Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

J.C. van Vliet

STARS and stripes

Department of Computer Science Note CS-N8507 August

STARS and stripes

J.C. van Vliet
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

The overall goal of DoD's Software Initiative is to meet DoD's future software needs by an order of magnitude improvement in the state of the practice, and to hasten the transition of new technology. The STARS Program (Software Technology for Adaptable, Reliable Systems) is one of the components of the Software Initiative. Other components are the Ada[1] Program and the Software Engineering Institute. The STARS Program is to result in a fully integrated environment which captures all phases of the software life cycle. From April 30 to May 2, 1985, the first DoD/Industry STARS Program Conference was organized, which brought together representatives of government, industry, and the academic community to review and discuss the STARS Program and other components of the Software Initiative. Below, we report on the presentations given at this conference.

1983 CR Categories: D.2 6.9.2.0
Keywords & Phrases: STARS, Software Engineering Institute, Ada, software engineering environments, software engineering methodology, reusable software, software tools, software transition, measurements, defense applications, business practices, education
Note: this paper is meant for publication elsewhere

1. Introduction

There is a growing gap between the needs for software and our ability to meet these needs. The demand for software personnel shows a 12% increase per year, while personnel and productivity each grow by approx. 4% per year. Furthermore, software requirements grow in complexity and sheer size, and software delivered does not meet the requirements of its users. The transition of software technology from laboratory to standard application in industrial settings takes, on the average, more than 15 years. Though this maturation process is not very different from that in other technical fields, the speed of change in the software field makes the situation considerably worse.

The recognition of these problems has led to DoD's Software Initiative. Its overall goal is to meet DoD's future software needs by an order of magnitude improvement in the state of the practice, and to hasten the transition of new technology. The Software Initiative has three components: the Ada Program, STARS (Software Technology for Adaptable, Reliable Systems), and the Software Engineering Institute (SEI). Other major technical initiatives of DoD, such as the VHSIC Program (Very High Speed Integrated Circuit) [1], also have a strong impact on future defense systems.

The Ada language development took place from 1975 to 1983. As of now, the language has been standardized [2], and various compilers have been validated. Within DoD, Ada is in use in a number of projects, sometimes as a program design language, and sometimes as a programming language. Within the Ada Program, work on the development of an APSE (Ada Program Support Environment) is in progress [3]. The APSE, however, to a large extent concerns the (technical) design and coding

1. Ada is a registered trademark of the US Government (AJPO)

Note CS-N8507
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

phases of a project.

STARS goes beyond the APSE, and is to result in a fully integrated environment which captures all phases of the software life cycle, from initial requirements analysis up to and including maintenance. Within the STARS program, the aim is to develop technology to automate the development process, to reuse existing components, and to increase the portability of systems developed.

The technology developed within the STARS Program must achieve widespread use. To that end, transition mechanisms must be established. There are bottlenecks between research and prototypes, prototypes and products, and products and widespread use. To overcome these, there has to be a need for new technologies. There must be a receptive target community, and the cost and benefits of those new technologies must be demonstrated. Technology transition is facilitated by successful demonstrations and encouragement of its use. The SEI is to play a major role in this transition process.

The STARS Program has evolved over a number of years. In [4], a number of technological areas to be addressed within STARS are identified and discussed. At that time, 7 task areas were envisaged. These tasks are shown in Table 1. The current six program areas were initiated in 1984. In 1985, the procurement process started. From April 30 to May 2, 1985, the First DoD/Industry STARS Program Conference was organized by the NSIA (National Security Industrial Association) in San Diego[2]. The purpose of the conference was to bring together representatives of government, industry, and the academic community to review and discuss the STARS Program and other components of the DoD Software Initiative. The conference drew very few representatives from academia and Europe.

Task Area	Strategy
Measurement	Development of evaluation criteria, associated measures and metrics, experimental evaluation of techniques, methods and tools.
Human resources	Developing training programs, knowledge-based teaching support systems, initiating and expanding academic programs, enhancing career paths.
Project management	Analyzing the software management process, providing an initial collection of support tools, identifying and prototyping longer range needs for support tools, identifying and removing impediments to smooth acquisition, developing acquisition-oriented incentives.
Systems	Defining and refining techniques for selecting and evaluating architectures, developing fault-tolerance techniques and tools, preparing modern systems software, developing automated support for building distributed systems.
Application-specific	Selecting important application areas, developing incentives for reuse, developing application libraries, composer systems, component warehousing capabilities, investigating advanced capabilities such as application generators, knowledge-based systems.
Human engineering	Designing or selecting workstations, investigating cognitive processes involved in software development and support, developing interface design methods and personnel training techniques.
Support systems	Experimental evaluation of current methods, design of project database, development of life-cycle models and standard tool interfaces.

