Unit 1

IEEE/EIA 12207:1995 Software Life Cycle Processes

Prepared by:
James W. Moore, moorej@ieee.org
The MITRE Corporation
January 1998

Seminar themes (1 of 3)

- Nature of 12207
  - A *framework of related names and concepts* ... not necessarily all of the best practices for software
  - *Processes* ... not procedures
  - *Life cycle processes* ... not a life cycle model

Seminar themes (2 of 3)

- The best use of IEEE/EIA 12207 is *enterprise level adoption*.
  - It is intended for *voluntary adoption* rather than contractual imposition.
  - It emphasizes *specific one-party claims of compliance* rather than two-party tailoring.
  - It has *relationships to contextual standards* affecting enterprise goals.
  - It has *relationships to process and data standards* that may be used to implement its processes.

Seminar themes (3 of 3)

- IEEE/EIA 12207 is a *strategic, integrating standard* for the IEEE software engineering collection.
  - It provides a unifying approach to *life cycle process standardization*.
  - It provides a unifying approach to *life cycle data standardization*.
  - IEEE is now *improving the fit*.
  - IEEE plans to build upon the standard with *future strategic efforts*.

Part 1 - Software engineering

- *Software Engineering*
  - Definition
  - Model
  - Relationship to other Disciplines
- Software Engineering Standards
- Software Engineering Standards Developers

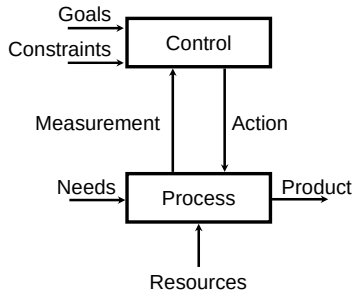
Software engineering: Definition

*(1) The application of a **systematic**, **disciplined**, **quantifiable** approach to the development, operation and maintenance of software, that is, the application of engineering to software*

(2) The study of approaches as in (1)

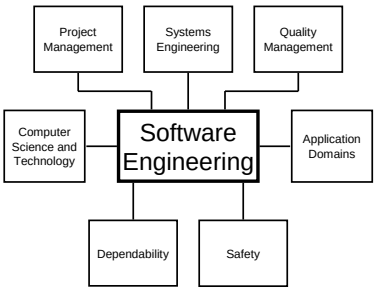
-- IEEE Std 610.12

Software engineering: A model



Source: [SESC93]

Software engineering: Relationship to other disciplines



Part 2 – Software engineering standards

- Software Engineering
- *Software Engineering Standards*
  - Scope
  - Importance
  - Roles and uses
  - History
- Software Engineering Standards Developers

Software engineering standards

- Approximately **315** software engineering standards, guides, handbooks, and technical reports are maintained by approximately **46** professional, sector, national, and international standards organizations. -- *[Magee97]*
- In 1981, IEEE had one software engineering standard. By year end 1997, the collection had grown to 44. -- *[SESC97a]*
- The 1994 edition of IEEE Standards Collection: Software Engineering is **1,300** pages long. The 1998 edition will be in four volumes totaling **2,400** pages
- Most software engineering standards are ***practice*** standards rather than the more familiar ***product*** standards

Software engineering standards: Scope

Process
- Acquisition
- Requirements definition
- Design
- Code and Test
- Integration
- Maintenance and Operations
- CM
- Documentation
- Project management
- Quality assurance
- V & V

Technique/Tool
- CASE tools
- Languages and Notations
- Metrics
- Privacy
- Process improvement
- Reliability
- Safety
- Security
- Software reuse
- Vocabulary

Applicability
- General
- Defense
- Financial
- Medical
- Nuclear
- Process control
- Scientific
- Shrink-wrap
- Transportation

Source: [Magee97]

Software engineering standards: Importance

- They consolidate existing technology into a firm basis for introducing newer technology
- They increase professional discipline
- They protect the business
- They protect the buyer
- They improve the product

