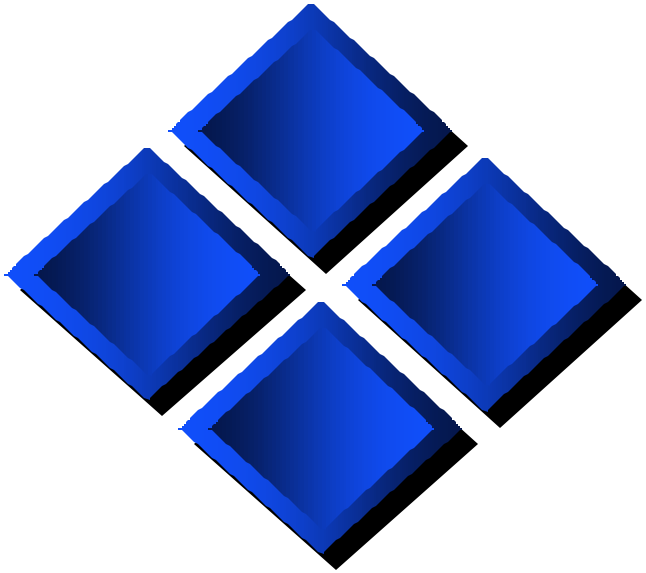


Best Practice Information Aids for CMMI[SM]-Compliant Process Engineering

Paul R. Croll

Chair, IEEE Software Engineering Standards Committee

Vice Chair, ISO/IEC JTC1/SC7 U.S. TAG

Computer Sciences Corporation

pcroll@csc.com

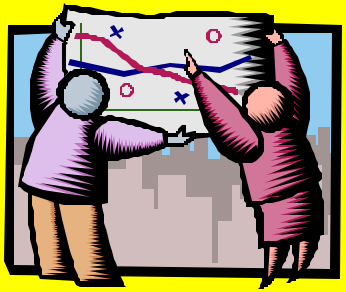
A Preparatory Exercise

How many of you know that there are two Framework Standards for System and Software Life Cycle Processes?

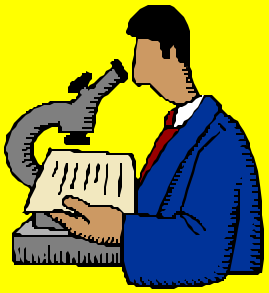
How many of you know that there are thirty+ Supporting Standards for Software and Systems Process Engineering?

8 Steps to Success With Best Practice Information Aids

1. Understand your business processes



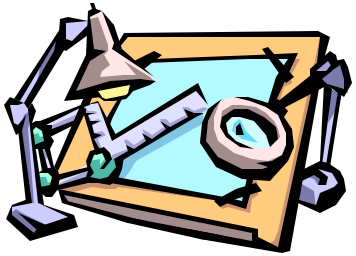
2. Look to the CMMI[SM] for Process Completeness



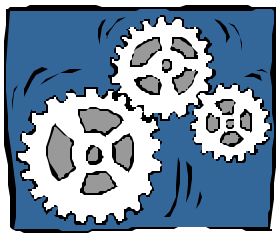
3. Look to Framework Standards for Life Cycle Definition

