfy its responsibilities under the Brooks Act include research in computer science and technology and the development of Federal Government-wide standards for data processing equipment, practices, and software. The software standards efforts comprise six families of standards, one of which deals with software quality control. Although it is recognized that software tools can aid in software quality control, NBS/ICST has concluded that there did not exist a clear body of techniques for making effective use of tools. The present and related efforts are intended to fill this gap.

The potential benefits of increased software tool usage by Federal agencies are described in a recent GAO report [3] which states:

> "During its work in automatic data processing, GAO found many areas where using better software tools and techniques might have saved a considerable amount of money and trouble. These include:

--Management control.

--User needs.

--Maintaining and modifying production computer programs...

--Software conversion..."

The purpose of this report is to describe the users and the use of software tools, to determine how the environment affects tool usage, and to identify factors which might lead to increased software tool usage in the future. In line with these primary objectives, the extent of tool use among several classes of software developers is studied, the features of tools encountered are analyzed, and the benefits, as well as difficulties experienced with current tools, are related.

Section 3 of the report describes the goals of the survey, how it was planned and conducted, and the selection of participants. Section 4 discusses the factors in the environment which might affect tool usage and introduces the classifications that are used in the remainder of the report. The central findings of the survey regarding present tool usage are contained in Section 5 which covers the extent of tool usage, the classification of the tools encountered, and user reaction to selected tools. Additional analysis of survey findings, primarily oriented towards future tool usage, is presented in Section 6. An interpretation of the findings in terms of factors that favor or inhibit tool usage and requirements for tools usage in the near future are presented in Section 7. Conclusions regarding the broad aspects of tool usage are summarized in Section 8.

Appendices A and B contain, respectively, the letter soliciting participation in the survey and the survey questionnaire. The findings from each interview are synopsized in a companion volume to this report [4].

Before leaving this introduction, the author wishes to acknowledge the contributions of collaborators in the survey effort. H. M. "Bob" Farmer conducted interviews and analyzed survey results, Myron Hecht analyzed survey results and formatted them for inclusion in this report, and Donald J. Reifer classified the tools encountered in the survey. Substantial assistance in conducting the survey, in contacting survey participants, and in providing background material, was rendered by the technical monitor for the tools studies, Mr. R. C. Houghton, Jr. Continued encouragement and many helpful suggestions were furnished by Dr. Martha Branstad and Dr. Selden L. Stewart, who supervised the effort.

SECTION 3

DESCRIPTION OF THE SURVEY

This section describes the goals of the survey, how it was planned and conducted, and the overall composition of survey participants. The following specific topics are covered in the order indicated:

Contractual requirements
Selection of participants
Approach to participants
Interview practices

3.1 CONTRACTUAL REQUIREMENTS

The Specific Assignments Section of the Statement of Work under which the survey was conducted reads in part:

> "The contractor shall determine the different programming environments that affect the different types of tools used in software quality control. Typical environments shall range over small systems developed by a single programmer, medium systems developed by small teams, and large scale systems. The software and systems developed may be characterized as scientific, management, business oriented, real-time command and control, word processing, real-time transaction oriented, or some other application area."
>
> "The contractor shall report the current state of the art regarding the development and use of tools in each of the environments. The final report shall include the following:
>
> * Types of tools being developed and used
>
> * When in the software life cycle tools are used
>
> * Software standards that the tools are attempting to enforce
>
> * Cost/benefit analysis of the tools (if this data can be easily obtained)
>
> * Compatibility of tools with respect to commonality of representation and redundancy."

The Statement of Work required at least eight private sector organizations to be surveyed. At the suggestion of the contractor the scope was subsequently enlarged to include Government and Government-support organizations as well. Also, in the planning discussion with NBS/ICST personnel, the meaning of 'software quality control' in the first quoted sentence above was clarified to include all activities that can be expected to affect software quality.

3.2 SELECTION OF PARTICIPANTS

The key factor in the selection of participants for the survey was to cover the wide range of programming environments specified by the contractual requirements within the limited resources and time available. The major emphasis was on including organizations of a wide range of size and on obtaining approximately equal representation of private, Government-support and Government activities. Also, representation of scientific and business oriented programming was sought within each organizational category. The detailed classification guidelines for the survey are discussed in Section 4, as well as the characteristics of the survey population.

It was realized at the outset that tool usage among tool developers represented a special case that was of interest in the overall assessment of software tool usage. Therefore an attempt was made to include several tool developers in the survey population. Although tool developers constitute only a very small part of the total software population, and thus it is difficult to capture a representative sample in a small survey, their participation was easily obtained. Tool developers are motivated by nature and self-interest to furnish information about their tools. So as not to bias the survey by the over-representation of tool developers among the survey population, significant findings are in most cases stated separately for tool developers and others.

In order to husband resources, a major factor in the selection of participants was location in the vicinity of Los Angeles where the contractor's offices are situated. Of the 23 sites visited, only seven were located outside of California. Of these, five were Government organizations for which no equivalent in terms of size and mission could be located near Los Angeles. One organization each in the private and Government-support sectors was outside of California. Thus, a strong bias in the geographic distribution of participants had to be acknowledged. It was not believed that this would have a major bearing on the significance of the findings.

Publicity during the conduct of the survey prompted four organizations to volunteer their participation. One of these was a Federal agency, the others were private companies. Due to schedule constraints it was possible to contact only one of these organizations in person, and findings from that interview were incorporated in the following analysis. The interview questionnaire was sent to the other volunteers, and these responded in writing. Because of possible inconsistencies in the interpretation of some questions, these responses have not been utilized in the quantitative data in Sections 5 and 6 (except where specifically stated). Tools and software problem comments from the volunteers have been included in the qualitative discussion where appropriate.

3.3 APPROACH TO PARTICIPANTS

Participation in the survey was entirely voluntary. After an organization had been tentatively selected by the criteria mentioned above, an informal inquiry was made, usually by phone, to determine willingness to participate in the survey. The scope of the survey, the nature of the questions, and the approximate length of the interview (one to two hours) were discussed. Of those contacted, only two declined, both because knowledgeable personnel were fully committed during the period over which the interviews were scheduled. In addition, one Federal district office in Los Angeles referred the request to the headquarters office, and the latter was then selected as a survey site.

Once agreement on participation had been reached, a letter was sent by NBS/ICST that stated the interest of the sponsor and made a formal request for an interview by SoHaR personnel. A copy of this letter is included as Appendix A of this report. All recipients of these letters were, in fact, interviewed. In some cases a request for anonymity of the interviewee or of the organization was made. It was therefore decided to omit this information for all interviews. However, no descriptive material has been omitted or disguised. It is believed that the classifications described in the following section and used throughout the rest of this report captured the primary detail of the user that has a bearing on software tools utilization. Additional information is contained in the interview synopses which are being published separately.

In several instances it was apparent from the initial telephone contact that there were several independent software centers within a given organization, and that it would be desirable to contact more than one of these. In two organizations three interviews were actually conducted, but it was found that the third one contributed little additional material. It was then decided to restrict interviews to two within a given organization, and to select centers that could be expected to reflect wide differences.

3.4 INTERVIEW PRACTICES

All interviews were conducted at the respondents' premises in accordance with a previously agreed to interview schedule and, with a few exceptions cited in the following, all appointments were adhered to. In one Government-support organization where two interviews had been set up, one of the prospective respondents was not available. Information obtained at the site indicated that the tool usage was not materially different from that of the center at which the interview was conducted; no attempt was made to reschedule that interview because considerable travel was involved. In one Federal agency the initial contact for the interview was involved in a high level meeting and the interview was conducted with a substitute who was very knowledgeable in most areas. Answers to the remaining questions were later obtained by phone from the original contact.

To conserve time during the interviews, and also to permit the survey to generate consistent information, a standard set of questions was used (reproduced in Appendix B). The interview consisted of two parts. The first part covered the user environment, significant software development practices,

and general experience with software tools. During the second part of the interview the experience with one specific tool was explored in depth, including the motivation for tools acquisition, actual versus expected use of the tool, recommendations for modifications or additional uses of the tool, and comments on actual or possible integration of the tool with other software development aids. Where tools of significantly different scope were in use at a given center (e.g., one tool for software development and another one for project management), part II of the interview covered each of these separately. Notes were made at the time of the interview but the interviews were not recorded.

About midway through the interview schedule, it became apparent that current tool usage and future interest in tools are influenced by what the respondent perceives as the major difficulty in generating high quality software. The following question was therefore added to part I: "What are the major software development problems in this environment?". Answers to this question are discussed in Section 6.

Because details on tool options and performance were not always available at the time of the interview, descriptive literature on a tool or a user manual were obtained in some instances. In other cases, the interviewees volunteered to furnish tools catalogs or technical papers describing development practices in their environment. Material obtained in this manner is, where applicable, integrated into the interview synopsis.

Two organizations requested that the record of the interview be submitted to them for review and additions or corrections were made in this process. In all other cases, the information obtained during the interview formed the only basis for the analysis in this report.

SECTION 4

CLASSIFICATION OF ENVIRONMENTS

This section considers a fairly large number of factors in the environment that might affect software tool usage. Some of these are eliminated either because they are highly correlated with more readily recognized factors, or because the discussion shows that their effect on tool usage is difficult to assess. The elimination of factors that are irrelevant or have only a minor effect on tool usage helps to focus the remainder of this report on factors of major significance.

The following factors will be discussed in the order listed:

Size of software organization
Type of organization (private, Government-support, Government)
Applications (scientific, MIS) and language
Development environment (batch, interactive)
Program running environment (batch, interactive, real-time)
Computer type
Involvement in tools development
Survey Coverage

4.1 SIZE OF THE SOFTWARE ORGANIZATION

Large organizations are expected to be more inclined to use software tools than small ones because (1) they have greater resources at their disposal, (2) they can amortize the tool and training costs over a larger base, (3) their software development procedures are usually more formal, (4) there is more emphasis on use of tools for organizational control of the software development process, and (5) they are more likely to be required by the customer to employ software tools or to furnish data which are most effectively obtained by tools.

Some problems are encountered in defining what is meant by 'size' and how to measure it. Software development and maintenance budgets, or the number of computer hours used for these functions, are suitable indicators of size, but they are not always known or made public. The size of the programming staff is an equally valid indicator, and one that is usually more readily available. It is used in the following as the size measure. It may also be argued that the overall size of the organization, rather than its software component, should be considered. Resources, and possibly also the degree of formality in development, are indeed determined by overall size. Nevertheless, the narrower definition is preferred because tool usage, as described below, is based on the scope of control exercised by software management.

Where all software development is carried out in a single location and under a single line management, the size that is significant for tool usage is taken as the total programming staff engaged in development, test, and maintenance. Where software is being developed or maintained in multiple locations, or where it is

distributed among several line managements (e. g., one software group supporting a revenue function and another one a disbursement function), it must be determined on an individual basis whether tool usage will be governed by identical criteria for all of the locations or groups, and in that case the 'size' is that of the total programmer population. If local criteria govern tool usage, then the 'size' is that of the programming group for which the criteria are valid.