Software engineering standards: Roles

More exciting

- Specify techniques to develop software faster, cheaper, better → IEEE 982.1 (Measures for Reliable SW)
- Provide consensus validity for "best practices" that cannot be scientifically validated → IEEE 1008 (Unit Testing)
- Provide a systematic treatment of "*ilities*" → IEEE 730 (SW Quality Assurance)
- Provide uniformity where agreement is more important than small improvements → IEEE P1320.1 (IDEF0)
- Provide a framework for communication between buyer and seller → IEEE/EIA 12207 (SW Life Cycle Processes)
- Give precise names to concepts that are fuzzy, complex, detailed and multidimensional → IEEE 1028 (SW Reviews)

More effective

Software engineering standards: Uses

- Terminology
- Best practice adoption
- Organizational badge
- Contractual agreement

Software engineering standards: Organizational goals

- Improve and evaluate software competence
- Framework for two-party agreements
- Evaluation of software products
- Assurance of high integrity levels for software products

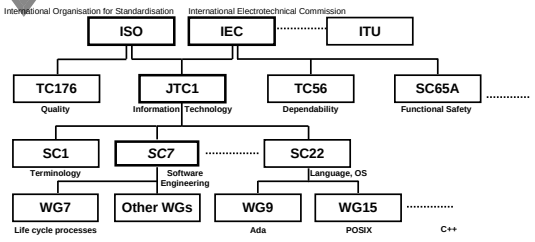
Software engineering standards: History

- 1968: Term *software engineering* coined at NATO conference
- 1973: US National Bureau of Standards writes *Guidelines for Documentation of Computer Programs and Automated Systems*
- 1974: US Navy initiates Mil-Std-1679, *Weapons System Development*, including guidelines for embedded computing resources.
- 1976: IEEE creates predecessor of SESC
- 1979: IEEE Std 730, *Software Quality Assurance Plans*
- 1987: ISO and IEC form JTC1 on Information Technology [Industry]
- 1998: JTC1/SC7 gains "horizontal" status

Part 3 – Software engineering standards developers

- Software Engineering
- Software Engineering Standards
- *Software Engineering Standards Developers*
  - International : ISO/JTC1/SC7 and others
  - US: IEEE and others

SWE standards developers: International



The focal point in international SE standards is **ISO/IEC JTC1/SC7.**

Other committees, though, deal with related work.

Members of these committees are "national bodies," i.e. countries, represented by "national delegations."

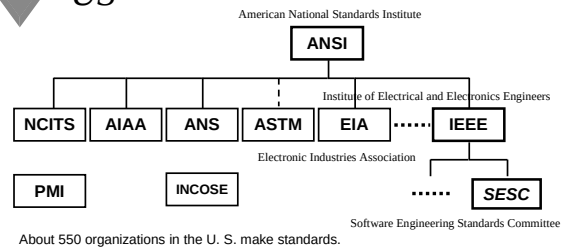
SWE standards developers: ISO/IEC JTC1/SC7 program

- WG2: System software documentation
- WG4: Tools and environment
- WG6: Evaluation and metrics
- **WG7: Life cycle management**
- WG8: Integral life cycle processes
- WG9: Classification and mapping
- WG10: Process assessment
- WG11: Software engineering data definition and representation
- WG12: Functional size measurement
- WG13: Software measurement process

SWE standards developers: Current standards of SC7

- Six "legacy" standards
- ISO/IEC 9126:1991, Product quality characteristics
- ISO 9127:1988, User documentation and cover information for consumer software packages
- ISO/IEC TR 9294:1990, Management of software documentation
- ISO/IEC 11411:1995, Representation of state transition diagrams
- ISO/IEC 12119:1994, Software packages: Quality requirements and testing
- ISO/IEC 12207:1995, Software life cycle processes
- ISO/IEC 14102:1995, Evaluation and selection of CASE tools
- ISO/IEC 14143-1:1997, Functional size measurement
- ISO/IEC 14568:1997, Diagram exchange language for tree charts