4. Look to Supporting Standards for Process Detail

5. Build or Refine Your Process Architecture

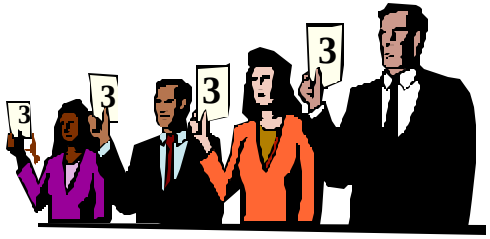


6. Execute Your Processes

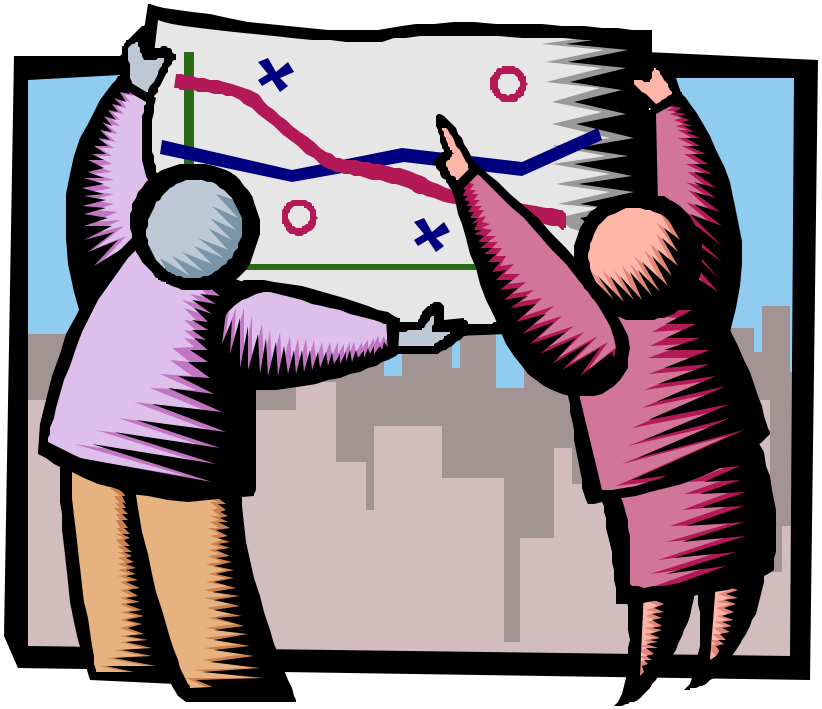


7. Measure Your Results - Modify Processes as Necessary

8. Confirm Your Status With Independent Appraisals

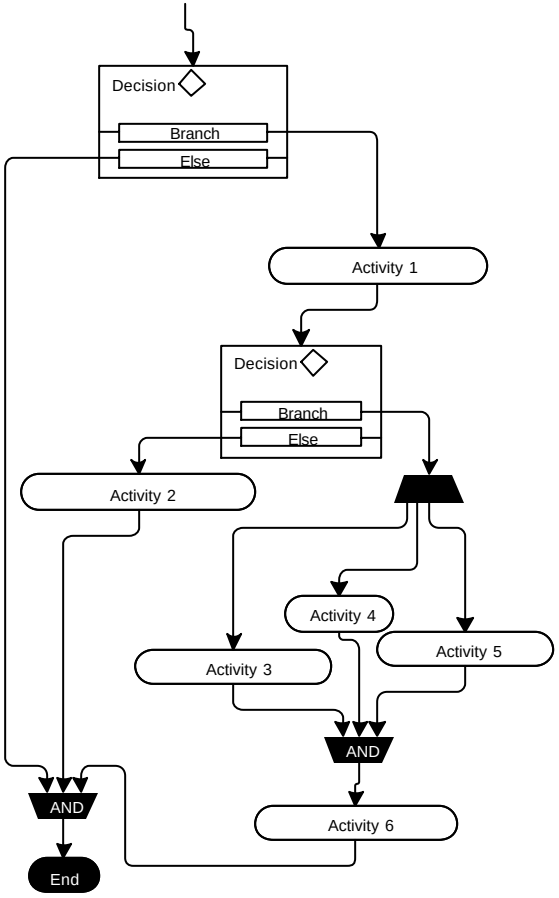


Step 1 – Understand your business processes



Your Business is Your Business – It's Not CMMI[SM] Implementation

- You must fully understand your business processes before you can address process completeness or process compliance.
  - What are your business processes?
  - Are they well-documented?
  - Are roles and responsibilities well-defined?
  - Are lines of authority well-defined?
  - Are internal and external interfaces well-defined?
  - Do your business processes satisfy your business goals?



Step 2 - Look to the CMMI[SM] for Process Completeness

The CMMI[SM] Is A Process Framework Model

- Contains the essential elements of effective processes for one or more disciplines
- Contains a framework that provides the ability to generate multiple models and associated training and assessment materials. These models may represent:
  - software and systems engineering
  - integrated product and process development
  - new disciplines
  - combinations of disciplines
- Provides guidance to use when developing processes

What The CMMI[SM] Is Not

- CMMI models are not processes or process descriptions. Actual processes depend on:
  - Application domain(s)
  - Organization structure
  - Organization size
  - Organization culture
  - Customer requirements or constraints

The CMMI[SM] as a Guide to Process Completeness

Determine if essential elements of your processes are missing or incomplete

Process Management

- **Organizational Process Focus**
- **Organizational Process Definition**
- **Organizational Training**
- **Organizational Process Performance**
- **Organizational Innovation and Deployment**

Engineering

- **Requirements Management**
- **Requirements Development**
- **Technical Solution**
- **Product Integration**
- **Verification**
- **Validation**