Most software tools, and particularly development tools that provide the format and audit functions, impart uniformity to the software which has been processed by them. This uniformity is regarded as a distinct benefit in environments where there is a deliberate attempt to control the software development process and software quality. The size of a software organization, as far as its effect on tool usage is concerned, is in a significant way determined by the scope of control of the management level that enforces uniformity of the development process and of quality assurance. Thus, a large software organization, in which individual small groups set their own software quality standards represents a tool usage environment that is equivalent to that of several small organizations.

This definition of size is particularly important in classifying the environment represented by Federal agencies. It will be recognized as extremely naive to represent all Federal software development as a single environment. But even major departments or individual agencies or bureaus usually do not enforce a sufficiently uniform approach to software quality that would make them the proper unit for the assessment of size in the present context. The size of a software organization is in the following expressed as the number of programmers who are subject to a single set of guidelines for software quality and the software development process.

Size, as defined above, can of course range from a handful of programmers to a staff of several hundreds or even thousands. For the study of tool usage it has been found sufficient to divide this continuum into four categories:

Small - up to 14 programmers
Medium - 15 to 39 programmers
Large - 40 to 99 programmers
Very large - 100 or more programmers

These categories were primarily established on the basis of the supervisory structure by which they are controlled. The small environment is typically managed by a single decisionmaking individual. The medium environment is one where two layers of decisionmakers may be involved, but where there still is (or can be) direct communication among all individuals engaged in software development. The large environment usually involves three layers of management concerned with software quality or development practices, and formal communication among programmers may be the rule. Very large environments typically have staff departments that establish or coordinate software development practices and that evaluate, introduce, and frequently develop software tools.

4.2 TYPE OF ORGANIZATION

This study distinguishes between three organization types: private, Government, and Government-support. The latter includes Federal Contract Research Centers, National Laboratories, and similar quasi-Governmental bodies. These have fewer procedural constraints than Government agencies and may be more motivated to achieve a minimum cost solution to software development.

The conventional expectation is that all private software development is profit oriented and that tool usage will be governed primarily by economic considerations, while Governmental agencies are procedure oriented and will use tools accordingly. As will be seen in the following section, these stereotypes are seldom found in current software practice. To a large extent, tool usage is in all three types of organizations governed by preferences of one or a few individuals who consider themselves 'experts' in this field. These preferences may have been acquired by study, by generic or specific tools experience, or through meetings of user organizations of the major computer manufacturers.

A further classification of the private sector into Government end use and non-Government end use of the software or the service provided by the software may also be considered. To some degree this distinction parallels that provided by size, since the very large private sector organizations mostly produce software that in some way services Government functions or is subject to Government regulation. This subdivision is not further pursued here.

Similarly, a division of Government organizations into civil and defense sectors may be considered significant for tool usage. This division is largely paralleled by the distinction between scientific and MIS software which will be discussed in the following subsection. Most of the scientific program development is carried out by agencies associated with the Department of Defense or NASA, or by portions of other agencies which pattern their software development practices deliberately after those in the defense sector. Furthermore, in MIS software development there seems to be little difference between the practices of DoD and non-DoD agencies.

Although the findings of this survey do not indicate a large difference in tool usage between the three organization types discussed here, it is believed that continued use of this distinction is warranted. Status as a Government organization provides for the developer access to a large number of tools currently in Government inventory; easy communication with users (and frequently also with the developers) of these tools; and, in some cases, the possibility of using the tools on their present host via remote job entry or interactive terminals. Also, it is expected that there will be greater emphasis on uniform or, at least, similar software development practices among most Federal agencies. All of these will provide a motivation to use tools which is not present in the private sector.

Government-support organizations are expected to continue to occupy a middle ground, being partly influenced by the increasing uniformity of development practices among agencies which they serve and being more concerned with the economic factors of tool usage than Governmental bodies.

4.3 APPLICATIONS AND LANGUAGE

The two major program applications which affect tool usage are scientific and administrative. Because the latter have recently expanded considerably into the area of information retrieval, the term management information systems (MIS) is more descriptive of the types of programs handled and is generally used in this report.

The predominant language for scientific programming is FORTRAN. PL/1, BASIC, PASCAL and APL are used to a much lesser extent. In the aerospace and defense sector, JOVIAL (Air Force), CMS-2 (Navy), and HAL-S (NASA) are required to be used for embedded computer applications (and some others) but FORTRAN is a recognized and popular language there, also. The wide use of FORTRAN is significant for tool usage because many tools are specifically designed for this language. Thus, the FORTRAN user has a much wider choice of tools than programmers working in other languages.

The logic structures IF THEN ELSE, DO WHILE, and DO UNTIL which are preferred for structured programming are not directly implemented in the conventional FORTRAN dialects, and this has given rise to a large number of pre-processors which translate these constructs into FORTRAN and usually also provide some editing and format functions. These tools fill a unique, language dependent need of the scientific programming environment. Another language specific tool need arises from the large number of FORTRAN dialects in use which inhibits transfer of programs from one environment to another. This need has been met, on one hand, by analyzers that determine whether a program adheres to a standard subset of FORTRAN and, on the other hand, by tools that translate from one dialect to another.

In the MIS environment COBOL is the most widely used language, with PL/1 and ALGOL having a very minor share. A number of tools are specific to COBOL, and a number of others (file and data base managers, computer utilization reporters) address specialized needs of the MIS environment. Many current applications in this field, and practically all new ones, run interactively, usually from a large number of terminals. This places a high premium on computer resource utilization and implies a high benefit if the run time of programs can be reduced. Dynamic analysis tools with resource utilization, timing, and tuning features are frequently employed. Some of the more popular of these are specifically designed to eliminate inefficiencies due to current COBOL compilers.

MIS programmers frequently have less formal training than those working in the scientific data processing field. This is particularly true of some Federal ADP agencies where on-the-job training programs are used to upgrade personnel from the computer operator and clerical categories to the programmer classification. Unless tool usage is made a specific part of the training, such personnel will have no understanding of software tools and will therefore not be inclined to use them.

A positive factor for use of tools in the MIS environment is that commercial computer applications (which are primarily MIS oriented) are still expanding at a rapid rate which motivates tools development for this market. Many of the new computer installations are at the low end of the performance range (mini- and micro-computers), and tools for this specific area are attaining an importance which they have never had before. Because these tools are intended for less sophisticated users than tools running on mainframes and because many address basic program development and maintenance needs, they can be expected to be of considerable benefit to small Federal agencies.

It is sometimes assumed that scientific programmers tend to be more sophisticated than those in the MIS environment and are thus more inclined to use tools, to adapt tools (make minor revisions required for an application), and develop tools. The survey showed much greater involvement with tools development among scientific programmers but no significant difference between the two environments in the extent of tool usage. (The MIS population was heavily biased towards very large organizations and this may have affected this latter observation.) A negative factor specific to the scientific environment is that much programming is being done by personnel who are primarily scientists or engineers and who have little training or interest in the formal aspects of computer programming. These groups are highly motivated to get a program running in as short a time as possible and have little concern for maintainability, portability, or documentation. They will use only those tools that are obviously beneficial for the initial creation of a program, e. g., editors, compilers, and debug aids.

The area of office automation (word processing, electronic mail, and communications management) is becoming distinct from that of management information systems. It is being served by personnel who are usually without any computer training and work with programs (which are really tools) that generate prompts, menu selection, etc. This field is certain to increase in importance and will require much standardization effort (e. g., to permit interchange of media and to facilitate personnel transfer). However, at present it is outside the scope of conventional software tools. One of the organizations contacted in the survey is a developer of word processing software. Although the findings from that interview are incorporated in later analysis, the sample is much too small to permit any generalizations.

4.4 PROGRAM DEVELOPMENT ENVIRONMENT

Two distinctions in the program development environment may affect tool usage: submittal mode (batch vs. interactive), and assignment practice (individual vs. team).

Most software development environments today offer a choice of interactive and batch programming with the decision left either to the individual programmer or to a team leader. Exceptions are found at both ends of the size scale. Some small software organizations that depend exclusively on off-site equipment offer only interactive development. In some very large organizations, applications development is conducted only on interactive terminals, partly as a means of promoting tool usage (e. g., structure graphics, data reference generators).

There is usually a capability for developing systems programs and others that are not tied into the principal application in batch mode.

Very few, if any, tools are inherently constrained to either batch or interactive use, although the specific implementation may be directed to one of these modes due to the prompts, graphics representations, etc., that are employed. Some dynamic analysis tools are used predominantly in the batch mode because they entail a large run-time expansion factor which makes it inefficient to use them from a terminal and because the output is usually desired in hard-copy form. In general, however, the usefulness or efficiency of software tools is not greatly affected by whether the programming environment is batch oriented or uses interactive terminals.

Only in small software organizations is individual programming employed today. Medium sized and larger organizations favor a team approach, even when the scope of the effort would permit assigning it to a single individual. Part of this tendency is due to the need for training junior programmers which can be very effectively handled by a team assignment. Another advantage is that the project schedule is made less dependent on continued availability of a single individual when the effort is assigned to a team. Because they tend to impart a uniform format to a software product and because they automate project recordkeeping (e. g., the schedule feature of static analysis tools), tools tend to reduce some of the differences between individual and team programming. On small software development efforts there does not seem to be a major distinction in the motivation for tool usage between individuals and programming teams.

Larger projects are handled by larger organizations which, as a rule, employ team programming. For some projects, large teams (more than 10 programmers) or multiple teams are required. In this environment the use of tools for scheduling, cross reference, and program library management becomes desirable or mandatory. This aspect of tool usage is, however, already accounted for by the size of the software organization. Since this is more permanent and more easily determined than the attributes of the programming environment, a separate factor for the latter does not appear to be required. The assignment practice (individual or team) is therefore not regarded as a significant factor in determining software tool usage.

In summary, the program development environment as such has little effect on tool usage in general, although a specific tool may be only (or more) useful in a batch or interactive setting. The size of the programming team (which in turn is governed by the magnitude of the program) has some effect on tool usage, but this is highly correlated with the effect of organization size and is better modeled by the latter. Therefore program development environment is not further considered as a classification factor.

4.5 PROGRAM RUNNING ENVIRONMENT

Three major categories of running environment may affect tool usage: batch, interactive, and real-time. Typical programs that run primarily in batch mode are large scientific models (e.g., for determining neutron flux in a nuclear reactor) or periodic report generators (e. g., for monthly transaction

summaries). Typical interactive programs in the scientific area are those that perform spectral analysis or that generate graphics. In the MIS area, practically all programs for aperiodic querying of data bases are interactive, e.g., those that are used to handle inquiries about benefit payments or personal status of an individual. Real-time programs interact with a physical environment which is subject to effects not controlled by the primary application program. Aircraft or satellite attitude control programs and real-time simulation programs are representative examples.

