IEEE/EIA 12207 AS THE FOUNDATION FOR ENTERPRISE SOFTWARE PROCESSES

James W. Moore
The MITRE Corporation
1820 Dolley Madison Blvd., W534
McLean, VA 22102, USA
Work Phone: +1.703.883.7396
Fax: +1.703.883.5432
James.W.Moore@ieee.org

Abstract

It is widely believed that the adoption of predefined processes aids the productivity and quality of software development. Furthermore, it is widely believed that the adoption of these processes at the enterprise level provides advantages of repeatability and organizational maturity that amplify the benefits. Unfortunately, the existing corpus of software engineering practice standards has been targeted for adoption at the project level rather than the enterprise level. A new standard, IEEE/EIA 12207, *Software Life Cycle Processes*, addresses this problem—it is intended as an integrating, organizing, strategic standard specifically directed to enterprise adoption and intended to form the foundation for the improvement of enterprise processes through the adoption of related standards.

KEYWORDS: Software engineering, standards, process, capability maturity, process improvement

Biography

Jim Moore is a twenty-nine-year veteran of software engineering and a ten-year veteran of software engineering standardization. With degrees from the University of North Carolina and Syracuse, he has worked in both the commercial and defense sectors for IBM and, now, The MITRE Corporation, where he is the corporate focal point for standardization activities. Currently, he serves as the chairman of the international standards committee for the Ada language, as a member of the Management Board of IEEE Software Engineering Standards Committee (SESC), as a member of the IEEE Standards Board, and as the Chair of the U. S. delegation to the international standards committee responsible for software engineering. He served for four years as a member of the Defense Department's Federal Advisory Board on Ada. He is a Senior Member of the IEEE and the IEEE Computer Society has recognized him as a Charter Member of their *Golden Core*. His book, *Software Engineering Standards: A User's Road Map,* was published in 1997 by the IEEE Computer Society Press.

IEEE/EIA 12207 AS THE FOUNDATION FOR ENTERPRISE SOFTWARE PROCESSES

James W. Moore
The MITRE Corporation

1 Introduction

A project manager desiring to adopt a sound set of processes for software development faces a daunting task. A large number of important issues inevitably influence the definition of the needed software engineering processes. Some of these influences include:

- The *project management practices* of the enterprise and/or the customer.
- The *systems engineering methods* being used for the containing system.
- Specific *process requirements* levied by the customer.
- *Total Quality Management* practices imposed by the enterprise, by the customer, or by choice.
- *Governmental regulations* applicable to the software and/or system.
- *Safety, security, and privacy requirements.*
- *Best practices* selected by the development team.
- *Contractual requirements*.
- The organization's competitive need for satisfactory *capability evaluations.*
- Tooling and other *infrastructural constraints.*
- The *corporate initiatives* du jour.

Confronted with so many important sources of “help,” it is little surprise that many project managers choose to “get on with the job” rather than pay attention to careful process definition. The result, of course, is that many software development projects utilize ad hoc processes implemented and modified on the fly by capable and not-so-capable developers.

The ultimate contribution to the project manager’s problem would be to take away the problem entirely. We would hope that any given project could, with minimal customization, execute a set of processes already defined at the enterprise level. Besides the head start given to the project, this approach offers important advantages to the organization: repeatable execution, portability of staff, common infrastructural support capabilities, the potential for organizational improvement. One approach might be the application of voluntary standards for the practice of software engineering.

Software engineering standards cover a wide scope of subjects. Although they sometimes deal with supporting tools or product characteristics, the most familiar ones are process standards dealing with subjects like Configuration Management, Verification, Validation, and Quality Assurance. Unfortunately, the erstwhile user of these standards is presented with a huge problem in selecting the ones appropriate for usage because more than 315 different standards, guides, handbooks, technical reports and other normative documents are offered by nearly 50 different professional, sector, national, and international organizations [Magee97]. Some sources cite even higher numbers. The two most important collections are provided by ISO/IEC JTC1/SC7[1] and the Software Engineering Standards Committee (SESC) of the IEEE Computer Society.

After more than a decade of focus on organizational process improvement, it is generally accepted that a software organization is best served by the adoption of a single, uniform set of processes repeatedly executed by all of its projects. Nevertheless, most software engineering standards have not yet caught up with this development. They continue to address the subject of process adoption by the individual project. Conformance is evaluated at the project level. Although there is nothing inherently wrong in this approach, it ignores both the investment and the benefit in adopting a sound and comprehensive set of enterprise processes. Needed is a new approach that gauges conformance by the enterprise-level adoption of suitable processes and their implementation through organizational procedures, practices, and infrastructure. In this view, the enterprise conforms to the standard process and the project conforms to the enterprise process. IEEE/EIA 12207 is a key standard for this use.