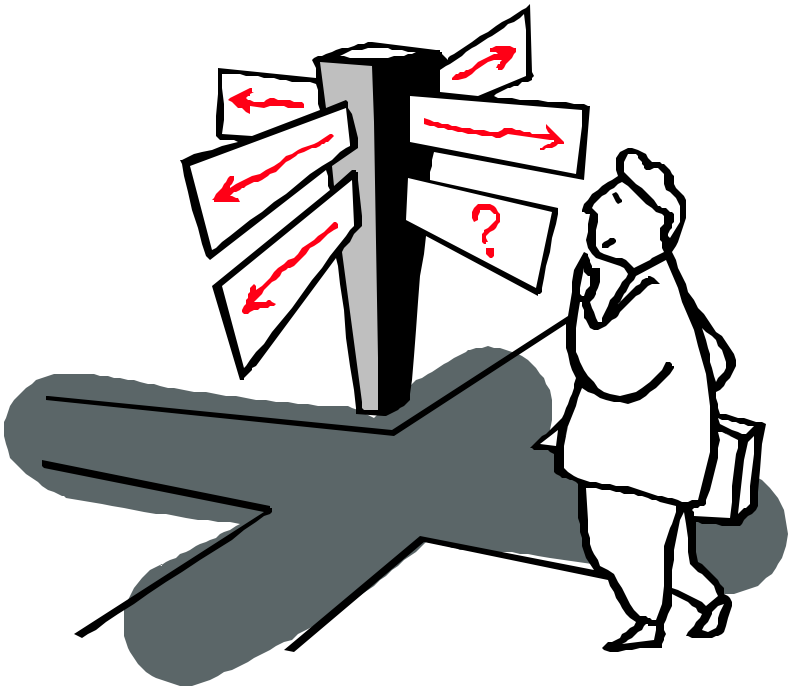
Project Management

- **Project Planning**
- **Project Monitoring and Control**
- **Supplier Agreement Management**
- **Integrated Project Management for IPPD**
- **Risk Management**
- **Integrated Teaming**
- **Integrated Supplier Management**
- **Quantitative Project Management**

Support

- **Configuration Management**
- **Process and Product Quality Assurance**
- **Measurement and Analysis**
- **Decision Analysis and Resolution**
- **Organizational Environment for Integration**
- **Causal Analysis and Resolution**

Step 3 - Look to Framework Standards for Life Cycle Definition



Standards As Sources of Best Practice Information

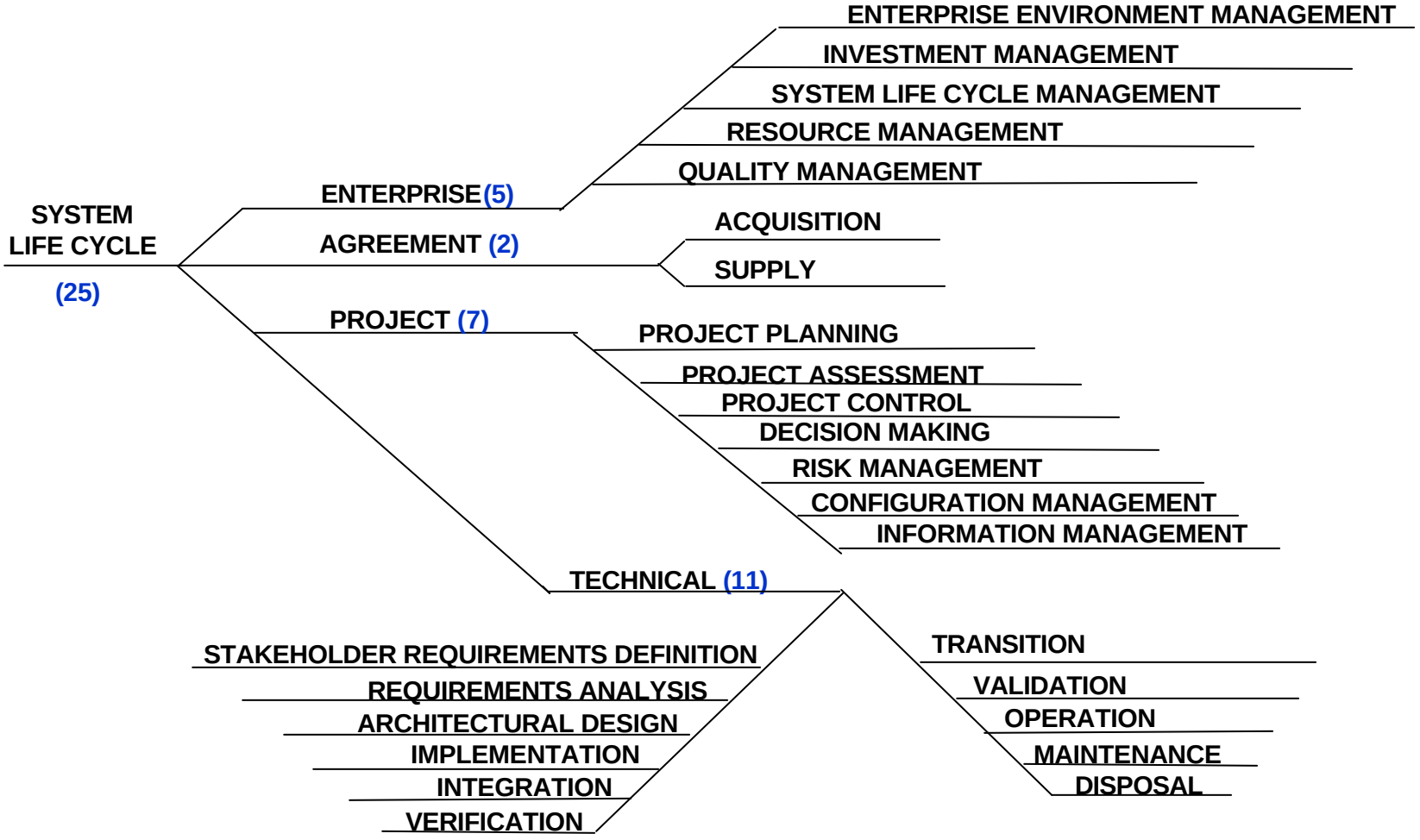
Standards, are consensus-based documents that codify best practice. Consensus-based standards have seven essential attributes that aid in process engineering. They:

- represent the collected experience of others who have been down the same road,
- tell in detail what it means to perform a certain activity,
- can be attached to or referenced by contracts,
- help to assure that two parties have the same meaning for an engineering activity,
- increase professional discipline,
- protect the business and the buyer,
- improve the product.

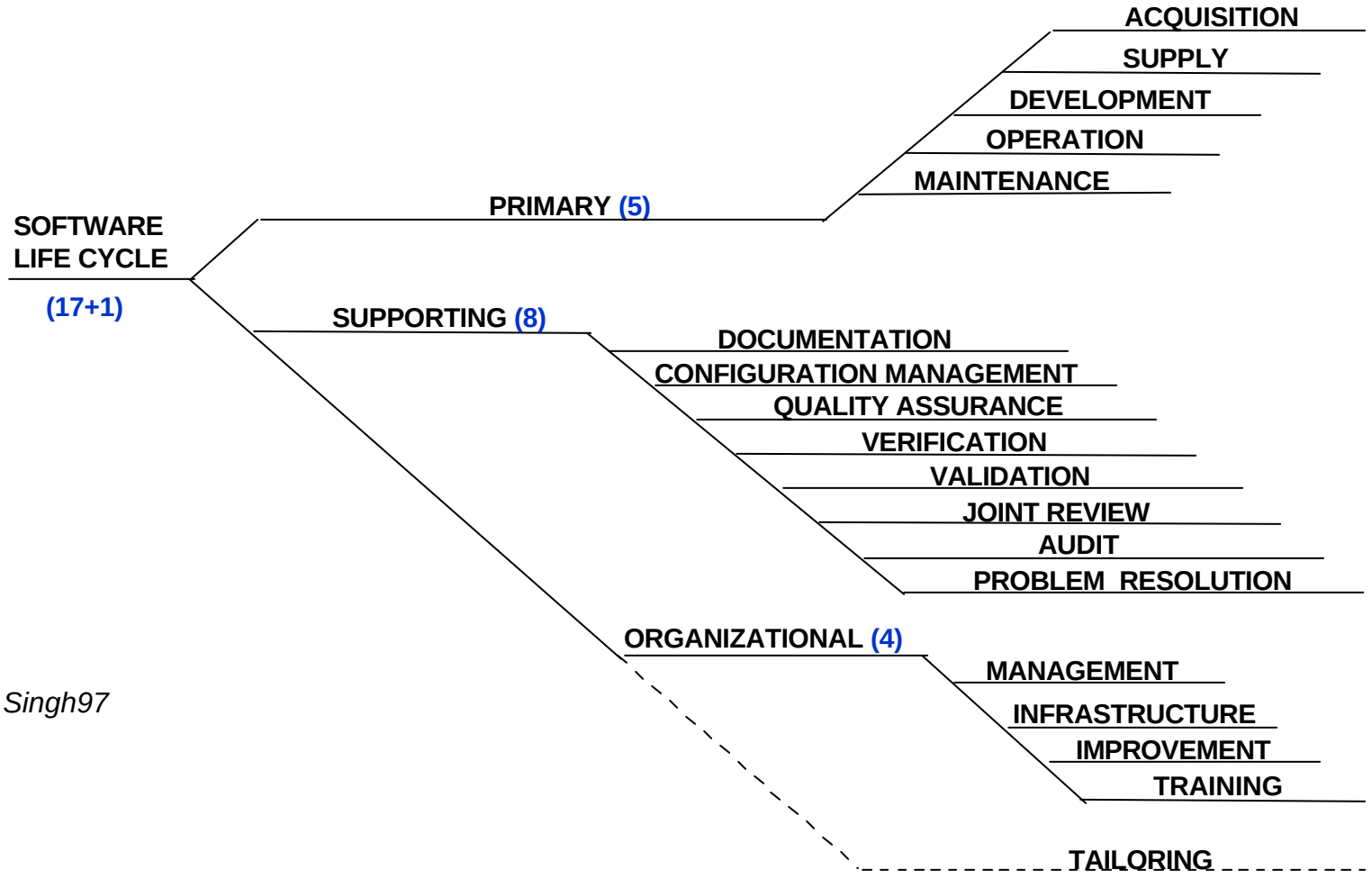
Process Framework Standards

- Systems Life Cycle
  - ISO/IEC 15288
- Software Life Cycle
  - ISO/IEC 12207
  - IEEE/EIA 12207.0, 12207.1, 12207.2
- Process Assessment
  - ISO/IEC 15504