Table 1: STARS overview as of 1983 (adapted from [4]).

The next sections further elaborate on the STARS Program and the Software Engineering Institute,

2. Our participation in this conference was made possible by grants from the Dutch Ministry of Internal Affairs and North-Holland Publ. Cy.

as they were discussed at this conference. We end with a few concluding remarks.

Unfortunately, the proceedings of the conference were not available at the time this report was written. The following discussion is based on notes taken during the conference. Though we have done our best to accurately capture the intent of the various speakers, certain small inaccuracies might have crept in, for which we apologize.

2. The STARS Program

In [4], one envisaged the development of a common automated support environment, to be used by both DoD and its suppliers. The Ada program was to be a cornerstone in the development of this environment. Emphasis was placed on an evolutionary approach, building on current research efforts.

Currently, the STARS Program identifies 6 program areas:

- software engineering environment;
- methodology;
- measurement
- human resources;
- business practices;
- applications.

In the following subsections, each of these areas will be discussed in turn. Most of the issues raised in Table 1 return in one form or another in the current scheme, which looks much more solid than that of 1983. Much progress has also been made in getting the services involved. For each program area, a coordination team with representatives from the services has been set up. The various threads come together at the STARS Joint Program Office under the Director of Computer Software and Systems.

A major difference is that there is much less emphasis in the current scheme on the issue of developing **one** common environment. Instead, a generic environment is planned upon which application-dependent tools can be developed. Because of their common basis these tools can then be ported across environments. Within STARS, DoD wants to work with industry (and share the cost) to build these tools.

As for the environment, it is planned to issue multiple competetive awards for the architecture and interface design in late 1985. The generic environment is to be prototyped and built by 1988, and incremental releases are expected in later years. By the end of 1988, very firm interface specifications must be set, so that industry can be encouraged to adapt to these interfaces, and tools resulting from (prototyping) experiments in the other program areas can subsequently be ported.

This scheme is very tight and ambitious. J. Batz (STARS Program Director) put it thus: "We are aiming high in order to hit high".

The funding profile for STARS is given in Table 2. This excludes the Software Engineering Institute, which is separately funded as a Federal Institute. From the funding profile, is is clear that the STARS Program represents a major and important effort in trying to improve the state of practice in Software Engineering.

FY84	FY85	FY86	FY87	FY88	FY89	FY90
4.9	14.9	52.0	53.6	51.5	56.5	58.6

Table 2: Funding profile for the STARS Program

2.1. Software engineering environment

P. Andrews, chairman of the area coordination team, characterized the underlying problem as: how to adequately support software production and maintenance, and increase quality while curtailing costs. The main line of the program is to develop a framework for integrating methods, tools and procedures for the entire life cycle. To this end, one will prototype, perform experiments, and develop advanced environments and supporting technology.

The main goal of the program is to develop a common automated environment for mission-critical

software. From this common environment, service and project specific environments can be configured. A production-quality environment should be available by mid-1990. Additional goals and objectives are that the environment should:

- lead to an improvement in productivity;
- have built-in stages of increased capability;
- be based on the Minimal Ada Programming Support Environment (MAPSE);
- gain wide acceptance through heavy involvement of industry and academia.

Up to 1988, the main emphasis will be on experiments and prototyping. Early inputs from the other program areas will be used for architecture analysis, demonstrations, and specification. From 1987 onwards, a conceptual model (architecture and specification) of an advanced software engineering environment will be developed. In the later years one will in addition be engaged in transition, and enhancement of the concepts involved.

In their respective presentations, R. Bollinger (Boeing) said that the environment should not be based on one technology, and P. Mauro (Hughes) recommended that several environments be considered, rather than one.

2.2. Methodology

According to G.E. Sumrall, many ad-hoc methods are currently in use within DoD. The existing methods lack tools, do not cover the entire life cycle, and are not widely used and accepted. More specifically, these methods do not support Ada or reusability. DoD has no capabilities to evaluate and intelligently select methodologies.