2 IEEE/EIA 12207, Software Life Cycle Processes

IEEE/EIA 12207, *Software Life Cycle Processes*, benefits from a heritage spanning a quarter-century of work on software process standards. Distant ancestors include the pioneering Department of Defense (DoD) standards better known by number than by name—Mil-Std-1679 and DoD-Std-2167. More recently, a cross-DoD working group developed Mil-Std-498 to replace DoD-Std-7935A, NSA 1703 and DoD-Std-2167A. When DoD policy deprecated the use of defense-specific process standards, a joint working group of the IEEE and the Electronic Industries Association (EIA) converted much of the content of Mil-Std-498 into a commercial standard, EIA/IEEE J-Std-016, *Software Development—Acquirer/Supplier Agreement.*

Meanwhile, ISO/IEC JTC1/SC7 was working on the standard that eventually became ISO/IEC 12207. Although Mil-Std-498 and an existing IEEE process standard, 1074, *Developing Software Life Cycle Processes*, were inputs to the effort, the result is quite different from both of them.

The developers of the ISO/IEC 12207 standard succeeded in developing a standard remarkably different from its predecessors. It establishes a common framework for software throughout its life cycle from conception through retirement and addresses the organizational context of those software processes both from the technical viewpoint of the system and the business viewpoint of the enterprise. The standard is widely regarded as providing a basis for world trade in software services; adoption of the standard is completed or underway in most of the major countries of the world. Next to the ISO 9000 series, it is probably the most important information-technology-related standard ever written.

2.1 Key Concepts of ISO/IEC 12207

The 12207 standard made many important innovations. (Only a few are selected for treatment here.) Most importantly, 12207 is defined at the level of *process* rather than *procedure*. Rather than provide the step-by-step requirements characteristic of a procedure, it describes *continuing responsibilities* that must be achieved and maintained during the life of the process. The standard addresses the *functions* to be performed rather than *organizations* to execute them. (For example, the standard describes a quality assurance process; this does not imply that a conforming enterprise must establish a quality assurance department.) The standard describes the development, maintenance and operation of software within the context of the system, thus effectively establishing the *minimum system context* essential to software processes.

The processes of ISO/IEC 12207 are described in three categories:

[1] Subcommittee 7 (responsible for software engineering) of Joint Technical Committee 1 (responsible for information technology) formed under a cooperative agreement of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC).

1. *Primary processes* are executed by parties who initiate or perform major roles in the software life cycle. They include the business roles of:

- Acquisition
- Supply

and the technical roles of:

- Development
- Operation
- Maintenance

2. *Supporting processes* contribute to the execution of other processes as an integral part with distinct goals. They include:

- Documentation
- Configuration management
- Quality assurance
- Verification
- Validation
- Joint review
- Audit
- Problem resolution

3. *Organizational processes* inherently exist outside the scope of the individual project but instances of them are employed by the project. They include:

- Management
- Infrastructure
- Improvement
- Training

From the viewpoint of this paper, ISO/IEC 12207 makes many important improvements, but has a key problem—it still addresses conformance at the project level. When the joint IEEE/EIA working group developed the industrial adaptation of 12207, it tackled that problem and added provisions for enterprise-level conformance. The resulting document, IEEE/EIA 12207, is now ready for use.

2.2 Key Improvements made in IEEE/EIA 12207

The joint working group responsible for the IEEE/EIA adoption of the international 12207 standard desired to make some important improvements while retaining a clear relationship to the international standard. They chose to achieve this goal by embedding the text of the international standard within the IEEE/EIA standard. IEEE/EIA 12207.0 ("Part 0" of the 12207 standard) is a "sandwich" with the complete text of ISO/IEC 12207 in the middle and additional material on the outside, as shown at the left side of Figure 1.

The IEEE/EIA project added two additional guidance parts to the standard. Part 1 provides guidance on life cycle data. It is cross-referenced to the provisions of Part 0. Part 2 provides guidance on the implementation of processes. It quotes the complete text of Part 0 and intersperses guidance notes.