SWE standards developers: US

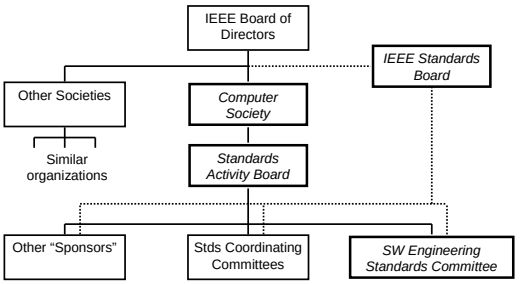


About 550 organizations in the U. S. make standards.

About half of them are accredited by **ANSI**, allowing them to participate in international standardization activity.

The focal point (in the U. S.) is the **SESC** of the IEEE Computer Society

SWE standards developers: IEEE



Unit 2

IEEE/EIA 12207:1995 Software Life Cycle Processes

Prepared by:
James W. Moore, moorej@ieee.org
The MITRE Corporation
January 1998

Unit 2 - ISO/IEC 12207

- *History*
  - Purpose of 12207
  - History of 12207
- Key concepts
- ISO/IEC 12207 processes

Purpose of 12207

- To establish a common framework for the life cycle of software
  - Acquire, supply, develop, operate, and maintain software
    - Undertakes broader scope than previous standards
  - Manage, control, and improve the framework
    - Recognizes that software is part of a system and that a project is part of an enterprise
- To establish a basis for world trade in software

Motivations for 12207

Previous standards ...

- … focused on the single contract or project in isolation
- … described a single monolithic process
- … failed to encourage investment in the discipline and capitalization of processes
- … induced 20-50% added costs in documentation and formal reviews

Adapted from a slide by Perry DeWeese

Unit 2 - ISO/IEC 12207

- History
- *Key concepts*
- ISO/IEC 12207 processes

Key concept of ISO/IEC 12207: Process/activity/task hierarchy

- Processes are subdivided into cohesive ***activities***
- Activities are subdivided into ***tasks***
- You may think of tasks as being the *specifications* for the execution of an activity
- A task may be a self-declaration of intent, a requirement, a recommendation, or a permissible action

Key concept of ISO/IEC 12207: Processes and parties

- Identification of processes is based on two principles:
  - ***Modularity***: Processes should be cohesive and should have *low coupling* to other
  - ***Responsibility***: Each process should be executable by a *single party*
  - A particular *organization* may become the party responsible for executing a process

Key concept of ISO/IEC 12207: Continuing responsibilities

- The activities and tasks of a 12207 process are not steps to be performed
- 12207 does not require that the activities and tasks are to be performed in any particular order
- The activities and tasks of 12207 are ***continuing responsibilities*** whose execution is assigned for the duration of the process

Key concept of ISO/IEC 12207: Categories of processes

- ***Primary***
  - Acquisition, Supply
  - Development, Operation, Maintenance
- ***Supporting***
  - Processes used as "subroutines" by other processes
- ***Organizational***
  - Processes inherent to the organization and "instantiated" by the project
- Also, a special ***tailoring*** process

Key concept of ISO/IEC 12207: Integral evaluation

- Evaluation is not a distinct process or activity
- Evaluation is treated as an ***internal***, ***integral*** task of many activities in the standard
- In general, the evaluations have stated purposes and stated criteria
- Other processes may supplement internal evaluations: Verification, Validation, Joint Review, Audit, Quality Assurance, Improvement

Key concept of ISO/IEC 12207: Temporal issues

- The standard does not specify a life cycle ***model***, e.g. waterfall, spiral, etc.
- The standard does not place ordering dependencies or time dependencies on the tasks → that is the job of the chosen life cycle model and the project plan
- Tasks may be iterated, repeated, recursively invoked, etc

