Should the DoD Mandate a Standard Software Development Process?

by Donald G. Firesmith

DOD-STD-2167 should be viewed from a historical perspective. The opinion seems to be that many software developers did not know how to properly develop software—otherwise, why would so many products of questionable quality be delivered overdue and overbudget? One DoD answer was to mandate a "proven" process and life-cycle and to establish "proven" default methods.

Like MIL-STD-1679 before it, DOD-STD-2167 is a process standard that "establishes a uniform software development process" and mandates "requirements to be applied during the development . . . of software in Mission-Critical Computer Resources." Unlike its predecessors, DOD-STD-2167 is a tri-services standard that will be required on many projects.

DOD-STD-2167 mandates the classic "waterfall" developmental life-cycle, and its basic process is strongly tied to that life-cycle. It also either mandates, or establishes as a default, certain specific methods, such as standard formal reviews, Program Design Languages (PDLs), Software Development Files (SDFs), and functional and hierarchical decomposition methods.

This approach has serious drawbacks. Before looking at problems, however, we should consider DoD's position relative to DOD-STD-2167. "For the government to maintain a common, single development process throughout a variety of software development projects, it is necessary to define and direct the methodology. . . . A contractor always has the option of proposing an alternative methodology in the SDP. . . . The intent . . . is to allow the contractor to propose what he believes to be the best development methodology."

The question is not whether the software development process, methods, and life-cycle mandated by DOD-STD-2167 are the best ones currently available, but rather whether any single process, method, or life-cycle, however general, should be mandated. Even the best ones rapidly obsolesce as our industry evolves. By singling out specific methods in DOD-STD-2167, DoD is ensuring that the standard will become obsolete sooner than is necessary. Since the standard cannot be easily and rapidly updated, this will certainly have major negative consequences on the development of DoD software.

The following general problems have been raised with regard to the process standard nature of 2167:

How-to Constraints. Although the "intent . . . is to permit any systematic, well-documented, proven software development methodology," a process standard must, by definition, contain how-to restrictions. By mandating a standard life-cycle and activities based upon the phases of this life-cycle, DOD-STD-2167 is no exception. It permits only those software development methods consistent with its mandated process. Other methods are permitted only if one either proposes the inconsistent methodology in the Software Development Plan (SDP) or tailors the requirements out, approaches that are difficult, with significant economic risk to the contractor.

Although it may be reasonable for the government to adopt a process standard to dictate how the government is to procure software, it is something else entirely to mandate, how contractors are to develop software. This is an improper how-to constraint, and DOD-STD-2167 is therefore in direct violation of the government's Acquisition Streamlining Directive, which states that "As a first priority, this Directive establishes policy for streamlining . . . contract requirements by: Specifying contract requirements in terms of the results desired, rather than how-to-design and how-to-manage. . . . 'a contractor's management systems, internal procedures, methods, processes, and data products shall be used instead of specifying other approaches unless the acquisition activity determines that the contractor's approach cannot satisfy the program needs.' It is clearly the contractor's, not the government's, responsibility to define the process, methods, and life-cycle model to be used to produce software.

One counter argument is that an important objective of the Software Standardization Program of the JLC was to establish a well-defined and easily understood software development process. DOD-STD-2167 was thus intended to be a process standard, was meant to dictate how-to constraints, and has fulfilled its intention. For years, while software has become an ever larger part of any system, the government has operated without a standard process for procuring software. This lack has

resulted in many failures and additional expense.

However, there is a great difference between a process standard for procuring software and one for developing it. As far as failures and additional expense is concerned, one can easily argue that the how-to constraints have not, and cannot, solve the software crisis, but rather add greatly to the cost of software.

Another counter argument is that the contractor is free to propose alternatives so long as they are specified in the SDP and is not disapproved by the contracting agency. If the contracting agency is truly interested in innovative approaches, the requirements and defaults of 2167 do not necessarily prejudice them.

Although defaults may not unduly influence all contracting agency personnel, many developers are convinced that this is a real problem with some personnel. Thus, the ability to propose an alternative is an insufficient loop-hole since many contractors will feel pressured to comply with the standard to win contracts.

Another counter argument is that default approaches are needed for those instances where the contractor has not defined a software development process. This is clearly specious because paragraph 5.1.1.3.c.1 of DOD-STD-2167 already requires the contractor's proposed software development methods and techniques to be documented in paragraph 10.2.5.1 of the Software Standards and Procedures Manual (SSPM). Although not mentioned in the text, the same requirement is redundantly stated in paragraph 10.2.7.1.1 of the DID for the SDP. Besides, when a contractor does not propose a process or methods, the SSPM and SDP should be rejected as noncompliant rather than mandating a default approach that may well not be appropriate.

Innovation Inhibition. Process standards inhibit innovation. The perceived political and economic risks of proposing anything different from what the contracting agency expects, create a natural tendency for contractors to hesitate proposing software development processes, methods, and life-cycles that significantly deviate from those of 2167, regardless of their technical merit. Managers of several companies have already mandated compliance with DOD-STD-2167 and use this policy as part of their marketing strategy. Yet innovation is necessary and must be promoted in a rapidly evolving industry in which major improvements are necessary for solving the software crisis, software engineering advances come almost daily, and our foreign competition excels in technology insertion. It is not enough that some RFPs state that an innovative methodology is favorably considered in the contractor selection criteria. DOD-STD-2167 must also encourage innovation.

To quote General George S. Patton, Jr.: "Never tell anyone how to do something. Tell them what needs to be done. They will surprise you with their ingenuity."