The most important improvement in the IEEE/EIA standard is a set of alternatives for conformance with the standard. Rather than addressing only the project, four situations are addressed:

- *Enterprise*: The enterprise adopts the standard and implements policies, procedures and infrastructure to implement the provisions of the standard.
- *Project*: Either the project directly conforms to the standard or the standard conforms to enterprise processes that, in turn, conform to the standard.
- *Multi-supplier program*: A program adopts the standard and, viewed as a whole, conforms to the provisions of the standard regardless of whether individual suppliers qualify for conformance.
- *Regulatory program*: A program conforms to a tailored version of the standard provided by the regulator.

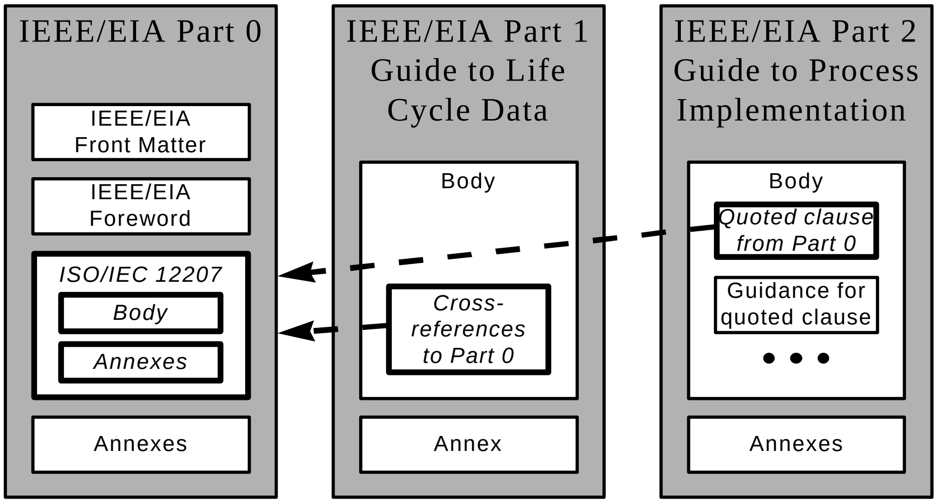
IEEE/EIA Part 0
IEEE/EIA Front Matter
IEEE/EIA Foreword
ISO/IEC 12207
Body
Annexes
Annexes
IEEE/EIA Part 1 Guide to Life Cycle Data
Body
Cross-references to Part 0
Annex
IEEE/EIA Part 2 Guide to Process Implementation
Body
Quoted clause from Part 0
Guidance for quoted clause
• • •
Annexes

Figure 1. Structure of IEEE/EIA 12207

Part 1 also provides cross-references to other IEEE standards that may be helpful in implementing the provisions concerning data. For instance, a user might choose to adopt IEEE Std 1016, *Software Design Descriptions*, to detail the data provisions related to the information item for software item description. Working in the other direction, the SESC has recently supplemented many of the referenced IEEE standards with a content map describing the extent to which the standard satisfies the data provisions of Part 1. Within the next few years, the content of each of the standards will be revised so that it fully conforms to the relevant provisions of Part 1.

In overall terms, SESC has adopted policy designating 12207 as a strategic, integrating standard for its collection. All of the relevant standards of the SESC collection will be revised to improve their fit with 12207; in particular, many of them will serve to detail the processes of the 12207 framework. From the process viewpoint, IEEE/EIA 12207 will serve as a single entry point to all the standards of the IEEE software engineering collection.

3 Contextual Processes and Standards

One of the complications in adopting enterprise-level software processes is that software engineering does not exist in isolation from other information technologies. Apart from the obvious influences of computer science and the various application domains, software engineering exists in the context of more general disciplines such as *project management*, *systems engineering* and *quality management*. Furthermore, software engineering projects are profoundly affected by cross-cutting disciplines like *dependability* and *safety*. Implemented software engineering processes must be consistent with the processes demanded by these other disciplines.

The good news is that SC7 and IEEE SESC are taking the necessary steps to coordinate their collections with other important disciplines. One important example—quality management—is discussed here. (A more extensive treatment of these and other examples can be found in the guide to the SESC collection, [Moore97].)

The ISO 9000 series is the most successful example of an information-technology-related standard in the brief history of the information age. World-wide, more than 135,000 organizations claim conformance [Kiang97], more than 80 countries have adopted them as national standards [Peach94], and more than 50 countries actively participate in developing the standards [Kiang97]. For an enterprise claiming or desiring ISO 9000 conformance, it makes no sense for their software engineering processes to remain unsupportive of that goal. Nevertheless, until recently, the relationship of software engineering to quality management was described by a guide, ISO 9000-3, whose relationship to the relevant standard, ISO 9001, was murky and whose references to software development were regarded by some as uninformed.