Key concept of ISO/IEC 12207: Approach to documentation

- The standard requires some outputs to be documented
- The standard does not prescribe format, media, or content of the documentation
- The Documentation Process permits the user to make these decisions

Key concept of ISO/IEC 12207: Approach to baselining

- Standard differentiates between *items* and *configuration items* (CI). The Configuration Management (CM) process can handle both, but the latter are handled more rigorously
- A ***baseline*** is a formally approved version of a CI. Baselines (as clarified by the IEEE/EIA version) are established by the primary processes, not by the CM process

Unit 2 - ISO/IEC 12207

- History
- Key concepts
- *ISO/IEC 12207 processes*

ISO/IEC 12207 processes

- Primary
- Supporting
- Organizational

ISO/IEC 12207 processes: Primary processes

- Primary processes are executed by parties who initiate or perform major roles in the software life cycle:
  - Acquisition
  - Supply
  - Development
  - Operation
  - Maintenance

ISO/IEC 12207 processes: Supporting processes

- A supporting process supports another process as an integral part with a distinct purpose:
  - Documentation
  - Configuration Management
  - Quality Assurance
  - Verification
  - Validation
  - Joint Review
  - Audit
  - Problem Resolution
- But *fundamental responsibility* remains integral to the *primary* process

ISO/IEC 12207 processes: Organizational processes

- Organizational processes inherently exist outside the scope of the project but *instances* of them are employed by the project:
  - Management
  - Infrastructure
  - Improvement
  - Training

Global Harmonization of Systems and Software Engineering Processes

James W. Moore, F-IEEE, CSDP

The MITRE Corporation

May 2008

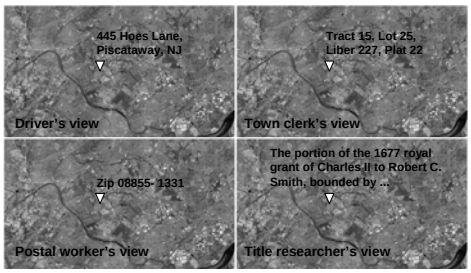
The authors' affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the authors.

Process Harmonization

- **ISO/IEC JTC 1/SC 7 (software and systems engineering) has a large collection of standards.**
  - **Some of the key process standards are difficult to use together.**
- **IEEE Software and Systems Engineering Standards Committee has a large collection of standards.**
  - **Some of the key process standards were adopted (and slightly modified) from SC 7.**
- **The goal of process harmonization is to:**
  - **Create a single definitive set of processes ...**
  - **... described in a set of standards that are easy to use together ...**
  - **... agreed and shared by both organizations.**
- **This presentation describes progress toward that goal.**

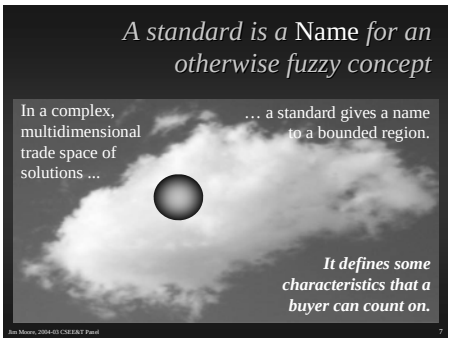
Names are Important

We use names to localize the subject under discussion. But sometimes confusion results because we use different name spaces.



Would you know that these are different names for the same thing? Would you know without the map?

