RADC SOFTWARE DEVELOPMENT SPECIFICATION

MICHAEL LANDES

RADC/ISIE, GAFB, ROME, N.Y.

This presentation will describe the RADC Software Development Specification CP 0787796100D. This document, currently in use at RADC, Griffiss Air Force Base, is designed to facilitate the procurement of more reliable and less costly software by requiring the use of modern software engineering techniques. It was created to ensure that Project Managers engaged in procuring systems that included software were protected from the common pitfalls of buying software. While the project manager may be thoroughly versed in the field with which his project deals, he may be quite innocent of the difficulties inherent in contracting for software generation.

The Specification is appropriate for the acquisition of system, application, and support software. It is designed to offer maximum flexibility to fit individual efforts. It consists of a series of sections, properly worded for insertion in a Statement of Work or a System Specification, that mandate the use of the various elements of modern software engineering.

General Introduction

The historical attributes of the software acquisition process addressed in this presentation are the very high cost of software development and the difficulty of maintaining or modifying the software after it's operational.

The history of government software system acquisitions is liberally spread with financial horror stories. Projects die in midstream when costs from overruns and unforeseen developments occur. And even those projects that finally fly, are often burdened with costs that are prohibitive. Historically, the systems acquisition organizations focussed on the hardware portion of the system and the software just happened incidentally. It is only recently that we have become aware of the preponderance of the software costs and the immediate need for doing something about them.

We, at RADC, have compiled a Software Development Standard in an attempt to provide a partial solution to the problem. No single document can completely cure the situation, but we feel that the use of this standard will move us a giant step forward in the right direction.

This Standard will attempt to present a comprehensive set of engineering techniques, procedures, conventions, and restraints that we believe are necessary for the generation of high quality, highly reliable software. The standard's contents consists of items that are thought of, collectively, as Modern Programming Practices.

A software document of this kind presents a new way of doing business for the government. Traditionally, the government would specify all requirements of the hardware in the system and leave the software to the discretion of the contractor. One or two sentences describing the functions of the software were considered sufficient to deal with the software aspect of a system.

The imposition of a specification of this nature changes all that radically. We feel confident that we know enough about software development to mandate that a set of tools and procedures be applied to an ongoing effort. We know that telling the contractor how he should design his programs in general, how he should restrain code and the selection of the language used, how he should keep track of the software development status, etc. will result in more reliable timely software. Moreover, hardware produced under such restraints will prove more amenable to debugging, modification, and eventual expansion without having to start all over again. All of the above will result in reduced costs.

It often happens that smaller than system projects which combine hardware and software are managed by an engineer who might be fully competent in radar technology, for instance, but is totally innocent of the correct way to buy software. Having a Standard from which he can select appropriate procedures and tools for use in his effort will be of great help.

We, as a laboratory with several sections specializing in the state of the art in software engineering are in a position to know how a software system can be properly developed. And putting all the various aspects of the task in one document provides a focus for modern programming technology.

Since this Standard appears now, long after many software houses have built up impressive inventories of tools and procedures in-house or on other government contracts, provisions have to be

made for allowing their use wherever possible in satisfying the mandated requirements. This is simply done by stating the requirements as functionally as possible. Any proposed tool that will satisfy the required functions is considered acceptable.

Pertinent System Life Cycle Factors

The proposed AF Software Engineering Standard (AFSES) will cover that portion of a system life cycle beginning early in the requirements phase when the software portion of the system has been defined and the requirements for the software are being generated. It affects elements in the design, coding, test, and integration continuing on through the deployment phase.

Although this defines the area over which the Standard is actually operative, its effect and resultant benefits extend on through the operation and maintenance phases of the system. Ease of maintaining and ease of modifying the software resulting from techniques mandated by the Standard continue to pay dividends throughout the life of the system. Attributes inherent in the design and coding of the software system produced yield cost and time advantages in every phase of the life cycle subsequent to their implementation.

Ongoing studies are being conducted at RADC on the applicability and validity of various software engineering methods, procedures, and tools. Data repositories are being developed to serve as historical data from which insights can be gained into fruitful areas of research.

Maintaining the Competitive Environment

In the overall world of system acquisition, the ever-present element of maintaining a free competitive environment is nowhere more complex than the software portion of the system. How do you mandate engineering requirements, requirements that tell the contractor how to do things, without inhibiting his unique skills and inventiveness?

Furthermore if he cannot propose the superior fruits of his individual skills freely, how can he produce a winning proposal? If certain tools and procedures developed by a rival firm are mandated, why should he be burdened with the task of responding to a requirement that is clearly the invention of some rival corporation? Contractors are very reluctant to produce and use a software tool invented by a rival software house.

The problem is, "How much and what kind of direction should be given the contractor?"