To the extent that batch programs are run infrequently (not a necessary inference), there may be little incentive to optimize run-time or memory requirements. However, in all other respects, they represent the same tool needs (for quality, documentation, etc.) as other programs of an equivalent application.

Interactive programs, particularly those for MIS applications that are being run throughout the business day, place heavy demands on computer resources, and they are therefore prime candidates for the use of optimization tools. Because a large amount of current COBOL programming is directed at the interactive programs, this aspect of tool usage is more effectively considered as a language and application (MIS) specific factor. Interactive scientific programs are usually not run on a continuous basis and therefore the benefits of optimization are not as significant.

Real-time programs usually must execute very efficiently in order to meet the time constraints imposed by their environment; e. g., aircraft flight control routines must furnish a complete three-axis output 25 to 40 times per second. In order to assure the required control over timing, these programs, or parts of them, may be coded in assembly language. This may, at times, generate requirements for special tools, such as decompilers (tools that combine the static analysis feature of structure checking with the transformation feature of restructuring) or flowcharters that accept assembly language input. In general, however, the tool usage is not significantly different from that encountered in other running environments.

The criticality of the application (which tends to be particularly high in real-time environments) may also impose special tool requirements. However, it will be found that a failure in practically any major application program, particuarly in a Government environment, can have very serious consequences that warrant application of test tools (dynamic analyzers) and static analyzers. That these are not currently in wide use is partly due to lack of information and partly due to the difficulties in applying some of these tools. Differences in the criticality of programs among the classes discussed here are relatively insignificant in determining the need for these tools.

The security aspects of the running environment may also affect tool usage. Typical of special security needs are programs that deal with the control of nuclear weapons or with the guidance of vehicles used for the delivery of nuclear weapons. This environment has not been included in this survey, but it is inferred from the general literature that it may involve quite extensive tool usage. The Defense Advanced Research Projects Agency has sponsored the development of tools for formal verification of programs, and it surmised that

these are intended for security sensitive applications [5,6]. A specific instance of a program verification tool is AFFIRM [6].

Aside from the rather special case cited in the preceding paragraph, the various program running environments do not represent significantly different tools needs. This factor is therefore not further considered in this classification.

4.6 COMPUTER TYPE

Both the development (host) and the target computer type can have a major effect on tool usage. This effect is due to the size of some of the software tools, their execution time requirements, and the operating system and language requirements posed by the tools. It is quite natural for commercial tool development to address the most widely used computers and operating systems. Federal agencies have a large variety of computer types, and comparatively few of these are those in common use in other environments. This poses a serious obstacle to employment of commercial software tools, and the variety of computers also restricts the benefits of common development or tools sharing among agencies.

A number of approaches can overcome this obstacl,e but none of them are cheap or immediately available (see Section 8). At present the dependence of tools on specific computers and operating systems must be considered a serious obstacle to wider employment of tools. The great variety of computers used by Federal agencies makes tools standardization and tools sharing very difficult. A classification of tool usage by specific computer types may be beneficial but it was not within the scope of this survey.

4.7 INVOLVEMENT WITH TOOL DEVELOPMENT

As might be expected, involvement in tool development provides a strong motivation for tool usage, not only of the local product but also of that from other sources. This was well borne out in the tool user survey, where every tool developer used at least one general purpose tool. Since some tools development takes place in small or medium organizations, this makes their tool usage distinct from that of non-developers.

Several factors seem to be involved: interest in software technology (as distinct from programming for a specific application), recognition of the similarity of programming tasks that stimulates an attempt to automate them, and the greater facility of tools use that comes from being active in their development (recognition of options and switches, being familiar with the format of an instruction book, etc.).

Classification as to involvement in tools development therefore seems to be desirable in order to identify the likelihood of tool usage in a given environment. The classification as a tool developer is based on primary or major product of the organization. Occasional tools development, particularly development of simulations or report generators, was not so classified. Tools development may not be immediately useful for bringing about more widespread use

of software tools. It is neither possible nor particularly desirable to make every lead programmer into a tool developer. However, the fact that familiarity with one tool seems to stimulate interest in others should be noted. This may motivate a greater effort for introduction of the first tool into a given environment, because it may be assumed that further tools will then be adopted more readily.

4.8 SURVEY COVERAGE

The distribution of participants in the survey with regard to type of organization and size is indicated in Table 4 - 1. Where two centers in a given organization were contacted each is treated as an individual respondent in Table 4 - 1 and in subsequent analyses in this report.

It is seen from Table 4 - 1 that approximately equal participation by all three organization types was obtained. In terms of size, the sample is somewhat more heavily weighted toward large and very large development groups than had been anticipated. This is partly due to the fact that Government organizations are more likely to fall into these size groups. However, a fair representation of all size groups is present, and for each size group at least one participant was obtained in each of the organization types.

TABLE 4 - 1 ORGANIZATION TYPE AND SIZE OF PARTICIPANTS

Size	Number of Participants			Total
	Private	Gov't Support	Government	
Small	3	2	1	6
Medium	1	2	3	6
Large	2	1	4	7
Very l.	3	3	4	10
Total	9	8	12	29

The volunteer participants which returned written responses are not included in the above. These were all private organizations, one each in the small, medium and very large categories.

Twenty-one of the participants listed in Table 4 - 1 did predominantly scientific programming, practically all in FORTRAN. Six of the participants worked predominantly in the MIS area, five using COBOL and one ALGOL. One organization had an approximately balanced MIS/scientific workload, and one was engaged in word processing development, using a microprocessor assembly language. All MIS organizations except one were Government agencies.

Seven of the organizations in the survey were tool developers. Two of these were Government-support organizations. All others were in the private sector. All of the tool developers were from the scientific programming community.

SECTION 5

PRESENT TOOL USAGE

This section summarizes the findings of the survey in several dimensions. First, an overall classification by the extent of tool usage is presented. The tools identified by users in part II of the interviews are then classified by the previously generated taxonomy so as to permit an analysis of individual tool features that were encountered. Finally, comments are presented on certain groups of tools for which specific advantages or disadvantages could be identified from the survey findings. This material is organized under the following topics:

Overall extent of tool usage
Tool features classification
User reaction to selected tools

5.1 OVERALL EXTENT OF TOOL USAGE

It is nearly impossible to interact with a current computer without the use of some tools. Assemblers, interpreters, debug aids, and compilers can all be rightfully called software, tools but they are not of the type that one associates directly with software quality tools. Admittedly, however, any distinction based on the contribution to software quality will be quite subjective. In the above grouping, a good case can be made that compilers support software quality because a source program in a high order language is more readily reviewed and maintained than one coded in assembly language. A better delimiter is that the tools listed above are normally part of the operating system and are acquired without a deliberate decision on the part of the user. Participants in the survey who used only tools that are customarily furnished with an operating system were classified as 'minimal tool users'.

Another level of tool usage is represented by those who acquire, or in some cases develop, special purpose tools suited for their mission and with no direct impact on software quality. Simulators and file managers are typical tools in this category, as are editors and precompilers that do not include static analysis features (see tool features classification below). Survey participants who only used tools in this category are called 'intermediate tool users'.

The final group of participants is made up of those who deliberately acquired or developed general purpose tools that include static or dynamic analysis features or that otherwise directly support software quality. Test and audit tools, program design languages, optimizers, and performance measurement tools are representative of these tool classifications. Survey participants in this category are called 'general purpose tool users', and these represent the most extensive level of tool usage.

The overall classification of participants by the extent of tool usage, expressed in the three categories just described, is presented in Table 5 - 1. The participants are defined by the interview number (the alphanumeric preceding the colon), and the extent of tool usage is indicated by the letter following the colon. An asterisk preceding the interview number indicates a tool developer as defined in the preceding section.

TABLE 5 - 1 EXTENT OF TOOL USAGE

Staff Size	Organization Type: Private	Gov't Support	Government
Small	5:M *9:G 14B:G	11A:I 22:I	10A:M
Medium	2:I	*11B:G 18B:I	3:I 8:M 21:I
Large	*14A:G *17:G	18A:I	4A:I 4B:I 15:G 20:G
Very L.	*1:G *13:G 19:G	6:G 12A:G *12B:G	7:G 10B:G 16:G 23:G

Tool Usage Designation:
M - minimal I - intermediate
G - general purpose

One of the immediate observations suggested by this table is that all tool developers are general purpose tool users. Thus among tool developers in the survey, the extent of tool usage was not affected by size or organization type (note that none of the Government organizations was classified as a tool developer by the criteria stated in Section 4, although some incidental tools development was carried out at two of the sites).

In the following analysis, tool developers are omitted from the data. To obtain a better insight into the overall status of tool usage, it is convenient to assign numerical (ordinal) indices to tool usage. Arbitrarily, we assign 0 for minimal tool usage, 1 for intermediate tool usage, and 2 for general purpose tool usage. The content of a cell in Table 5 - 1 can then be characterized by a number that represents the average tool usage of the members of that cell. Thus, among small private participants there are one minimal user (index 0) and

one general purpose user (index 2). The average index for that cell is therefore 1.

In order to permit meaningful aggregation of cells in a row or column, a weighting factor corresponding to the number of entries in a cell must also be available. This weighting factor is shown in the left part of Table 5 - 2. The first row average is computed as

$$(2 \times 1 + 2 \times 1 + 1 \times 0) / 5 = 0.8.$$

It is emphasized again that tool developers have been omitted in this quantitative evaluation of tool usage.

TABLE 5 - 2 NUMERICAL INDICES OF TOOLS USAGE

Size	NUMBER OF USERS, N[i,j] Organization Type			TOOL USAGE INDEX, T[i,j] Organization Type			
	Private	Gov't Support	Gov't	Private	Gov't Support	Gov't	Row Average
Small	2	2	1	1	1	1	0.8
Medium	1	1	3	1	1	0.7	0.8
Large	0	1	4	-	1	1.5	1.4
Very l.	1	2	4	2	2	2	2
Total (N) or Column Average (T)	4	6	12	1.3	1.3	1.3	1.3

Before discussing these findings in detail, it should be recognized that the total population involved here is not large enough for a meaningful statistical evaluation. Nevertheless, the difference in tool usage by small and medium software organizations, as opposed to that by large and very large ones, is made very clear in this presentation. Also, for the population analyzed here, the organization type does not seem to affect tool usage. Small and medium Government organizations had a lower score than equivalent private and Government support organizations, but the population of the individual cells involved is so small that this finding should not be generalized.

For the six MIS organizations in the survey, the average index of tools utilization is 1.2, slightly less than for the population as a whole. Again, the small difference and the overall limitations of the survey suggest caution in interpreting this finding.

5.2 TOOL FEATURES CLASSIFICATION

In a Work Assignment that preceded the Tool User Survey, a Taxonomy of Software Tool Features was generated, and this formed the basis of an NBS Special Publication [2].