Many Standards are Names



- **Many software engineering standards assign names to practices or collections of practices.**
- **This enables communication between**
  - **Buyer and seller**
  - **Government and industry**
  - **Insurer and insured**

Harmonization of 15288 and 12207

- **Two technically excellent standards are at the center of process harmonization:**
  - **ISO/IEC/IEEE 15288:2002, System life cycle processes**
  - **ISO/IEC 12207:1995, Software life cycle processes, and the substantially identical IEEE/EIA 12207.0:1997**
- **A large international user base wants standards that can be used together for development of systems with software content**
  - **However, the standards were difficult to use together.**
  - **Furthermore, a set of amendments to 12207 (for the process assessment community) compounded the difficulty.**
- **A two-step project is underway to "harmonize" the two standards.**
  - **The first step – aligning the processes – was completed early in 2008.**
  - **The second step – integration – will seek a single set of shared software/system processes.**
- **In addition, a companion project, 24748, is underway. It will provide a guide to life cycle management using the two standards.**

15288 and 12207 Give "Names" to Processes

- **ISO/IEC 15288:2002 gave names to 25 processes in the life cycle of a system. It is more *descriptive* than 12207.**
- **ISO/IEC 12207:1995 gave names to 17 processes in the life cycle of a software product or service. It is more *prescriptive* than 15288.**
  - **(Two amendments re-described the processes for assessment purposes.)**
- **The names are important so that acquirers and suppliers can communicate regarding their practices.**
  - **"Oh, when you say 'implementation', you include 'testing'? No, no, no, that's a separate thing; our contract doesn't include that!"**
- **The names are important as a basis for process evaluation and improvement.**
- **The names are important to provide a context for implementing *improved practices*. – Our goal.**

The Harmonization Problem

Starting from a diverse set of standards with …

- **different terms**
- **different process sets**
- **different process architectures**
- **different levels of prescription, and**
- **different audiences***

… develop a revised set of standards with …

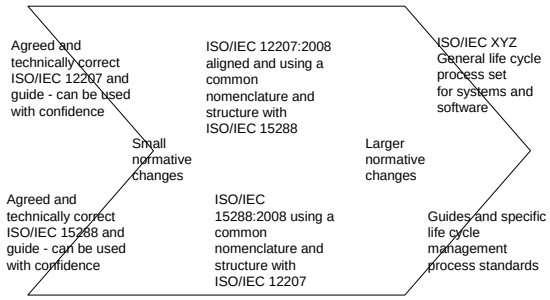
- **a single vocabulary**
- **a single process set**
- **a single, uniform architecture**
- **a shared level of prescription, and**
- **suitability across the audiences**

… all without needlessly disrupting current organizational investment based on usage of the current set of standards.

*** Systems versus software and process definition versus process assessment**

Overview of approach for Harmonization

Source: ISO/IEC JTC1/SC7 WG7 briefing material



The Harmonization Problem – Problems addressed in the Alignment Phase

Starting with a diverse set of standards with …

- **Different terms**
- **Different process sets**
- **Different process architectures**
- **Different levels of prescription**
- **Different audiences**

… develop a revised set of standards with …

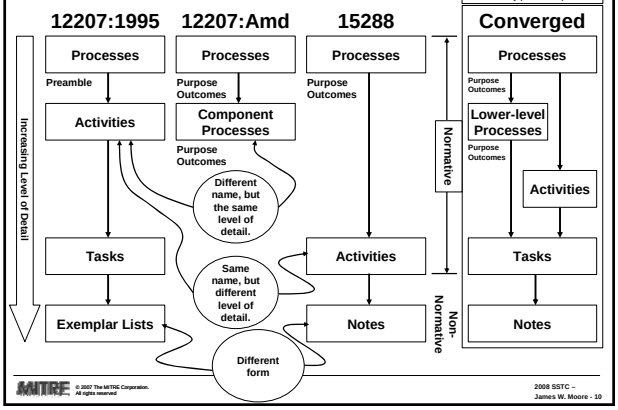
- **A single vocabulary** 😐
- **A single process set** 😊
- **A single, uniform architecture** 😊
- **A shared level of prescription** 😐
- **Suitable across the audiences** 😊

… all without needlessly disrupting current organizational investment based on usage of the current set of standards. 😊

*** System versus software and process definition versus process assessment**

Architectural Constructs

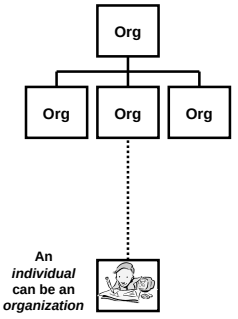
It is not required to use all architecture constructs in any particular document or for any particular process.