The ISO/IEC 15288 Systems Life Cycle Process Framework

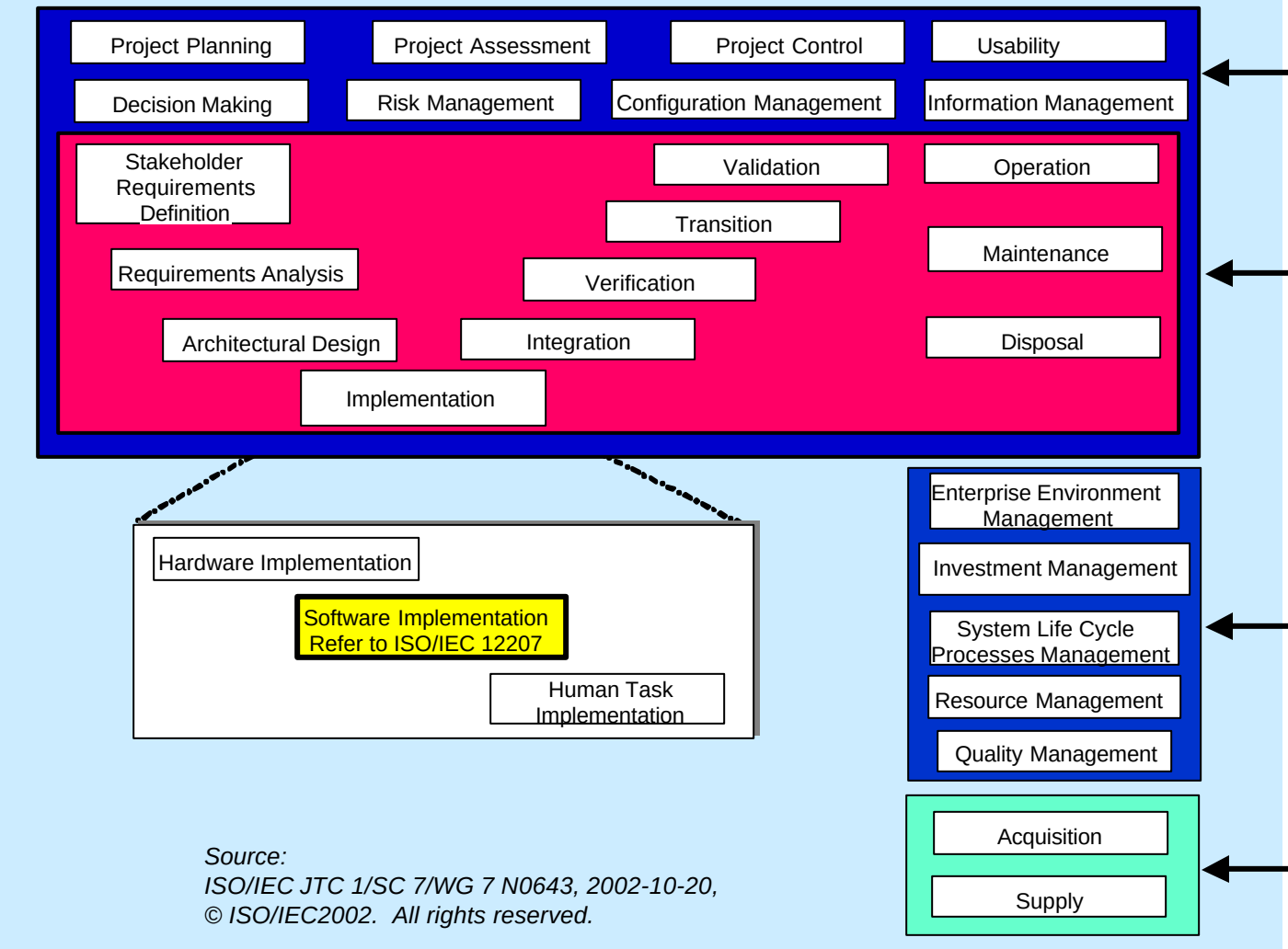


The IEEE/EIA 12207 Software Life Cycle Process Framework



Source: Singh97

Relationship between ISO/IEC 15288 and ISO/IEC 12207



Source:
ISO/IEC JTC 1/SC 7/WG 7 N0643, 2002-10-20,

But Wait . . . The Life Cycle Standards Are Not Consistent!!

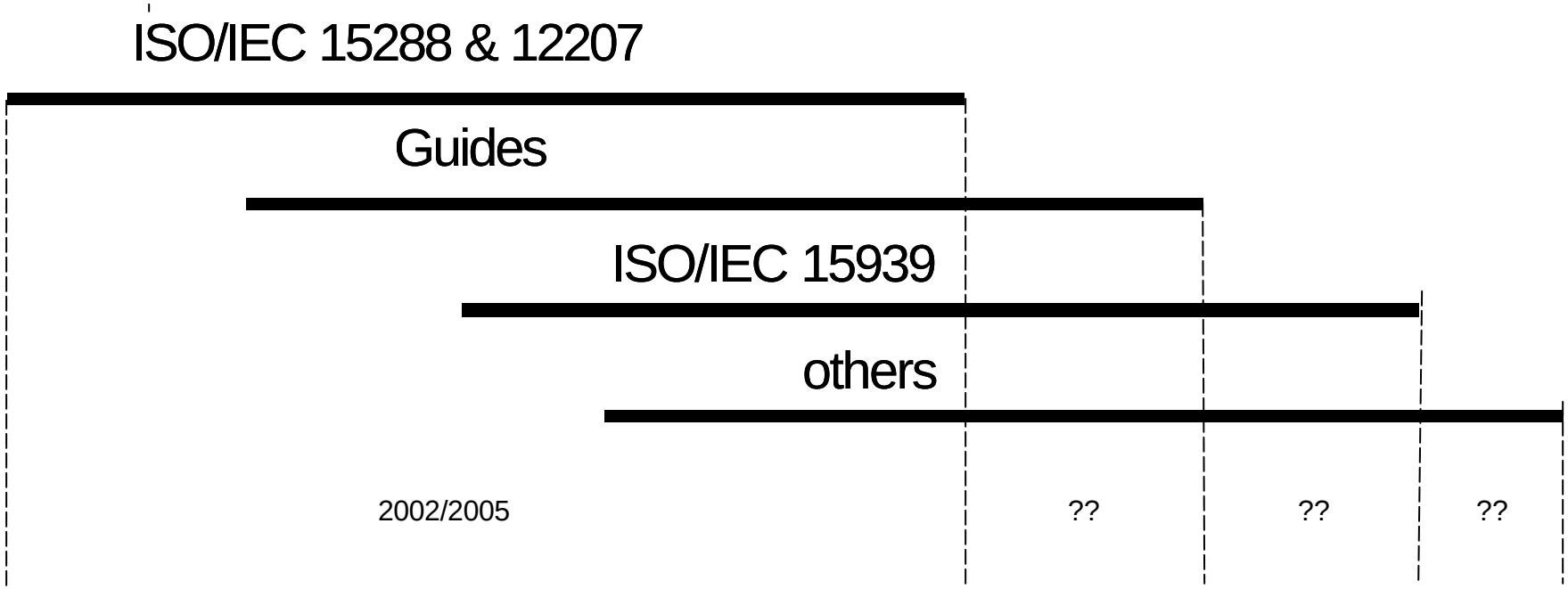
Harmonization – Consistency Goals

- Concepts
- Terminology
- Readability
- Level of detail
- Processes
- Document structure
- Normative references
- Common interfacing mechanism with the ISO 9000 family of standards
- Conformance with requirements from ISO/IEC 15504 as applicable

Harmonization – Priorities

- Primary
  - ISO/IEC 15288
  - ISO/IEC 12207
- Secondary
  - ISO/IEC 15939 - Software measurement process
  - ISO/IEC TR 15271 Guide for 12207
  - ISO/IEC TR 19760 Guide for 15288
  - ISO/IEC TR 15504 - parts 5 and 6
- Possible Additional
  - ISO/IEC 14764 Software maintenance
  - ISO/IEC TR 16326 Project management
  - ISO/IEC 18019 Documentation process
  - ISO/IEC TR 15846 Configuration management