This taxonomy deviates from previous efforts at tools classification in that it evaluates a tool as a processor rather than by the purpose for which the tool might be employed. The features of a processor are defined by input/function/output capabilities. The key tools identified in part II of each of the interviews were classified by this taxonomy. A summary of the principal functions found in these tools is presented in Table 5 - 3. The functions are classified into three major groups: transformation, static analysis, and dynamic analysis. Each of these is in turn divided into detailed functions. Thus, under transformation the detailed functions that were most frequently encountered were editing, formatting, instrumentation, optimization, restructuring, and translation.

TABLE 5 - 3 UTILIZATION OF FUNCTION FEATURES OF KEY TOOLS

Feature	Number of Uses In Key Tools
Transformation	22
Translation	10
Editing	9
Formatting	8
Restructuring	6
Static Analysis	21
Management	11
Scanning	10
Auditing	3
Dynamic Analysis	13
Coverage Analysis	5
Tracing	3
Simulation	3

The transformation feature was utilized by pre-processors, editors, general development tools, and test analyzers; all of these were represented by multiple entries in the key tools list. The static analysis feature was used by some pre-processors, all test tools, general development tools, and libraries. Dynamic analysis was involved in simulations, performance optimization, and test tools. The frequency with which certain detail features were utilized suggests

that considerable simplification of tool usage may be possible in an integrated tools environment.

A very interesting finding, somewhat related to tools integration, is that six of the tools used all three major function features. With one exception, these tools were in use at large or very large organizations. The exception involved an integrated collection of small tools that was used and regarded as very helpful in a small software organization. It must be stated, though, that this small software organization was part of a very large overall organization, so that one assumes that the sophistication in tool usage was not completely home-grown.

5.3 USER REACTION TO SELECTED TOOLS

The reaction of the immediate user to the tools identified above was not uniformly favorable. Most tools, however, were regarded as useful. Comments on some selected tool groups are presented below under headings of:

Immediate user oriented tools
Management oriented tools
Tools primarily used by large organizations
Special purpose tools

5.3.1 Immediate User Oriented Tools

The tools that elicited the most favorable comment from the immediate user were editors, pre-processors, and (general purpose) development tools. The detailed features that most of these have in common are editing, formatting, scanning, and, to a lesser extent, tracing. From the user's point of view, these tools obviously simplify the generation and correction of programs, and the computer input operations. Because most of them address a restricted area of software development and have few options, they are generally regarded as easy to use. User documentation seldom exceeds 20 or 25 pages. In most cases the documentation was adequate for the original version of the tool. Some complaints were stated regarding updating of documentation for revisions in the tools.

5.3.2 Management Oriented Tools

From management's point of view, libraries, data base managers, and performance analyzers received the most favorable comments. In addition to the management function of static analysis, the detailed features most frequently found in these tools were editing, restructuring and optimization. Many of these tools furnished reports in a format tailored for second and higher level management, and they helped to improve software or computer productivity. The reaction of programmers and first level supervision was not always positive. The use of these tools involved some effort on their part and the payoff was not apparent to them.

The most adverse comments on the use of some of these tools arose from misunderstanding of the purpose for which they were employed. A resource utilization analysis tool was declared useless for achieving performance improvements on specific programs, the reports generated by the tool were criticized as being too voluminous, and the admittedly small demands that the tool made on computer resources were felt to interfere with normal operation. The initial contact for this interview had been made at a high management level, and it was known from this conversation that there was above average appreciation of software tools in this organization. There was an opportunity to question the same manager at a later time about his evaluation of this tool. He explained that the reports were very valuable to him because they clearly identified abnormal events in the computer center, and conversely, when these reports showed no abnormal events, he was assured that the computer operations were proceeding normally. He was well aware that this tool was not suitable for fine-tuning an individual program. In this case the difficulties perceived at the working level were not due to shortcomings in the tool, but rather to lack of information about its intended purpose.

5.3.3 Tools Primarily Used by Large Organizations.

For several tools used primarily by large and very large software organizations difficulties were cited repeatedly by both the immediate user and management. Most test analyzers are in this category. The characteristic features of these tools are instrumentation and coverage analysis. The most frequent complaint was associated with run-time expansion, and problems were also voiced regarding difficulties in use and restrictions on program size or structure. In spite of these criticisms, test analyzers were regarded as useful tools because the information which they furnished was impossible to obtain by other means. The problems cited should be evaluated in connection with any efforts aimed at expanded use of software tools.

Requirements analyzers represent another generic type that was regarded as valuable but difficult to use by survey participants. Requirements analyzers process a very high level language (VHLL) and their static analysis features include completeness checking and consistency checking. These tools make extensive demands on computer resources (the program for one of them was reported as one million FORTRAN statements) and are usually operated by specialist teams of four or five members. A tool of this type is suitable only for use by a very large software organization. Yet, automated requirements analysis can make a very valuable contribution to software quality because requirements errors are an ever present cause of software failures and one that is very costly to correct if it is not found in the early life cycle stages.

Requirements analyzers provide a particularly efficient means for handling changes in requirements, with regard to checking for consistency with the unchanged parts of the original requirements and assuring propagation into documents that implement the requirements. An extensive review of requirements by a specially convened team of experts can usually be arranged at the beginning of a project, and this practice will not be completely eliminated, even if a requirements analysis tool is employed. However, such review teams cannot usually be assembled when changes have to be phased in, frequently on short notice, nor can they be depended on for perfect recall of the considerations

that went into the original requirements formulation. Thus, even if there are no economic benefits from a requirements analysis tool at the outset of a project, its use will pay off over the years as the inevitable changes in requirements are processed, and the established requirements data base can then be utilized. Yet it is difficult to obtain a commitment to use a requirements analyzer if the benefits are, as they should be, described as accruing primarily in the future. Hence the use of these tools is sometimes being induced by unrealistic promises of immediate payoffs. When these are not being achieved, a negative attitude is being generated with regard to requirements analyzers and, frequently, with regard to other software tools as well.

Since requirements analyzers are usually operated by specialist teams, and without interaction with the immediate user, it should be possible to host them on centralized facilities that can be accessed by multiple users on a fee for service basis. Such an arrangement may remove some of the heavy start-up costs, and it will make the services of these tools available to smaller users.

5.3.4 Special Purpose Tools

By their nature, special purpose tools can be expected to be well suited for the environment in which they are employed, and user reaction is therefore predictably favorable. Two types of tools encountered in the survey fall into the special purpose category: simulators and data base managers (for a unique data base). The former are identified by the use of the simulation feature of dynamic analysis (and usually very few additional features), the latter by formatting, management and scanning features (this identification is not unique). Difficulties, insofar as they were mentioned, related to the maintenance of these tools, particularly in the area of program documentation and in the updating of user manuals.

Special purpose tools are typically developed as a temporary measure, or to serve what is perceived to be a very short term need. If the tool is successful, it is used on a continuing basis and requests will be received to extend or adapt it for additional applications. The developer is enthusiastic about the acceptance of the tool and seldom owns up to the fact that considerable effort is required to transform the hurriedly developed program into a software product that can be maintained over a period of time or can serve multiple users. If the inevitable difficulties are experienced, they are frequently attributed to inherent problems in the use of software tools. A definition of a basic programming format for a 'tool', and minimum standards for both user and maintenance documentation may avoid some of these problems.

SECTION 6

ADDITIONAL ANALYSIS OF THE SURVEY

This section provides a further analysis of selected questions from the tool user survey and an evaluation of the impact of these findings on tool usage. The subsections analyze, respectively, the tools acquisition process, the experience with current tools, user statements about projected tool usage, and user characteristics that may affect tool usage.

6.1 ACQUISITION OF TOOLS

A number of survey questions that deal with the tools acquisition process are analyzed for the purpose of capturing information that may be useful for future efforts in the introduction of additional software tools. Specific headings deal with the sources of current tools, the authority for tools procurement, and the motivation for obtaining the tools.

6.1.1 Sources of Tools

Of 31 tools for which the source could be determined, 12 were developed in-house. Three of these were special purpose simulations, and three others also served rather specialized applications. The remaining six were general purpose tools, and most of these were in use also in other organizations. The high proportion of in house development is probably not typical for the overall tool user population and may reflect that tool developers are likely to be overrepresented in a survey of this type. A summary of sources of tools that were procured from the outside is shown in Table 6 - 1.

TABLE 6 - 1 SOURCES OF PROCURED TOOLS

Source	No. of Tools
Open market	7
Public domain	5
Computer vendor	5
Commissioned	2
Total	19

The public domain category includes tools acquired by a Government agency in the survey from another Government agency. Both of the commissioned tools are dynamic analyzers that had been developed by private companies for specific Government applications. The use of commissioned tools was encountered only in Government agencies. There were no other significant distinctions in the source of tools among the types or sizes of organizations.

6.1.2 Acquisition Procedures

The authority for tools procurement rested primarily with local supervision (defined as any level of supervision collocated with the tool user) as shown by the summary of the authority for tools procurement presented in Table 6 - 2.

TABLE 6 - 2 AUTHORITY FOR TOOL PROCUREMENT

Authority	No. of Organizations
Local supervision	18
Headquarters	6
Committee	2
Diffuse	2
Total	28

The need to refer to headquarters was encountered in five Government organizations and one Government-support organization. Committee approval was required in one private and one Government organization; in both cases the committees were local. Diffuse authority was reported by two very large private organizations. In addition to the normal authority shown in the table above, two very large private organizations (both defense contractors) indicated that tools could be procured out of project funds if the contract so stated or permitted. The project manager rather than the in-line supervisor then became the procurement authority.

It was also quite apparent that informal routes of tools procurement exist in large and very large organizations of all types. This involves 'borrowing' tools from a friend in another organization or through computer user groups or in-house development of tools without specific authorization. The latter process does not always produce beneficial results. The tools thus developed frequently lack adequate documentation and are not designed for portability. If it is attempted to apply them away from the native environment considerable difficulties are experienced, and this may discourage further tool usage.

6.1.3 Motivation for Tools Acquisition

Motivation for tools acquisition was not expressed by the survey participants in ways that could be easily classified. In many cases several reasons were undoubtedly at work at the same time. In only two organizations, both large and private, was a deliberate cost-benefit study made prior to tools procurement or in-house development. In a few other instances economic considerations were a factor in tools procurement, but no specific analysis was undertaken. At least two of the survey participants mentioned that the tool was 'free', and it was implied that economic benefits were therefore obvious. This reasoning does not account for the labor required to evaluate, adapt and install the tool, or for the training in its use.

In many cases the procurement or development of a tool was based on a clearly identified need, and economic factors were not explicitly considered. Examples are pre-processors for structured programming, simulations, and management tools that generate reports required or desired by a customer. In others, the tool was perceived to present a better way of accomplishing a necessary task, and technical management authorized its procurement out of discretionary funds. Program design languages and performance monitoring or optimization tools may be acquired out of this motivation. In still other instances the technical challenge, the desire to innovate, or the possibility of overcoming present difficulties may lead to tool development or procurement. This was encountered in the survey particularly with respect to overall software development facilities and requirements analyzers.