Having started as an Ada-directed program based on Methodman [5], there has been a shift in emphasis towards identifying a consistent set of existing methodologies, and extending this set to support Ada and the entire life cycle. Reusability concepts will be covered, and consistency with DoD practices ensured. Techniques for evaluating methodologies, and tools to support methodologies in the STARS environment have to be developed. In order to be able to recommend and promote appropriate methodologies, further research and development, measurements, and demonstrations are needed.

In the near term (up to 1987) one will survey and catalog existing methodologies, develop a classification and evaluation framework, and specify requirements for methodologies. Subsequently, a methodology catalog, consumers guide and experience digest will be published. In the long term (1989-1991) a "how to evaluate methodologies" handbook will be published.

R. Bate (TI) subsequently presented a pragmatic view on the methodology issue. A likely reaction from industry is that present methodologies do provide results, that new methods require retraining, and that there is no evidence for the superiority of new methodologies. Philosophy is nice, but software simply has to be there on time. There is a clear need for incentives and/or penalties to encourage industry to pay attention to new methodologies.

G. Booch (Rational) discussed some trends in methodology, such as the increasing attention being paid to formality and evolutionary aproaches. He stressed the point that at present human factors still far outweigh the technical issues involved.

2.3. Measurement

The underlying problems were sketched by D. Weiss as:

- DoD lacks the ability to specify quantitative goals, predict whether goals are met, track progress towards goals, and assess how well goals are met;
- acquisition is hindered by lack of quantitative information;
- the resulting symptoms are: poor cost and schedule estimates, poorly defined goals for quality and performance, software duplication, and inadequate inline control and feedback.

The field of software engineering in general is hindered by a lack of quantitative information. Measurement is a difficult technology. What is known about measurement is not widely used; lessons learned are seldom applied because of difficulties in comparing projects, and causal factors of success

and failure are often elusive. There are no automated means, which makes measurements very expensive. There are no widely accepted definitions and measures for data collection.

The overall goal of the measurement area is to solve these problems. Definitions and measures must be improved, theories must be developed and validated, baselines for quality, resource expenditure and productivity must be generated. The benefits of measurements need to be demonstrated, and measurement tools developed. In the long run, these will lead to a software measurement repository that can be used to develop and validate models for software engineering and acquisition.

B. Boehm (TRW) stressed the point that metrics should give insight, not numbers. The various STARS programs are hypotheses which ought to be tested.

2.4. Human resources

E. Gallagher described the mission of the human resources area as: assess and improve productivity and performance of people by 1991. Because of the trends in personnel supply and demand, a shortage of people is projected. There ar further problems in education (no synchronization between industry and academia), training (rapid technology changes and little or no use of automated tools), and carreer management.

Career patterns will be surveyed and analyzed. The resulting career path recommendations will be used to improve career management structure and promote innovative management. Novel training systems, such as computer aided instruction, knowledge-based systems and other automated tools, will be evaluated. Tools and techniques to support training will be developed and their utilization encouraged. Similarly, an evaluation and survey of existing curricula will be used to develop recommendations that are subsequently introduced to academia and professional organizations.

2.5. Business practices

The business practices area focusses on activities involved in acquisition and program management. Software management is difficult. Resulting symptoms are cost and schedule overruns, and software that does not meet its specifications.

The goal of the area is to improve and integrate the acquisition and management process. To this end, a number of projects have been defined, and requests for proposals (RFPs) have been released. These include:

- Acquisition Program Manager's Assistant: the development of an expert system to facilitate acquisition functions, such as program formulation and definition, planning and estimating;
- Guidebooks for software contracting and acquisition: to consolidate, integrate and improve guidance, identify provisions and mechanisms to improve acquisition;
- Expert system for RFP preparation: to define consistent requirements in RFPs and augment acquisition experience with expert system technology;
- Program manager's toolset: to facilitate and integrate management functions, and define a common interface and database between acquisition and development environment;
- Incentives to reuse software: to define contractual incentives for reuse, and integrate these into acquisition procedures.

2.6. Applications

E. Wald mentioned problems with productivity, quality, management discipline, supportability and reliability. The goal of the application area is to increase productivity, but not to promote mediocrity through homogenization or unsupported/misused standardization. One aims at reusable software parts and system building aids, and application development without programming. Reusability seems technically feasible, but there are sociological issues of trust, access and incentive. It may take time time and special effort to gain acceptance. Over the whole spectrum, one aims at 30% reuse.