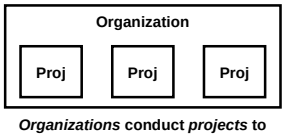


Key Terminology and Concepts

- ***Organization*: a person or a group of people and facilities with an arrangement of responsibilities, authorities and relationships [ISO 9000]**
  - **A part of an organization is an organization if it meets the definition.**
  - **An *individual* can be an organization if s/he meets the definition.**
- ***Party*: an organization entering into an agreement**
- ***Project*: an endeavour with defined start and finish dates undertaken to create a product or service in accordance with specified resources and requirements [adapted from ISO 9000]**

Key Terminology and Concepts



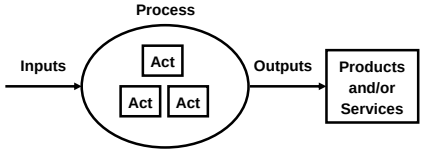


***Organizations* conduct *projects* to do things, notably to deal with *systems*.**

***Organizations* make *agreements* to *acquire* and *supply products* and *services*.**

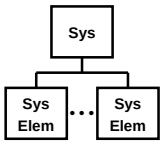
Agreeing *organizations* are called *parties*.

Key Terminology and Concepts



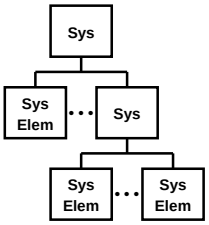
- ***Process*: set of interrelated or interacting *activities* which transforms inputs into outputs [ISO 9000]**
- ***Product*: the result of a process [ISO 9000]**
- ***Service*: performance of activities, work, or duties associated with a product**

Key Terminology and Concepts

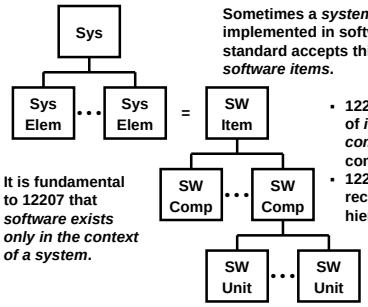


A *system* is composed of *system elements*. Each element is *implemented* and then *integrated* into the system. One invocation of 15288 suffices to create a system composed of a set of elements.

However, 15288 states that a system element can itself be regarded as a system. So, 15288 can be invoked recursively to create a hierarchy of systems and their elements. A hierarchy of systems often implies a hierarchy of projects.



Key Terminology and Concepts



Sometimes a *system element* is to be implemented in software. The 12207 standard accepts this as one or more *software items*.

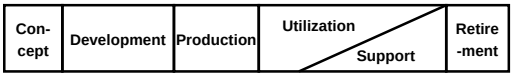
- **12207 uses a hierarchy of *items* – composed of *components* – composed of *units*.**
- **12207 is *not* invoked recursively to create this hierarchy.**

It is fundamental to 12207 that *software exists only in the context of a system*.

Key Terminology

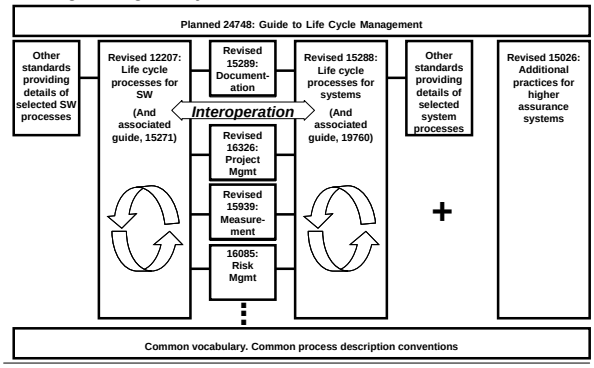
- **Every *system* has a *life cycle* which is viewed as composed of *stages*. (The standards do not require a particular set of stages.)**
  - **Each stage has a purpose and makes a contribution to the life cycle.**
- **Stages are initiated and terminated by *decision gates*.**
- **Stages may overlap and may be concurrent.**
- **The purpose of each stage is accomplished by executing *processes*.**
- **Any process may be useful in any stage.** This is important.