6.2 TOOLS EXPERIENCE

The users were in general quite satisfied with the tools in current use. In only one case was a recently acquired tool characterized as unsatisfactory, and that for some of its intended applications. This case involved a dynamic analysis tool procured by one Government agency for use by another in which user constraints had not been correctly interpreted to the developer. Comparatively minor modifications could probably have corrected most of the deficiency but a proper avenue for funding these was not currently available. A local 'toolsmith' would have been very effective in this case. Comments on non-use or abandonment of tools, reliability/maintainability, adequacy of training and documentation, and options used are analyzed individually below.

6.2.1 Reasons for Non-Use or Abandonment

This question was asked with particular reference to tools used in the past. Meaningful answers fall mostly into three categories: the tool was too difficult to use, it was functionally not suited, or a better tool was found. The relative frequency of these reasons is shown in Table 6 - 3. The last category includes two cases where documentation was unavailable or very inadequate.

TABLE 6 - 3 REASON FOR NON-USE OR ABANDONMENT OF TOOLS

Reason	No. of Organizations
Better tool available	9
Difficult to use	5
Functionally not suited	4
Other	3
Total	21

6.2.2 Reliability/Maintainability of Tools

In only a few cases did the organization contacted during the survey maintain the tool with regard to which this question was asked. In all of those cases the maintenance was rated easy or without problems. The reliability of tools was also generally rated high. Pertinent user comments have been classified below as unqualified good, qualified good, and other. The latter category includes the dynamic analysis tool mentioned in the introductory paragraph of this subsection, and a computer simulator for which it was stated "There have been some unexplained crashes. There is no protection against illegal instructions". Typical comments that were translated into qualified good are "Reliability is now acceptable" and "No known bugs. Input syntax errors are not always well diagnosed". There were no apparent differences in the reliability ratings between various types or sizes of organizations.

TABLE 6 - 4 USER'S RELIABILITY RATING OF TOOLS

Rating	No. of Organizations
Unqualified good	16
Qualified good	9
Other	2
Total	27

6.2.3 Training and Documentation

The training provided for tool use was characterized as predominantly formal (in-house or outside), sometimes formal, and predominantly informal. The latter frequently was restricted to making documentation available to potential users. In the survey population as a whole the three types of training were pretty evenly represented but small organizations relied primarily on informal training, while large ones preferred formal training. This is shown in Table 6 - 5.

TABLE 6 - 5 TRAINING FOR INTRODUCTION OF TOOLS

Type of Training	Size of Organization				
	Total	Small	Medium	Large	V. large
Formal	10	0	2	3	5
Variable	11	1	4	3	3
Informal	8	4	0	2	2
Total	29	5	6	8	10

The type of organization did not have a significant effect on training method. Large organizations are more likely to be general purpose tool users (see Table 5 - 1) and also depend more heavily on formal training. This produces an apparent correlation between formal training and general purpose tool usage which is not considered to be a cause and effect relationship.

Specific comments on documentation were received for 30 tools. Of these, 13 were rated as good or equivalent (useful, helpful). Twelve were described as adequate or in other terms that indicated less than complete satisfaction. In five cases the documents were considered unsatisfactory by the user. Typical comments in this category were "A User's Manual is available but it is considered marginal. It should be redone" or "Documentation for this tool is not particularly good. It should be improved". A manual that did not pertain to an identified tool in the survey (and is not counted among the five responses reported above) was characterized as unsatisfactory because "It did not warn the user of all the special conditions". Documentation was the item most criticized by the survey population.

6.2.4 Tool Options Used

The use of options was described for 28 tools as shown in Table 6 - 6

TABLE 6 - 6 EXTENT OF TOOL OPTIONS USAGE

Extent	No. of Tools
All options used	13
Partial use of options	8
Extent not known	6
No options available	1
Total	28

The first category contains some cases in which general purpose use was apparently a rare occurrence. One user stated that 90 percent of the runs involve only 10 percent of the available options. However, as long as a feature is used at least occasionally it was classified as 'used'. Partial use of options was due to two causes: features that were not needed at all in the survey application, and features which might be desired but were awkward to use or usurped excessive computer resources (this latter factor was mentioned with regard to two dynamic analyzers). The large number of 'unknown' responses was due to the fact that the individual interviewed did not use all of the options and felt he could not, within reasonable effort, determine the extent of usage in his entire organization.

There was slight evidence that partial use was more prevalent in Government installations than in the other types of organizations. This may be due to the

presence of several dynamic analyzers with very comprehensive features among the Government tools. Size of the organization did not seem to be a factor in determining the extent of options use.

A number of the respondents indicated that options were removed at installation or after some experience had been gained with the tool. Note, however, that several users would have liked more flexibility in the use of their tools (see discussion of desired improvements, 6.3.1). It may be concluded that when a new tool is introduced the number of options should be kept to a minimum. This simplifies training and will also reduce the computer resource requirements. After experience has been gained with the basic tool function, the user may be motivated to consider further options.

6.3 FUTURE TOOLS USAGE

Both the extension in the use of present tools and the acquisition of additional tools are covered under this heading. About 80 percent of current users indicated a desire to improve some features of their current tools, and all but one of the organizations contacted had plans for acquiring additional tools.

6.3.1 Improvements and Integration of Present Tools

Improvements in specific features of their current tools were mentioned in 15 organizations. The need for improved documentation has been discussed above and is not repeated here. A classification of the desired improvements is shown in Table 6 - 7.

TABLE 6 - 7 DESIRED IMPROVEMENTS IN PRESENT TOOLS

Improvement	No. of Organizations
User interface	5
Performance	4
Flexibility	4
Portability	2
Total	15

Examples of desired improvements in the user interface are (1) replacement of a very formal command language for a management tool with a less formal one, and (2) provisions for forward and backward traceability between structured code and FORTRAN in a preprocessor. The recommended performance improvement involved processing speed and, in at least one case, the total size of the code. In the flexibility area the suggestions involved handling more language forms, removing size restrictions on the code to be analyzed, and adaptation to new versions of operating systems. The desire for portability was raised with regard to one general software development system that is at present completely tied to a military host computer, and an improvement in portability was desired for a restructuring and analysis tool.

The issue of tools integration was commented on with respect to 21 tools. For five of these it was stated that they existed already in an integrated environment. Three of the tools had been developed from the outset as an integrated system, in one case integration of project management and test records had been achieved around a dynamic analysis tool, and the remaining tool was a simulator of broad scope. The remaining comments on integration were classified as shown in Table 6 - 8.

TABLE 6 - 8 PROPOSED INTEGRATION

Type of Integration	No. of Tools
With library	4
Development/test system	3
Non-specific	5
No integration desired	4
Total	16

The suggested integration with a library usually involved the ability to invoke commands or data sets automatically, to identify and store the results of the tool's operation, and to compare various sets of tool outputs. These comments were made with regard to editors and report generators. Integration into a development/test systems was recommended for preprocessors and a test management tool. A negative response on this question was furnished for several large tools, including requirements analysis and management tools.

6.3.2 Contemplated Acquisitions

Twenty of the responding organizations indicated a desire to acquire tools in the near future. No distinction was made between desire at the technical level and firm committment to procure the tool. The 28 tools identified in these responses are classified by purpose in Table 6 - 9. Tool features that might serve these purposes are discussed later.

Source editing and control tools were primarily desired by small and medium organizations, and test support tools primarily by large and very large organizations. For all other classes in Table 6 - 2, size did not appear to be a major determinant. The type of organization did not seem to affect the purpose for which the tools were to be employed.

TABLE 6 - 9 PURPOSE OF CONTEMPLATED TOOL ACQUISITIONS

Purpose	No. of Tools
Source editing & control	5
Test support	5
Requirements analysis	4
Code analysis	4
Project management	3
Performance improvement	3
Software development systems	2
Other	2
Total	28

Source editing and control tools include sophisticated editors, library managers, and file managers. These tools typically include some transformation functions and the static analysis functions of management and scanning. Test support tools all include the dynamic analysis functions of coverage analysis and tracing, the static analysis function of structure checking, and usually a number of other static and dynamic analysis functions as well. Requirements analysis tools are characterized by accepting very high level language as an input on which a number of transformation and static analysis functions are performed. Tools for project management typically accept a data input and perform the static analysis functions of cost estimation, scheduling, and management. Code analysis tools rely primarily on static analysis functions of auditing, scanning, and structure checking. The static analysis tools were desired to improve both the uniformity and the portability of the code. The performance analysis tools utilize dynamic analysis functions of timing and/or tuning, and usually a number of static analysis functions as well. The two tools in the 'other' category were both intended for specialized purposes, one for language research and the other one as a macro facility.

6.4 USER CHARACTERISTICS OF INTEREST FOR TOOL USAGE

Under this heading two user characteristics that were captured in the survey and which may be pertinent to actual or potential tool usage are discussed: the major software problems seen by the user, and the involvement in standards and professional activities.

6.4.1 Major Software Problems

Only a part of the survey population was asked to identify what they regarded as their major software problem. Where there was contact with several levels of management, the question was addressed to the highest management level. Responses were obtained from 14 organizations (including the three volunteers)

who identified 19 problem areas. These responses are classified in Table 6 - 10.

TABLE 6 - 10 MAJOR SOFTWARE PROBLEMS

Problem	Total	Organization Type		
		Private	Govt. Support	Govern-ment
Requirements	7	3	0	4
Maintenance	4	3	1	0
Personnel	3	2	0	1
Software engnrg.	3	2	0	1
Managmt. reports	2	1	0	1
Total	17	9	1	7

Inadequate requirements definition and traceability was identified as a problem in each of the four Government agencies which responded to this question. In the private sector it was less of a problem. The three organizations which identified it there were small software activities, although two were in large corporations. In the other private organizations there is either better communication between user and developer, or the problem is not being recognized. Where the private sector develops software for the Government, inadequate requirements may be perceived as a Government problem rather than as a software problem.

Requirements definition is a very serious problem throughout the software community. The above fragmentary data indicate that it is being felt with special severity in Government organizations. Current requirements analyzers are technically capable of helping in this area but are not extensively employed because of the personnel and computer resource requirements which they impose (see 5.3.3). This area deserves further investigation because of the potentially great benefits that can be derived from a systematic and, at least partially, automated approach to requirements definition.

That maintenance was not seen as a problem area in the Government organizations is somewhat surprising. Two of the respondents contracted maintenance to private companies and may thus have been shielded from this problem. Shortage and skill levels of personnel were commented on by three organizations. Software engineering practices were described as a problem in terms of the difficulty of introducing them and auditing for compliance. These may also be regarded as personnel problems but they have been separated here because they could be more directly resolved by the employment of tools than other personnel problems. Inadequate management reports represent a similar category in that the problem can be directly attacked by introduction of proper tools.