Specific goals of the application area include:

- reusable software for specific applications (results: 1987-92);
- reusable library system by 1990;

- system building aids (results: 1989-92);
- development environment (results: 1989-92);
- self-evaluation against goals, beginning in 1988.

The approach to reusability is to define and refine basic definitions, do domain analysis for selected applications, seed part pools for selected applications, and measure the use and merit of parts. As for the system builder aids, one plans, amongst others, to provide for part retrieval assistance and part composition tools. These composition tools will make use of CAD-like graphics in the near term, while in the far term knowledge-based approaches will be used.

The activities of the application area are tightly coupled with the other STARS areas and industry. There is a serious involvement of the services, who also provide part of the funding.

J. Mortison (Sperry) stated that reusability might reduce innovation, since one uses old things. One therefore needs a trade-off study for each project. Though reusability is felt to be technically feasible, the key issue is management involvement.

3. The Software Engineering Institute

The objective of the Software Engineering Institute (SEI) is to accelerate the transition of modern software engineering techniques into practice. Its task is not to develop new technology, though some (goal-directed) research will obviously be needed. Figure 2 gives an indication of how the efforts of the SEI will be distributed.

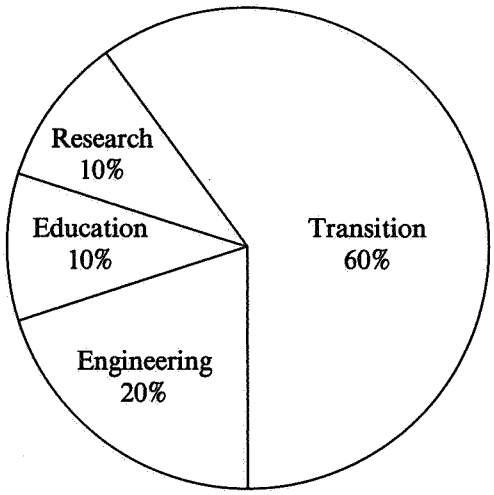

Figure 2: distribution of SEI efforts

The current 5-year plan lists the following program areas:

- technology identification and assessment;
- the nature of the transition process;
- education (MS-curriculum);
- reusability and automation;
- system design, construction and integration (showcase of an open environment that supports programming in the large);
- reasoning about software engineering processing (improve the intellectual control through improved means of analysis and evaluation).

From 1988 onwards, the relative distribution of effort is as follows: approx. 8% is spent on each of the first three projects, and approx. 25% is spent on each of the latter three projects.

The SEI is a Federally funded Research and Development Centre. In addition, it is sponsored by

DoD. Table 3 gives the budget in $M. The SEI was awarded to Carnegie-Mellon University on December 28, 1984 [6]. N. Habermann is its director, while M. Shaw is chief scientist.

	FY85	FY86	FY87	FY88	FY89
funding	5	11.7	14.7	19.6	19.6
sponsors	-	3	6	10	14

Table 3: Funding profile for the SEI

4. Concluding remarks

At the end of the conference, a review session was held in which both positive and negative remarks were made. It was generally felt that much progress was made in getting the services involved. Also, the measurement area was seen as a key area. The program looks much better and more solid than that of 1983.

The single-core software engineering environment was seen as a risk area. Incentives for industry are not clear yet; a related issue is the data-rights problem. There seems to be quite some overlap between the various programs. Details of the SEI — its industrial liaison and connection with other STARS areas — still require attention. To some extent, the program is one of consolidation — let's put into practice what we currently know. This might be risky; future systems must be built better, not just present ones.

References

1. L.W. Sumney (October 1982). VHSIC: A promise of leverage, *IEEE Spectrum* 10.10, 93-94.
2. US Dept. of Defense (January 1983). *Reference manual for the Ada programming language,* ANSI/MIL-STD-1815A-1983.
3. US Dept. of Defense (February 1980). *Requirements for Ada programming support environments "STONEMAN".*
4. IEEE Computer (November 1983). *Special issue: The DoD STARS Program.*
5. US Dept. of Defense (November 1982). *Software development methodologies and Ada, "METHODMAN",* DTIC#AD A123710.
6. R. Steier (May 1985). The Software Engineering Institute at Carnegie Mellon, *Communications of the ACM,* 28.5, 455.