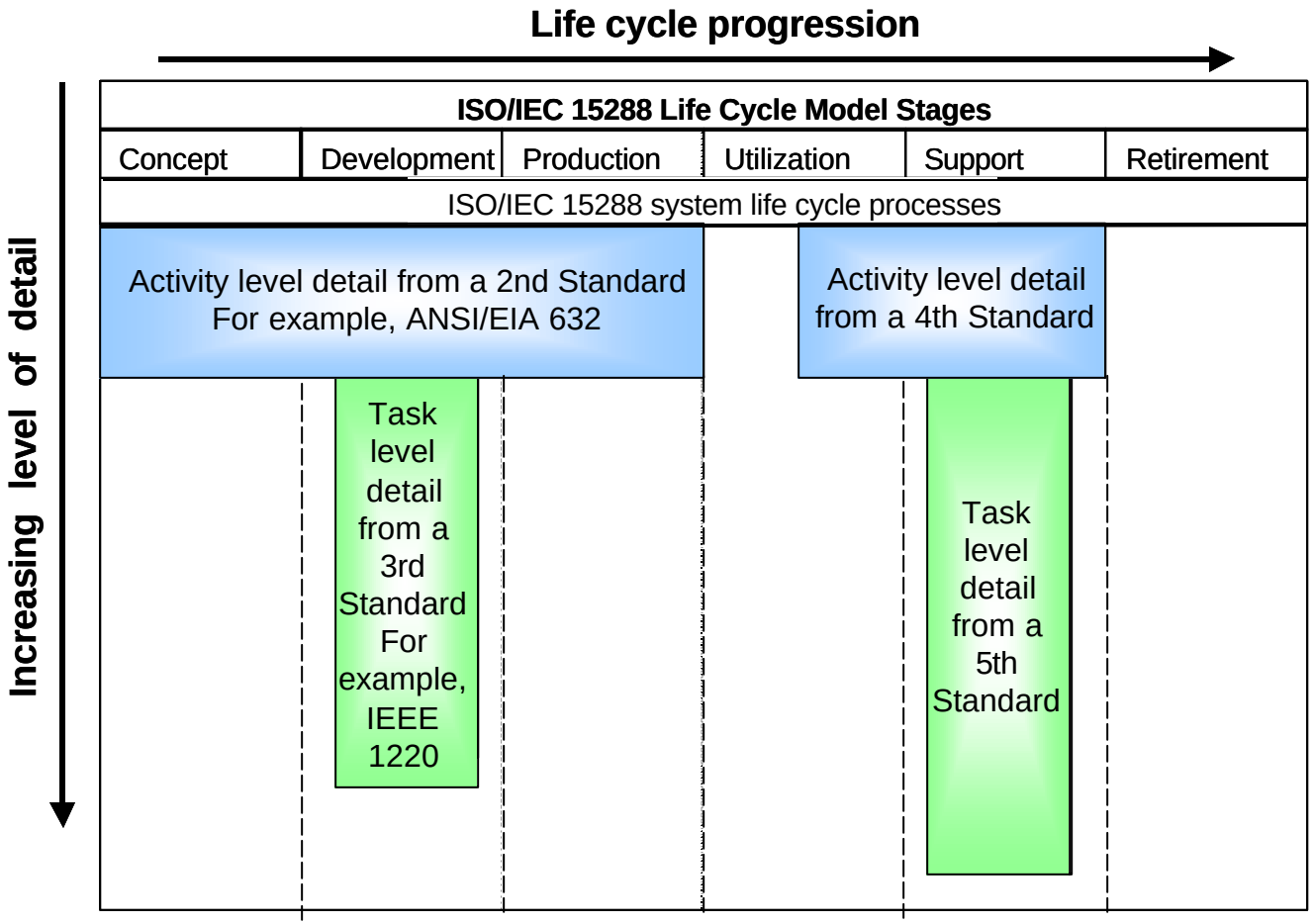
Life Cycle Standards Harmonization Phasing



Source:
ISO/IEC JTC 1/SC 7/WG 7 N0618, 2002-7-21 , © ISO/IEC2002. All rights reserved.

Step 4 – Look to Supporting Standards For Process Detail

Best Practice Support for the System Life Cycle Framework



A1. ISO/IEC 15288 and other engineering standards

Source: Guide for ISO/IEC 15288 (System Life Cycle Processes), PDTR, © ISO/IEC2002. All rights reserved.