None of the participants specifically mentioned reliability or downtime in responding to this question. This may be due to the fact that comments on major software problems in <u>their</u> environment were requested. The organizations were software developers rather than users, and in their own operation was probably

not severely impacted by software failures. At least two of the respondents were working on software for reliability-critical applications. This points to a possible information gap in current software practice: the developer may be aware that the software is for a critical application but receives no formal feedback on the number of failures or the consequences of failures. Thus the actual benefit of a test tool cannot be evaluated in that environment.

6.4.2 Standards and Professional Activities

Ten of the organizations contacted as part of the survey participated in standards activities, either Governmental or voluntary (professional or trade associations, ANSI, etc.). Nine of these were classified as general purpose tool users in Section 5, and one as an intermediate tool user. This is a significantly higher degree of tool use than is found in the survey population (and presumably in the general software population) as a whole. It is not concluded that standards activities per se lead to tool usage. All but one of the participants in standards activities were large or very large organizations, and the exception was a tool developer. Thus their tool usage can be accounted for by other factors. However, that standards and tools are closely related must also be kept in mind. One user made a strong case for considering standards as tools.

All of the participants in standards efforts were also active in the professional area. In addition, seven organization that did not have a significant involvement with standards indicated participation in professional activities. Of these, four were classified as general purpose tool users and three as intermediate tool users in Section 5. All but two of the seven additional participants in professional activities were also in categories that had very high tool usage. The two exceptions were non-tool developers in small organizations who were both classified as intermediate tool users. These findings do not indicate a strong relationship between participation in either standards or professional activities and tool usage.

SECTION 7

INTERPRETATION OF FINDINGS

In this section the findings of the Survey of Software Tools Usage are interpreted in a broad context to define factors favorable to tool usage, obstacles to tool usage, and requirements for effective tool usage derived from these. The comments made here are reflections on the survey rather than reporting of results, and they are presented for the purpose of stimulating further evaluation of the survey findings on the part of the software community as well as on the part of tool developers. Wherever possible, specific data presented in earlier sections are referenced to support the interpretations discussed here.

7.1 FACTORS FAVORABLE TO TOOL USAGE

Factors that appear to promote the usage of software tools are considered here under two headings: factors arising from current practice and additional factors that can be expected to become significant in the near future.

7.1.1 Current Factors

Explicit economic analysis is rarely used as a criterion for tools acquisition, and it was not identified as a determining factor by any of the 12 Government organizations included in the survey (see 6.1.3). The decision to acquire a tool is most likely to be made by a local manager or committee who may take into account only that subset of cost and benefits which is an immediate constraint, e. g., purchase or lease cost, or computer throughput (see 6.1.2). The tools which received the most favorable comments, particularly in small organizations, were those that met clearly perceived technical or management needs, e. g., precompilers for structured code and report generators. Where portability was emphasized, code auditors that checked for adherence to standard language syntax were in the same category.

Another group of tools which received very favorable comments were those which produced benefits in the immediate area of concern to the tool user as pointed out in 5.3.1. Use of these tools is self-motivated, and there are few obstacles to their introduction as long as they are available in a suitable format (for the computer, operating system, and programming language in use at a given facility), are reasonably well documented, and have a 'friendly' user interface.

With regard to all of the tools mentioned above, there is somewhat of an embarassment of riches: there are many good ones available, and several that were functionally very similar were encountered in the comparatively small population of this survey. Greater uniformity in tools selection among Government and Government-support organizations would be beneficial in providing uniformity of software practice and in facilitating transfer of code,

documentation, personnel, and tools operating experience. It will also reduce the cost of maintaining and improving several sets of programs and documentation.

7.1.2 Future Trends

Just as structured programming with 'non-structured' languages has made the use of precompilers essential, it can be expected that further development of software engineering practices and their adoption in organizational or broader guidelines will lead to the acceptance of additional tools. This trend is already evident in the translation of design into code. A design language and two development systems that include the equivalent of a design language were encountered in the survey, and all of them received favorable comments from the user.

Factors associated with software quality have been identified in a number of past studies, and metrics for these have been proposed [7,8]. In at least one case, metrics of this type have been included as a contractual requirement. The evaluation of the metrics has been carried out by mostly manual means [9], but if this practice spreads, software tools will undoubtedly be used extensively for automated reporting of the metrics.

Cost modeling and reliability modeling are other areas in which large amounts of data on software characteristics are required in order to validate or apply current research. Software tools will be required to support the extensive data collection. Further impetus for tool usage may arise from the adoption of naming conventions within large programs and data bases. Adherence to conventions and verification of uniqueness of names can best be carried out by software tools.

Last but not least, the increased interest at high levels of project and organizational management in gaining visibility and control of software development is expected to speed the introduction of tools. Initially these may be management oriented tools that generate timely reports on resource utilization and software status. As a result of these efforts, certain software practices will be recognized as beneficial, others as detrimental to efficient development and design. Tools may be directly introduced in order to support the beneficial practices, or they may be included in a broad effort to promote uniformity of software development over larger aggregates of both Government and private organizations. The survey participants frequently commented that present software practices were fragmented but that there was perceptible pressure or already existing directives toward greater uniformity.

7.2 OBSTACLES TO TOOLS USAGE

Obstacles to tool usage that were encountered during the survey are discussed here under three headings: factors arising out of the computer environment, organizational factors, and tool related factors.

7.2.1 Obstacles in the Computer Environment

Some very elementary obstacles to the use of tools arose from the incompatibility of existing tools for a desired function with the computer environment that wanted to use them. The tool may not be available for the computer in use at that facility or for the operating system employed there, or it may not have been able to handle the high order language and dialect in which the programs were coded. In a few cases, the tool was compatible with respect to these factors, but it required too much memory or imposed too great a throughput penalty to be beneficial.

As far as new tools are concerned, these obstacles can be partly avoided by greater emphasis on portability (see 7.2.3). For both existing and new tools, the use of standard conversion packages or access to tools on a remote host are alternatives that need to be explored. One of the Government organizations contacted in the survey mandates that all integration and testing be conducted by remote access to its own mainframe computers, using target computer simulators that it supplies. This is a division of a defense agency that has played a leading role in the standardization of both high order languages and computers. The portability of both source and object code is therefore no problem in this specific environment, and the tools compatibility problem has been solved by having a permanent host for the tools.

7.2.2 Organizational Obstacles

The primary organizational obstacle to use of tools is that many of the most effective tools produce benefits that lie partly or entirely outside the scope of the organization that is responsible for their acquisition and use. The following discussion focuses on requirements analyzers, test tools, and software management systems. Several other tool types may be affected by the same considerations.

The potential benefits of requirements analyzers have already been mentioned (see 6.4.1). The perceived obstacles to their use are (1) the personnel required for initial installation and training, (2) the personnel required to establish the requirements data base for each project, and (3) the computer resources for performing requirements analysis. The first of these can be overcome by funding from computer technology budgets which exist at most agencies. In all cases where requirements analyzers are currently in use, both in the private sector and in Government, the acquisition funding has come from such a source, and a means for dealing with this difficulty is thus identified. The computer resources have been a serious obstacle in the past because both the program and the data base for requirements analysis tend to be very large. With higher performance computers, and with memory limitations being largely removed by virtual memory techniques, this obstacle, too, may be overcome.

This leaves the second factor as the crucial one. The cost of establishing a data base has to be paid for out of funding in the early stages of a program when budgets tend to be very slim. There is some benefit from having an automated requirements analysis during the initiation phase, but it is not a compelling factor, and it is questionable whether it warrants the cost. The

chief benefit of having a data base occurs when changes are introduced later in the life cycle, and when the validation and documentation of these changes can be accomplished automatically at a fraction of the cost and schedule that is required by present methods. Because of the high cost, validation and documentation of software changes are sometimes omitted with very severe consequences for the overall project. The present organizational structure does not usually provide incentives for management to allocate resources during the project initiation phase to achieve benefits in later life cycle phases. Aside from very broad directives (specific emphasis on this item in life cycle cost studies), no clear resolution is currently envisioned. To provide for a more comprehensive solution of this problem in the future, a creditable case study of the use of a requirements analyzer through initiation, development and operations phases should be undertaken.

Similarly, the primary benefits from the use of test tools occur in the operations phase whereas their cost has to be borne during development. Here, again, the cost of the tool itself and some of the start-up expenses may be funded as part of a technology effort. But the use of a dynamic analyzer, particularly if it performs a sophisticated coverage analysis, will increase both personnel and computer resource requirements during test, and the only benefit to the test function may be knowledge that they have done a 'good job'. Where test tools have been used consistently and successfully, it has usually been either in the tool developer's own organization (or one affiliated with it), or in an environment where the use has been mandated. Recently an increasing number of test analyzers have been put into use, and a there may be a tendency to consider a computer operation backward if it does not have one. However, there is a difference between running a demonstration project and efficient daily use of such a tool. Because of the large number of different test tools in current use, and the somewhat differing objectives for their use, conclusions about the merit of these as a group are not possible at this time. Further study of functional, performance, and resource usage differences, and an overall assessment of their benefit, will be necessary if broad guidance on the employment of test tools is to be generated.

Software management systems as a minimum attempt to chart the progress of development against milestones established at the project's inception. In many cases they also generate expected rates of expenditure on the basis of historical trends or on the basis of information gathered from the specific software under development. The effort required to initiate these management systems frequently has to be borne by elements of the project organization which are in close contact with the details of the program and who don't need a sophisticated tool to tell them whether the next scheduled review dates will have to be postponed. The benefit of the tool is felt at a much higher level of management where the uniform format and the consistent standards for data entries which are enforced by a software management system provide significant savings in the review of multiple projects. A further benefit of the use of these tools is that they automatically generate a well-formatted data base that can improve cost estimation and scheduling of future projects. If these benefits can be brought to the attention of the management level at which they are really targeted, a much more widespread acceptance of software management systems may result.

7.2.3 Obstacles Arising from the Tools

Not all tools provide a net benefit, regardless of the scope over which the assessment is made. Difficulties reported with specific tools have already been discussed in 6.2.1. In this connection, attention is also called to a critical evaluation of the benefits of maintenance tools in a recent publication [10]. The following addresses obstacles arising from generic features of some tools categories.

Some tools automate fairly routine activities that are now being performed by personnel of low skill levels, and will therefore reduce labor requirements in that category. However, it was found that the use and maintenance of the tools require additional personnel at high skill levels. Although there may be ample economic justification for the use of a tool in terms of overall labor reduction, some agencies feel that they cannot provide the required skill level. A frequent complaint was that Federal ADP organizations cannot attract or retain highly skilled personnel whereas they can get lower skilled personnel or trainees. This objection can be partly overcome if tools are shared among a number of agencies, and the effort for tool maintenance on the part of any one agency can be proportionately reduced.