Fortunately, that problem has been solved by the recent publication of the 1997 revision to ISO 9000-3. That guide is now organized identically to ISO 9001; it quotes every relevant passage of 9001 and explains the passage in terms of the requirements levied upon software processes. Furthermore, each clause explicitly references the relevant clauses of ISO/IEC 12207, establishing a firm relationship between software and quality processes. (The guide to the SESC Collection provides additional cross-references to IEEE standards that may be useful in detailing the relationships.)

4 Adopting Enterprise Level Software Processes

4.1 Process versus Procedure

Before discussing the enterprise level adoption of software processes, it is important to emphasize the distinction between a process and a procedure. I found the most eloquent description of this distinction in an unpublished report of the Rand Corporation:

> A *procedure*...is a series of steps. When you have completed the steps, you have completed the procedure. This is the only guarantee of a procedure. It is unrelated to the ostensible objective of the procedure.
>
> A *process*, on the other hand, has principles that contribute to an objective. Everyone involved must understand and look at the contribution to the objective and use the principles to guide him rather than following them blindly as in a procedure.

The "principles" mentioned in this quotation are what I have termed "continuing responsibilities" elsewhere in this paper. The distinction between process and procedure is at the heart of the fundamental difference between 12207 and its ancestor process standards. The distinction is also fundamental to the framework for process adoption described below.

4.2 Basili Model for Process Descriptions

In performing his work on the "component factory," Victor Basili needed to provide appropriate process descriptions. He and his colleagues found [Basili92, Heineman94] that three different forms of description seem useful:

- The *Reference* view describes a process as a coherent, cohesive sets of activities that can sensibly be performed by a single *agent*. The 12207 standard fills this role in enterprise adoption.
- The *Conceptual* view describes flow of control and data among the *agents*. For enterprise adoption, additional standards may be adopted to detail the processes of 12207 and to fill this role.
- The *Implementation* view maps the *agents* to the organization chart and selects policies, procedures and tools to implement the processes. For enterprise adoption, this role is filled by various corporate standards.

4.3 A Framework for Enterprise Process Adoption

Minding the distinction between process and procedure and applying the Basili model leads to the framework for process adoption depicted in Figure 2.

The first step is to select a life cycle process standard for enterprise adoption. Although IEEE/EIA 12207 is the obvious choice, there are alternatives that may make sense for specific industry sectors including space, nuclear, airborne and legacy defense systems.

The next step is to analyze the needs of the enterprise to identify contextual processes and requirements that may affect software engineering. In performing the analysis, one should look for organizational strengths to build upon as well as organizational weaknesses that should be remedied. One should consider:

- *Organizational imperatives*—for example, ISO 9000 conformance.
- *Organizational competencies*—for example, systems engineering or project management.
- *Industry sector characteristics*—for example, requirements for dependability or safety.

Each of the cited examples has a related body of standards (see [Moore97]) that should be considered when planning enterprise-level processes. Armed with the selected body of standards, one can now describe enterprise level processes at the reference level. In many cases, this description can be accomplished simply by referring to appropriate standards, perhaps with auxiliary explanation of areas where the selected standards may overlap or leave gaps.