The counter argument has been raised that innovation is not for large systems acquired with taxpayer dollars, that the software development process, methods, and life-cycle used should be well-understood, generally accepted, and have withstood the test of time. Thus, DOD-STD-2167 "incorporates practices that have been demonstrated to be cost-effective from a life-cycle perspective." New methods and life-cycles should first be proven on research and prototype projects.

However, as systems grow larger and more complex, software development processes, methods, and life-cycles do not usually scale up. Are we to believe that projects such as the SDI are best accomplished using a process little changed since the early 70s?

Standard Obsolescence. Most of DoD (certain advanced R&D efforts excluded) will always be technically several years behind industry, and further behind the research community. Thus it is vital that the contractor be encouraged to apply the methods it believed most appropriate for producing the product and associated documentation at the least possible cost, while still providing the government adequate oversight into the development effort.

DOD-STD-2167 was developed using a process that ensures, through numerous government and industry reviews, that the standard is acceptable to the majority of those who must use it. While laudable in principal, this concensus nature of DoD standards development is not without disadvantages when applied to a process standard. Some companies take conservative approaches to software development, while others use advanced methods. Any specific process acceptable to the majority of industry and government must therefore lag significantly behind the state-of-the-art. Thus, the JLC Software Standardization Program objective of integrating modern methods of developing software into DOD-STD-2167 is probably both inappropriate and impossible.

Acquisition Process Drivers. The basic process mandated by DOD-STD-2167 and the remaining standards is partially based upon the DoD acquisition process, which was historically developed to support hardware rather than software acquisition. As a consequence, the basic life-cycle and review process is not necessarily consistent with the most modern ways of developing software. This becomes especially important in light of DoD's recent realization of the prime importance of software development to systems development.

Process Inappropriateness. Because the best software development process, method, and life-cycle

are clearly application and implementation language specific, it is surely counterproductive for the DoD to choose any specific ones as either requirements or defaults (and therefore preferred).

The following arguments have been raised in favor of keeping DOD-STD-2167 a process standard:

- The lack of a standard software development method complicates the training of government personnel and leads to problems when personnel frequently transfer and are unable to apply their knowledge from the old project to the new. A standard process allows the government to train managers and technicians to work effectively with a contractor on a project. The best contractor-proposed method will not result in better software if the government does not understand it.

These advantages are probably the strongest arguments for a process standard. However, any advantages that the government would gain from maintaining "a common, single development process throughout a variety of software development projects" would be outweighed by the resultant inhibition of innovation.

- Some feel that there is not yet sufficient justification for allowing the contractor to alter such fundamental concepts such as the DoD acquisition process and the DoD established software development process, methods, and life-cycle model.

However, DoD did not introduce the Acquisition Streamlining directives without valid reasons. Innovation is necessary for the continued growth of the defense software industry. Besides, a small number of contractors do this via tailoring now.

- The contractor may lack the expertise to propose and implement an alternative process, method, or life-cycle. However, if that is true, the contractor should not be developing software.

- The government is probably not qualified to evaluate contractor-proposed processes, methods, or life-cycles. Even when contracting agency personnel are qualified, the evaluation of alternatives would be subject to argument. However, just as contractor personnel must keep up with rapidly evolving technology to remain competitive, government personnel must do so also. One hardly expects government personnel familiar only with vacuum tube technology to manage and maintain modern computer systems, and what applies to hardware applies equally to software. If the government is not qualified to evaluate contractor-proposed processes, methods, or life-cycles, then it should hire an independent expert.

- Because only the classic software development process, methods, and life-cycle mandated by DOD-STD-2167 has been proven to work, any alternative contractor proposed approach involves an unacceptable risk. The government needs to feel comfortable that the project will succeed prior to spending large amounts of money.

However, as previously mentioned, it is not at all clear that the classic approach is the only one proven to work. In fact, one can argue that has often failed to ensure DoD goals. No approach is without risk, and one must wonder how often the risk of trying something new is rejected for technical or economical reasons, and how often the reasons are psychological and social.

- Although the government is comfortable with the software development process mandated in DOD-STD-2167, it will not restrict other methods if the contractor clearly shows during proposal evaluation increased cost effectiveness and/or lower risk. All other factors being equal, a contractor who proposes a clearly better method should win.

However nice this is in theory, things are not always as they should be in practice. Many contracting agency managers are not aware of current trends in software engineering and are unwilling to take what they perceive as unnecessary risks. Even when they look favorably on innovative solutions to their problems, industry managers may well not be willing to propose something other than what DOD-STD-2167 requires or has as a default.

- It is the contractor's responsibility to ensure that the process used complies with government standards for software development.

One can argue, however, that the contractor has a higher duty to propose the best approach possible. Giving the contracting agency what it expects is not always in DoD's best interest. The contractor should feel free to use its expertise to propose the best solution to the problems.

- The government does not usually pay industry to develop new methods unless the new methods produce lowered cost, less risk, increased quality, etc. However, the government should encourage industry to use new methods already developed and not inhibit industry from the development of new ones.

- The government must define phase boundaries and milestones to be able to manage the software development process. However, it does not follow that any single specific set of phase boundries and milestones is optimal for all projects. Nor does it follow that the government is best able to define them. It is the government's responsibility to manage the software acquisition process and industry's to manage the software development process.

- The situation is no different than it was with MIL-STD-483, 490, and 1679. Though true, this is hardly a justification for keeping a less than perfect status quo. **DS&E**

In Part II, next month, we will look at specific problems.