Comments made by users and non-users suggest that increased acceptance of tools can be achieved if ease of use is specified as a primary criterion for tools procurement. Tools are typically developed in very sophisticated organizations who see the tool user as one who has at least an undergraduate degree in computer science. This does not correspond to the present state of affairs, and should not be expected to prevail in the future. Tool commands, interpretation of output, and the general documentation must be addressed to personnel who have either one year of on-the-job training or a junior college education.

Although the use of a tool may provide labor or resource savings over the long run, the introduction of a new tool was in many organizations found to require considerable personnel and computer time, and it therefore impeded routine operations over a short period. In some agencies, particularly the smaller ones, the temporary unavailability of people and computing resources was perceived to be a serious obstacle to investigating and using tools. It is undoubtedly true that most tools do not become fully effective as soon as they are introduced, and a temporary reduction of resource availability for routine operations may therefore be unavoidable. This factor can be partly overcome by providing good documentation and training aids (which will be facilitated if the expense can be allocated among a number of users). To some extent, this must also be addressed as an organizational problem, and a sufficiently high level of management must be involved in tools acquisition to permit a reasonable trade-off of long term benefits versus short term costs.

The final factor to be discussed in this category is the usage of computer resources by tools. This was not a significant factor with regard for most users, but where it was mentioned as objectionable it was felt to be extremely so. One major tool type was found to account for all of the adverse comments: dynamic analysis tools, particularly coverage analyzers. By their nature, tools in this category expand the run time of the programs which they analyze, and

this was accepted by the user. However, when more than a 10-fold expansion takes place, a serious curtailment of computer availability was experienced, and the value of the tool -- and of the information furnished by it -- was called into question. Some tools were reported to increase the run time by a factor in excess of one 100! A careful, specialized study of this tool type, which has already been suggested in 7.2.2, will be useful in clarifying and possibly resolving this issue.

7.3 REQUIREMENTS FOR FUTURE TOOLS USAGE

This subsection summarizes the principal requirements for effective tool usage that are derived from the above comments. The two headings deal, respectively, with requirements that must be levied against the tools and requirements pertaining to the environment, the latter including organizational factors.

7.3.1 Requirements on Tools

A primary requirement for tools is that they address a clearly recognized need in the environment to which they are targeted. This means that a tool that improves software quality (reliability, maintainability, portability, or a related factor) will be efficiently applied in an organization that recognizes a responsibility for providing the quality factor supported by that tool. In any Government sponsored future tool development it might be well to identify rather specific objectives.

Unless the tool is intended for a single host application, portability is a very important requirement. Past attempts at achieving portability have not been uniformly successful. During the conduct of the survey it was related that a well-known portability auditor for FORTRAN did not fully comply with the criteria that it enforced. Usually portability is only achieved at a considerable throughput penalty. A suitable compromise might be to allow machine-dependent code in identified sections of a tools program provided that its purpose and format are well documented so that a local toolsmith can accomplish required modifications without excessive effort.

That tools have to be useable by personnel of low skill levels was repeatedly mentioned during the survey. This requires that a 'friendly' interface be provided for the control input and for the user output: descriptive names for commands, prompts, and in output listings; the capability for corrections in the tool set-up; comprehensive checking of the control input for syntax, consistency, and completeness; and suitable diagnostics when irregularities are detected in either the tool or control input.

The most frequently mentioned deficiency of existing tools is poor documentation. To some extent this is due to tools developed for local use being accepted on a wider scale, a development which should not be discouraged per se. But the additional requirements for making it suitable for general use must be recognized and funded. As a minimum, tools for general use should have the following documentation:

Installation manual that identifies computer resource requirements, machine-dependent code, user settable variables, limits, etc., and which includes test cases.

User's manual, if possible, organized in several levels for beginning users and more advanced ones, with a detailed description of all user operations and commands, the effect of options on run-time and memory requirements, and an explanation of diagnostics.

Systems manual for use by the toolsmith or an advanced programmer for making adaptations, trouble shooting, and performing minimal maintenance.

Maintenance manual if a central maintenance organization has not been identified.

7.3.2 Environmental Requirements for Tools Usage

The following addresses primarily requirements for effective use of tools that enhance software quality. While the preceding heading pointed out that a tool should address a specific need for software quality, a corollary is that the environment should define the quality factors which are considered important in a given application. If software maintenance is a more significant item than software reliability, the acquisition of documentation tools may be preferred to that of test tools.

A closely related consideration is that those who use the tool and the tool output should be aware of the objective that is to be accomplished and should have some metric available to determine that this objective is being achieved. Development and test groups are currently being tasked with using documentation tools, code auditors, and test analyzers which support primarily maintenance and operational reliability. In a few cases, organizations interviewed as part of the survey did not have an accurate understanding of the purpose of one of their tools. But much more prevalent was the condition where the purpose was understood yet a measure of accomplishment was not available. This is particularly true of test tools where no quantitative data on software reliability are being collected.

As has been repeatedly stated, the benefits of tool usage must be explained at the management level which stands most to gain. Because a significant benefit of many tools is that they provide uniformity and consistency, and facilitate future software tasks, a management level concerned with those broader aspects must be addressed.

Tools use within the Federal Government could be made more effective if a central clearinghouse for tools information were available. The tools catalog recently published by NBS/ICST and the software tools exchange being established by GSA are a start in this direction. The collection and dissemination of tool users' comments, the development of guidelines for tools introduction and integration, the conduct of tools demonstration projects, and the comparative evaluation of tools to eliminate duplicate development and maintenance

activities (which were found in the survey particularly in precompilers and test tools) are also required to facilitate efficient tools use. While none of the Federal agencies contacted in the survey had conducted explicit cost/benefit analyses regarding tools acquisition and use, most would have liked to have data and techniques for such analyses made available to them. A tools clearinghouse would be the logical source for supplying this information which is considered essential for rational software practices in Federal agencies.

SECTION 8

CONCLUSIONS

This section summarizes findings of the survey that are significant for the application of software tools to a broad range of Federal agencies. Conclusions that affect individual tools are discussed in Section 5, and those that affect individual user segments are discussed in Section 6.

8.1 OVERALL TOOL USAGE

All organizations contacted in the course of the survey used some software tools. Where the tools set was restricted to those normally furnished as part of the operating system (compilers, assemblers, debug aids, etc.) tool usage was classified as minimal. Three of 29 organizations were at this level. In other organizations tool usage included special purpose programs that were essential for their mission (simulators and special purpose data base managers). These were classified as intermediate tool users and comprised 8 of the 29 survey participants. The remaining 18 organizations used at least one general purpose tool in their regular operations. Although a few of the larger organizations had tools that automated significant portions of software management, software development, or test, no 'complete' tool usage was encountered. All organizations contacted could have benefited from additional tool usage, and many were taking steps in that direction.

The small sample size precludes a formal statistical evaluation of the information from the survey but two groups were clearly identified as consistent tool users: organizations that have more than 100 programmers and analysts subject to a uniform software discipline, and tool developers. All survey participants in either of these groups used general purpose tools. Among the four size categories considered in this report (see 4.1), there was a consistent increase in general purpose tool usage with size. With regard to tool development one is faced with a chicken and egg problem: does tool usage lead to tool development, or vice versa? The survey did not provide a firm answer to this question. A high degree of sophistication and experience in software development is required to create a tool that can be successfully employed outside its native environment, and one can speculate that this experience must in many cases have included tool usage.

The survey included three types of organizations: private sector, Government support, and Government. When tool developers (which were found only among private and Government-support organizations) were eliminated, there was no perceptible difference in tools utilization among these three types. In small and medium organizations there was slightly higher tools utilization in the private sector which may have been due to other characteristics of the specific participants. Similarly, the finding that tools utilization was slightly higher among scientific programming organizations than among MIS organizations need not be generally valid.

8.2 MOTIVATION FOR ACQUISITION OF TOOLS

Where motivation for tools acquisition could be clearly established, it usually arose from a need which could not be readily met in any other way. Precompilers, simulations, and some management tools (report generators) are examples of this type of usage. Most tools acquisitions seemed to be motivated by multiple factors, e. g., a tool is described in the literature or seen in operation, and it is recognized as offering a better way of accomplishing some present tasks. Where performance monitoring, software development systems or design languages were in use, their acquisition seemed to have followed this pattern. Requirements analyzers and test tools may be installed out of the same reasoning, and sometimes also because of the professional challenge which they represent. The need in these areas can of course be demonstrated to management, but the tool capabilities, the cost of installing the tool, and the cost of the operation and support activities are usually evaluated in a subjective manner, if at all. In only two cases, both in commercial organizations, was a formal cost benefit analysis made prior to tools acquisition.

The responsibility for tools acquisition was in the hands of local management in approximately two-thirds of the organizations contacted. Approximately one-half of the Government agencies and one Government-support organization needed to refer the decision to higher (remote) headquarters. The localized control over tools acquisition encourages experimentation, and this is regarded as beneficial at the present stage of tool development and application. It also promotes a proliferation of tool designs addressing a single need, and this is not beneficial, particularly among Government agencies, because it inhibits the development of a common software environment and the duplication causes unnecessary costs.

8.3 BENEFITS OF TOOL USAGE

Some of the benefits of tool usage are evident to the user and need little comment: the precompiler, the library system, and the optimization tool can be evaluated on the basis of the service they render in the immediate environment. Many tools, and particularly those that can have the greatest impact on software quality and cost, achieve diffuse benefits not all of which are evident to the immediate user. Software development systems provide appreciable benefits in automating routine tasks of software development and permitting teams to interact in a systematic manner, and these are recognized by the development group. They can also furnish management reports, help in documentation and configuration management, and provide standardization of software development within a large organization. The latter attribute, in turn, facilitates transfer of personnel and activities, provides flexibility in project planning and assignment of computer resources, and permits sharing of experience in the use of the tool and in other aspects of software development which is not possible in present autonomous environments. These broader benefits need to be brought to the attention of a management level where they will be fully appreciated. The survey indicates that this is seldom the case at present.

Requirements analyzers, test tools, and software management systems also provide wide-ranging benefits, some of which will accrue only in much later life cycle stages. The obstacles that arise from this have been described in 7.2. Here, too, it is necessary to alert high level management of the availability and capabilities of the tools so that proper resources for their employment can be made available. Another significant information gap that may impact the proper employment of tools is the lack of quantitative software reliability feedback from the user to the developer. This was signaled in the survey by the fact that no software developer identified reliability or a related quality as a primary problem area although several were working on reliability-critical programs and a fair number were using, or were getting ready to use, test coverage analyzers.

8.4 POLICIES THAT PROMOTE TOOL USAGE

Before organizations can decide on the use of tools they must know about them. In some cases that knowledge can by itself motivate the tools use as discussed in 8.2. The establishment of a software tools data base at NBS/ICST and a systematic classification of the tools listed there by means of the tools features taxonomy are significant and desirable steps in that direction.