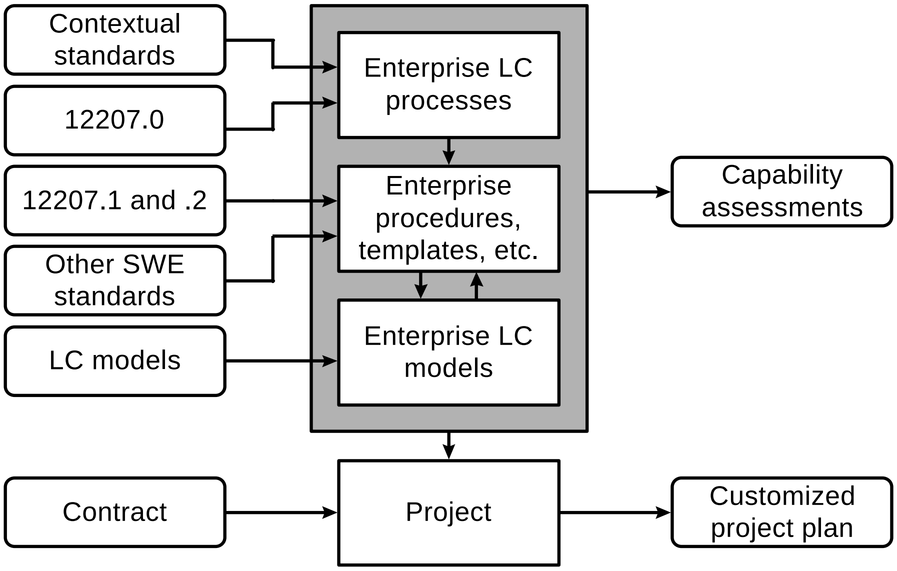

Figure 2. Overview of Enterprise Process Adoption.

The next step is to detail those processes, in part through the selection of additional standards. Parts 1 and 2 of IEEE/EIA 12207 can be helpful in this area. Both of those parts reference related standards. The guide to the SESC collection is intended to support this selection.

IEEE/EIA 12207 and most of the other standards are independent of any particular life cycle *model*, such as evolutionary development or waterfall. Mapping of enterprise processes to a selected life cycle model may be left to the individual project. Infrastructural support, though, is likely to be more effective if a set of models is selected and implemented at the enterprise level.

Given all of these resources, the real payoff comes from the implementation of procedures, templates, tools, training and other infrastructural support at the enterprise level. These resources enable repeatable execution of processes, portability of staff and the resulting organizational improvement.

Finally, of course, enterprise policies must be established to motivate usage of the enterprise processes, to systematically monitor execution for desirable changes, and to complete the cycle of organizational improvement.

5 Summary

In the past, available software engineering standards have not been designed to support enterprise-level adoption of processes. The new IEEE/EIA 12207 standard breaks new ground by defining software life cycle processes as a set of continuing responsibilities and by permitting claims of conformance to be made at the enterprise level. Besides adopting 12207 as a strategic, integrating standard, the IEEE Software Engineering Standards Committee has fitted its collection of nearly 50 standards into an overall framework and has sponsored a guide to describe the organization of the collection.

An organization seeking to implement enterprise level processes should supplement 12207 with other standards supporting contextual goals, such as quality management and dependability. The desired processes can be selectively detailed with additional standards and mapped to an inventory of life cycle models. The provision of additional procedures, templates, and tools enables an enterprise to provide a robust process capability to project managers.

List of References

[Basili92] Basili, Victor R., et al. “A Reference Architecture for the Component Factory.” *ACM Transactions on Software Engineering and Methodology* Vol 1 (1992): 53-80.

[Heineman94] Heineman, G. T., et al. “Emerging Technologies that Support a Software Process Life Cycle.” *IBM Systems Journal* Vol 33 (1994): 501-529.

[Kiang97] Kiang, David. “ISO 9000 Update.” presentation to Society of Reliability Engineers, Ottawa, Canada, 25 Feb 1997.

[Magee97] Magee, Stan, and Leonard L. Tripp. *Guide to Software Engineering Standards and Specifications*. Boston, MA: Artech House, 1997.

[Moore97] Moore, James W. *Software Engineering: A User's Road Map.* Los Alamitos, CA: IEEE Computer Society Press, 1997.

[Peach94] Peach, Robert W., ed. *The ISO 9000 Handbook*, (2nd Ed.). Fairfax, VA: CEEM Information Services, 1994.

Referenced Standards

EIA/IEEE J-Std-016:1995, [Interim] Standard for information technology—Software life cycle processes—Software development: Acquirer-supplier agreement

IEEE Std 1016-1998, IEEE recommended practice for software design descriptions

IEEE/EIA Std 12207.0-1996, Industry implementation of international standard ISO/IEC 12207:1995—Standard for information technology—Software life cycle processes

IEEE/EIA Std 12207.1-1997, Industry implementation of international standard ISO/IEC 12207:1995—Standard for information technology—Software life cycle processes—Life cycle data

IEEE/EIA 12207.2-1997, Industry implementation of international standard ISO/IEC 12207:1995—Standard for cesses—Implementation considerations

ISO 9000-3:1997, Quality management and quality assurance standards—Part 3: Guidelines for the application of ISO 9001:1994 to the development, supply, installation and maintenance of computer software

ISO 9001:1994, Quality systems—Model for quality assurance in design, development, production, installation, and servicing

ISO/IEC 12207:1995, Information technology—Software life cycle processes