A typical set of life cycle stages



- **It is a common error to talk about life cycle stages when one really means processes or vice-versa.**
- **Locating practices with respect to processes provides much greater precision.**

Intended Relationships of Key System and Software Engineering Life Cycle Process Standards



Related Projects

- **24774 summarizes the agreed conventions for describing processes.**
- **15289 summarizes the data products produced by the processes of 15288 and 12207. It exists, but must be revised to deal with the 2007 revisions.**
- **Three standards provide additional details on selected life cycle processes: 15939, Measurement; 16085, Risk management; 16326, Project management.**
- **15026 provides additional practices for the assurance of systems and software when particular critical properties are required.**
- **24765 is a database of vocabulary that will occasionally be published as a conventional printed standard:**
  - **Database is publicly available at: http://www.computer.org/sevocab**

Example Process: 12207 Stakeholder Requirements Definition Process (1 of 2)

6.4.1 Stakeholder Requirements Definition Process

NOTE The Stakeholder Requirements Definition Process in this International Standard is a specialization of the Stakeholder Requirements Definition Process of ISO/IEC 15288. Users may consider claiming conformance to the 15288 process rather than the process in this standard.

6.4.1.1 Purpose

The purpose of the Stakeholder Requirements Definition Process is to define the requirements for a system that can provide the services needed by users and other stakeholders in a defined environment. …

6.4.1.2 Outcomes

As a result of successful implementation of the Stakeholder Requirements Definition Process:

a) the required characteristics and context of use of services are specified;

b) the constraints on a system solution are defined;

c) …

Example Process: 12207 Stakeholder Requirements Definition Process (2 of 2)

6.4.1.3 Activities and tasks

The project shall implement the following activities and tasks in accordance with applicable organization policies and procedures with respect to Stakeholder Requirements Definition Process

6.4.1.3.1 Stakeholder identification. This activity consists of the following task:

6.4.1.3.1.1 The project shall identify the individual stakeholders or stakeholder classes who have a legitimate interest in the system throughout its life cycle.

NOTE This includes, but is not limited to, users, operators, supporters, developers, producers, trainers, maintainers, disposers, acquirer and supplier organizations, parties responsible for external interfacing entities, regulatory bodies and members of society. Where direct communication is not practicable, e.g., for consumer products and services, representatives or designated proxy stakeholders are selected.

Etc.

Selecting which Standard to Use

- **Both 12207 and 15288 contain process models that are nearly identical:**
  - **The differences are rational rather than accidental.**
- **15288 describes the processes at the system level.**
- **12207 specializes the same processes to software, and adds processes specific to software.**

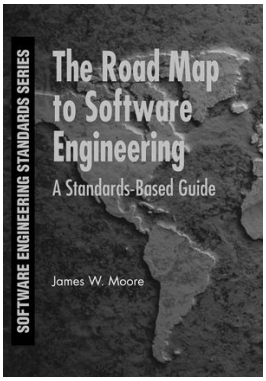
To deal with a system ...	**... use 15288.**
To deal with a software element of a system ...	**... use 15288 and the software processes of 12207.**
To deal with a software product or service (with minimal surrounding system) ...	**... use 12207.**

Questions?

For more information, contact presenter:

Jim Moore

moorej@mitre.org



John Wiley / IEEE Computer Society Press, 2006, ISBN-10: 0471683620, ISBN-13: 978-0471683629