The two opposite poles of this question are:

1. Leave the contractor a free hand to propose his own solution and methods for fulfilling the system requirements.
2. Specify exactly what and how the contractor should produce the system.

Between these two diametrically opposed poles lies a wide field of choice. History has clearly shown

that leaving the contractor a free hand and awarding the contracts to the lowest bidder is often the most expensive and time-consuming way to go. On the other hand, specifying everything not only inhibits the contractor's special talents and inventiveness, it also transfers the responsibility for the success of the software to the government.

Many approaches to handling this question have grown up and are in use at present. One typical approach is to suggest the techniques and procedures and let the contractor propose in detail the items you would like to see in the proposal. For example, here is an item from a current procurement document which guides the preparation of proposals; "Describe all software programming languages, practices, standards, methods and conventions."

The procuring agency wants the contractor to propose a higher order language, and he hopes the language suggested will be one of the languages officially approved by his agency. He hopes the contractor will propose top down design and implementation. He is looking for structured code or a very good reason why it shouldn't be offered. This leaves it all up to the contractor to realize what is implied. If all the contractors bidding on the job guess right and they all bid and price the same items, it will be entirely up to the evaluators to pick the best proposal based on the current state of the art and the peculiarities of the subject system. But it's not very likely that they all will guess right and forcing bidders to guess is a sure invitation to confusion at best.

Another problem is how to impose requirements without provoking legitimate objections from the contractors. Why should a contractor explicitly propose all the latest guarantees of timely, low-cost software development when many of them cost more at the beginning? It may push the proposal out of the running if the other contractors merely give lip service to the choice buzz words which will allow them to price the work as usual. This will let overruns and extended debugging as the project develops be paid for in engineering changes and adjustments where the costs don't show up on the competitive bids.

The only solution seems to be to require each item specifically so that the contractor must respond to it in detail. If the specified items have to be met by all the bidders, the costs remain competitive and the evaluators can examine the proposals for depth of understanding and clever useful items over and above those required. To overcome the problems of requiring devices developed by rival software houses, the requirements must be expressed in terms of their bare functions.

This aspect of procurement which tends to keep the original development contract cost down regardless of total life cycle costs is a source of resistance from systems procuring offices down to the project manager of a small procurement. Money is allotted to the software portion of the effort and the normal attitude is to get as much software for the

available dollar as possible and never mind spending anything extra on the support environment. The fact that money spent up front results in very real life cycle cost savings doesn't fit comfortably in the traditional scheme of doing business. A certain amount of education must take place before such resistance is overcome.

However, more and more responsible administrators in the System Project Offices (SPO) are coming to realize that it's worth it to pay more at the beginning to implement Modern Programming Practices thus upgrading the quality and reliability of the system. They are increasingly willing to specify the items that promise better, less troublesome software systems.

Components of the Standard and Their Use

The proposed Standard is composed of two principal parts. The first part consists of a set of sections stating the items required. They are carefully worded for as-is insertion in the various documents of a Request For Proposal (RFP) and are intended to go on contract. This part is referred to as the Specification part of the Standard. The second part consists of a Users Guide which contains instructions for using the standard and peripheral information useful to the SPO and the contractor. Thus the Specification contains the sections to be put on contract and the Users Guide contains instructions for tailoring the sections selected and criteria for selecting appropriate sections. Most of the Users Guide content is intended to support the software acquisition agency rather than the contractor.

The Standard is designed as a looseleaf document to allow for constant updating as research projects surface new techniques, devices, or procedures suitable for inclusion in the document. The entire concept is evolutionary to accommodate the most recent products in the software industry that will aid in producing more reliable and maintainable software. It will also serve as an effective device for transferring the latest technology since an item's appearance in the Specification will be immediately accessible to any user of the Standard. When the user references the Standard on his next procurement effort, he will see the new item in it and will decide whether or not it's appropriate for his present effort.

Besides improving the process of developing the software itself, the Standard will provide items that will improve visibility so necessary for effective management and control of the process throughout the development phase. The Program Support Library, for instance, provides management with updated status information on the developing software throughout the effort.

In an effort to avoid the common pitfalls of restrictive government specifications, the Standard contains two statements of overall philosophy. The first is that "justifiable negotiated waivers or substitutions of functionally equivalent devices or solutions shall be entertained." This is

intended to avoid the problem of what to do about proposing a device that a contractor already uses that does the same job as the item stated in the Specification. Reinventing the wheel on every proposal can prove costly and contractors could justifiably complain that the inventor of the device specified had an unfair advantage.

The second statement reads "The overall intent of this Specification is to mandate requirements as a minimum and shall not preclude exceeding this minimum." This is meant to encourage the contractor to propose functions over and above the stated requirements.

Taken together, the two statements provide the contractor with room for original or individual attributes he may have at his disposal.