Once it knows about the tool, an organization may want documentation at various levels of detail (see 7.3.1) and may benefit from the experience of a current user. In some cases this information can be obtained from computer users' organizations, from the vendor, or through professional acquaintances. Some of these sources have an obvious bias, and none of them are universally available (even the tool vendor may no longer support a tool). The software tools tools exchange being established by GSA is desirable for these reasons.

The training, 'hand-holding', and other information transfer that is required to bring a tool to productive status within a new user environment are frequently underestimated. Guidelines for these steps will be generated under a task that will follow the survey reported on here.

Difficulties for many potential tool applications arise from the lack of portability -- the desired tool cannot be interfaced with the computer installation, the operating system, or the language used by the target organization. Several avenues are available for improving portability, but none produce immediate results or are particularly cheap. Policies in this regard need to be developed carefully.

The first possibility is to make tools highly portable by coding in a standard high order language and to program interactions with the operating system (I/O operations) in a symbolic manner. This involves a considerable loss of efficiency because the non-standard features of high order languages are usually provided to improve efficiency, and because the symbolic designation of input or output operations will slow down the operating system. Because many tools place a heavy demand on computer resources even if coded for a specific host, this additional requirement may impair the usefulness of the tool.

A second approach is to restrict the commonality to the design of tools and to have several versions, each coded for efficient execution on a different computer. This obviously increases the development expense and will require considerable research in order to arrive at a suitable combination of tools and computer adaptations.

Finally, network access to tools that are hosted on a suitable computer may be considered. The most serious limitations of this approach will be encountered in communication channel capabilities and in the ability of the tool to handle language and data peculiarities arising from the target computer. The latter factor may not present much difficulty if the program is in a standard language and uses standard data formats, e. g., COBOL with 32-bit data.

As an overall conclusion of this survey, it may be stated that an interest by high level management in software and a commitment to control software development will be the most significant motivation for tool usage. It is presumed that this interest exists now, or will soon exist, because of the crucial dependence of most Government services on software and also because of the cost, personnel, and management problems which have been reported [3] and can be expected to increase in severity. The role of tools in providing visibility of software development and test, in promoting uniform and systematic procedures, and in providing flexibility of personnel and resource assignments must be brought to the attention of the highest management levels so that tools can be properly and speedily applied.

REFERENCES

[1] D. J. Reifer and H. A. Montgomery, "Final Report, Software Tool Taxonomy", Prepared for NBS/ICST under subcontract to SoHaR Incorporated by Software Management Consultants, Torrance, CA, 1 June 1980.

[2] Raymond C. Houghton, Jr., "Features of Software Development Tools", NBS Special Publication 500-74, February 1981.

[3] The Comptroller General, "Wider Use of Better Computer Software Technology Can Improve Management Control and Reduce Costs", FGMSD-80-38, General Accounting Office, April 29, 1980.

[4] H. Hecht, "Synopsis of Interviews from a Survey of Software Tool Usage", forthcoming NBS Interagency Report, 1981.

[5] S. L. Gerhart "Program Verification in the 1980s: Problems, Perspectives, and Opportunities", ARPA Order No. 2223, Information Sciences Institute, Marina del Rey CA, August 1978.

[6] S. L. Gerhart et al., "An Overview of AFFIRM: A Specification and Verification System" in Information Processing 80, (F. H. Lavington, ed.), North Holland Publishing Co., 1980

[7] B. W. Boehm et al., "Characteristics of Software Quality", TRW Systems Engineering and Integration Division, Redondo Beach CA, TRW-SS-73-09, Dec 73

[8] J. A. McCall, P. K. Richards, and G. F. Walters, "Factors in Software Quality", RADC-TR-77-369, Nov 77

[9] G. F. Murine, "On the Selection of Requirements Analysis Metrics for a Distributed Network", Panel presentation at the Third Minnowbrook Workshop, Syracuse University, Aug 1980

[10] B. P. Lientz and E. B. Swanson, "Impact of Development Productivity Aids on Application System Maintenance", Data Base, Winter-Spring 1980, pp.114-120

APPENDIX A

LETTER SOLICITING PARTICIPATION IN THE SURVEY

UNITED STATES DEPARTMENT OF COMMERCE
National Bureau of Standards
Washington, D.C. 20234

The National Bureau of Standards' Institute of Computer Science and Technology (ICST) has significantly increased its effort in support of the Brooks Act (PL 89-306) which aims to aid Government agencies to improve cost effectiveness in the selection, acquisition and utilization of automatic data processing resources. As part of these efforts, ICST has contracted with SoHaR Incorporated to conduct a study of software tool usage in both Government agencies and the private sector. It is the purpose of this letter to solicit your participation in this study.

The study is aimed at determining environmental factors which inhibit or promote tool usage, and thus you may make a valuable contribution even if you currently do not use any tools. For publication of the results of this study outside ICST the identification of the participants may be deleted, if desired. SoHaR Incorporated is not engaged in tool development, marketing, or furnishing tool related services. They have also agreed to safeguard any information which you consider proprietary in the same manner in which it will be handled by a Government agency.

Unless you notify us to the contrary, you will be contacted by SoHaR personnel during the April to June time period to set up a discussion session with a person or persons knowledgeable in your data processing activities. The discussions will be conducted at your facility at a mutually convenient time and will ordinarily be completed in less than one hour. Your cooperation in this study will be helpful to the ADP community inside and outside of Federal agencies and will be much appreciated.

Sincerely,

Raymond C. Houghton, Jr.

Raymond C. Houghton, Jr.
Programming Science Division

APPENDIX B

TOOL USER INTERVIEW

Part I - Tool User Information

1. Organizational identification of user.

2. Approximate annual budget of programming and computer operation departments (alternatively, the number of analyst/prgrammers and the number of computer operators).

3. Relationship to ultimate software user, and the nature of user imposed requirements (govt. standards, etc.).

4. Range of programmer skills.

5. On-site computer equipment and operating systems.

6. Off-site equipment, operating and communications systems pertinent to tools uasage.

7. Programming languages now used and considered for the future.

8. Software requirements and design (formality, reviews, updates).

9. Programming environment (size of groups, on-line vs. batch).

10. Software quality control (organization, formality, standards).

11. Documentation (responsiblity, formality, standards).

12. Changes and software maintenance (responsibility, procedures).

12A. What are the major software problems in this environment.

13. Participation in computer and software standards efforts.

14. Awareness of software engineering literature and meetings.

15. Circumstances of the first tool acquisition.

16. Tools currently in use.

17. Tools no longer in use.

18. Tools considered for future use.

19. Who decides on tool procurement and by what criteria.

20. How are new tools introduced (formal and informal training, documentation only).

Part II - Tool Information

21. Tool identification, originator, and date of first use.

22. Tool owneership data (licensed, on time-share, etc.).

23. Original purpose and current application(s) of tool.

24. Tool and target software data.

25. Tool usage data (set-up time, run time, analysis required)

26. Typical usage (case history).

27. Frequency and conditions of usage (customer specs).

28. Benefits of usage (vs. prior practice).

29. Standards serviced or enforced by the tool.

30. Principal original reason for tool acquisition; were other tools considered, and what was the basis for selecting this tool.

31. Current reason for tool usage,; target software characteristics important for use (e. g., safety critical applications, very tight timing requirements).

32. Formal cost/benefit anlaysis if available.

33. Are all features of this tool used? Most and least used features.

34. Suggested tool improvements.

35. Tool reliability/availability/maintainability.

36. General comments on tool use.

37. Recommendations for other applications of this tool.

38. Compatibility of this tool with other tools and with the procedures of the computing environment.

39. Are some functions of this tool also provided by other tools or general support software? Where should these functions be located?

40. Suggestions for integration of this tool into a highly automated environment or into a library.

41. How will anticipated future software engineering developments affect the usefulness of this tool.

42. Effect of tool usage on software quality.

43. Effect of tool usage on personnel requirements.

44. What documentation was furnished with this tool? How useful is it?

45. Was special training provided with this tool?

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBS SP 500-82	2. Performing Organ. Report No.	3. Publication Date November 1981

4. TITLE AND SUBTITLE

Computer Science and Technology:
Final Report: A Survey of Software Tools Usage

5. AUTHOR(S)

Herbert Hecht

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D C 20234

SoHaR Incorporated
1040 So. La Jolla Ave.
Los Angeles, CA 90035

7. Contract/Grant No.

8. Type of Report & Period Covered

Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

National Bureau of Standards
Washington, DC 20234

10. SUPPLEMENTARY NOTES

Library of Congress Catalog Card Number: 81-600112

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

The state of the art regarding the development and use of software tools is presented. The information was gathered from an in-depth survey of 23 sites with Government and industry. A comparison is made among the levels of tool usage based on the size and type of programming group. The survey examines aspects that both promote or inhibit tool usage and provide requirements for the future.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

Programming aids; software automation; software development; software engineering; software tools; software tools usage.

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☒ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D C 20402.

☐ Order From National Technical Information Service (NTIS), Springfield, VA 22161

14. NO. OF PRINTED PAGES 58

15. Price $4.25

USCOMM-DC 6043-P80

☆U.S. GOVERNMENT PRINTING OFFICE: 1981-360-997/1858

ANNOUNCEMENT OF NEW PUBLICATIONS ON COMPUTER SCIENCE & TECHNOLOGY

Superintendent of Documents,
Government Printing Office,
Washington, D. C. 20402

Please add my name to the announcement list of new publications to be issued in the series: National Bureau of Standards Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

(Notification key N-503)

NBS TECHNICAL PUBLICATIONS

PERIODICALS

JOURNAL OF RESEARCH—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. As a special service to subscribers each issue contains complete citations to all recent Bureau publications in both NBS and non-NBS media. Issued six times a year. Annual subscription: domestic $16; foreign $20. Single copy, $3.75 domestic; $4.70 foreign.

NOTE: The Journal was formerly published in two sections: Section A "Physics and Chemistry" and Section B "Mathematical Sciences."

DIMENSIONS/NBS—This monthly magazine is published to inform scientists, engineers, business and industry leaders, teachers, students, and consumers of the latest advances in science and technology, with primary emphasis on work at NBS. The magazine highlights and reviews such issues as energy research, fire protection, building technology, metric conversion, pollution abatement, health and safety, and consumer product performance. In addition, it reports the results of Bureau programs in measurement standards and techniques, properties of matter and materials, engineering standards and services, instrumentation, and automatic data processing. Annual subscription: domestic $11; foreign $13.75.

NONPERIODICALS

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NBS under the authority of the National Standard Data Act (Public Law 90-396).

NOTE: The principal publication outlet for the foregoing data is the Journal of Physical and Chemical Reference Data (JPCRD) published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NBS publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NBS publications—FIPS and NBSIR's—from the National Technical Information Services, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Services, Springfield, VA 22161, in paper copy or microfiche form.

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Washington, DC 20234

OFFICIAL BUSINESS

Penalty for Private Use, $300

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE
COM—215

THIRD CLASS