The main body of the Specification part of the Standard consists of properly worded sections that can be used in the various documents that form a Request For Proposal (RFP). In addition to the document itself, there are presently eight appendices which contain expanded information relating to certain sections. One example of this is App. 80 which details procedures for handling classified material.

The general headings in the Specification are as follows:

- Programming Languages
- Software Development Tools and Procedures
- Programming Standards and Conventions
- Management Aids and Quality Control Data
- Software Security Procedures
- Documentation Standards
- Operating Systems and Utilities Services
- National Software Works (NSW)
- Appendices 10,20,30,...,80

Several sections are divided into sub-sections which allow the user to select the appropriate sub-section for his project depending on size, cost, criticality, or any other considerations. Each succeeding sub-section increases the scope of the requirements and includes the requirements of the preceding section.

For example 5.4.2 is the section dealing with Program Support Libraries. Under it there are

- 5.4.2.1 Manual PSL - The simplest form.
- 5.4.2.2 Basic PSL - which requires some software but the smallest subset of functions
- 5.4.2.3 Full PSL with management Data collection and Reporting and all functions pertinent to managing a software development effort.

Each succeeding sub-section requires more functions, is more costly, and provides a more comprehensive tool. The user chooses the proper sub-section for his project and disregards the others in the same section. This makes use of the document without tailoring impossible because all sub-sections, taken together, would pose apparent contradictions, or at least, redundancies.

The Standard is continually being updated as

research produces new or better items for use in developing more reliable software. As new items are developed, thoroughly tested, and approved, they will become candidates for insertion into the Specification. Therefore, the document is considered open-ended in the sense that it is in a perpetual state of evolution. It would be fatal to ever cast it in stone because that would freeze modern software engineering technology at some particular time frame.

The accompanying document, the Users Guide, contains instructions with examples on how to use the Standard. It also contains information on each section giving its rationale and in which document of the RFP the section should be placed. Any comments or information pertinent to some aspect of the Specification document that would not be appropriate on contract, goes into the Users Guide.

Management and Technical Performance

One of the major benefits derived from the use of this Standard is the greatly improved management visibility throughout the development phase of the project.

Management visibility is a growing concern in the field of software procurement. One indication of this is the recent development of the Computer Program Development Plan (CPDP) in the Air Force. This is a document that is included in the Request For Proposal (RFP) package. The CPDP queries the bidder on every aspect of software management. The bidder is forced to respond in detail stating how he will manage the software development. Many of the responses to the requirements of the subject Standard will be included in the CPDP.

Both the CPDP and the subject Standard are intended to augment rather than supplant the traditional management procedures of system software procurements. An effort was made to maintain compatibility with existing standards and Air Force regulations governing acquisition and support procedures such as AF Regulation 800-14, Acquisition and Support Procedures for Computer Resources in Systems, and MIL STD 483 Configuration Management Practices for Systems, Equipment, Munitions, and Computer Programs and its follow-up, MIL STD 490, Specification Practices.

All the configuration management and quality assurance procedures with their reviews and audits throughout the life cycle of the system are still intact. Serious thought has been given to how the new requirements would blend and mesh with the traditional activities of a procuring agency. We have been able to demonstrate that no serious disruptions result from the imposition of our new Standard.

Furthermore, the Standard's requirements are not intended to negate any useful tools and procedures the software houses formerly used. For example, the Program Support Library is meant to augment any procedures a software house traditionally used. Many times, the traditional tools fulfill all or most of the required functions. In such cases, all

that would be required would be the addition of any missing functions to the existing tools.

In response to concerns expressed by small software houses, RADC conducted a study on the impact of modern program engineering technology on the small contractor. The results of this study showed that even the smallest software houses using a reputable time-sharing service could easily adapt to the new requirements without lessening the constraints or the severity of the requirements in any way.

Expected Benefits/Implications for the Software Industry

Although it may appear, at first glance, that industry would display great resistance to the imposition of a Standard on them, the fact is that software houses are not really averse to the idea. Many of the responsible houses know that the tools and procedures required in the document are useful in the orderly development of a software system and this Standard gives them a chance to propose them and cost them in the proposal. Another definite benefit is that the more explicitly you can tell the contractor what you want done and how he can best do it, the more secure he feels. Many projects that get in severe difficulties are the result of the contractor guessing wrongly what the government agency really wanted.

As further research produces new tools and procedures, they will be included in the standard for use on procurements. In spite of its necessarily incomplete form, the Standard has been in use for over a year at RADC. Coordination with the office that produced the Specification is mandatory at RADC whenever a procurement contains software. It has been used unofficially on several large-scale weapons system acquisitions. The Electronic Systems Division (ESD) at Hanscom Air Force Base near Boston is currently preparing a version of this Standard that will be appropriate for use by their System Program Offices (SPOs). Our intentions are that the standard will eventually be coordinated with the other defense community agencies and will be proposed as a MIL-STD for use throughout DoD.