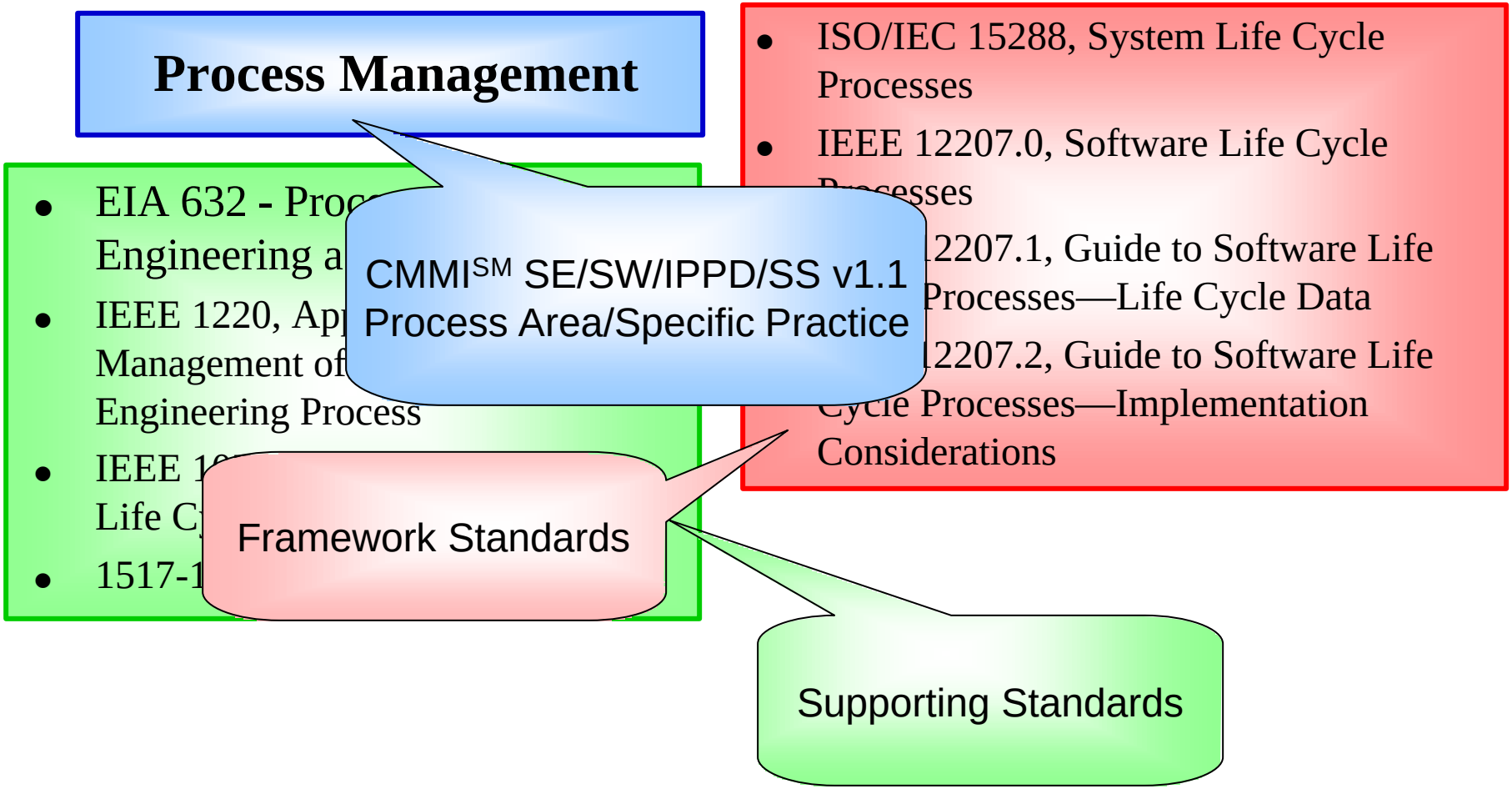
Best Practice Support for the Software Life Cycle Framework

Table 1—Information item matrix

Information item(s)	IEEE/EIA 12207.0 Clause	Kind of documentation	IEEE/EIA 12207.1 Clause	References (See annex A.)
Acceptance strategy and conditions record	5.1.1.9	Record (5.4)	—	IEEE 1062
Acquisition plan	5.1.1.8	Plan	6.1	ASTM E731, E1206, IEEE 1062
Acquisition requirements record	5.1.2.1	Record (5.4)	—	IEEE 1062, 1220
Audit agenda record	6.7.1.4	Record (5.4)	—	—
Audit procedure	6.7.1.4	Procedure (5.3)	—	—
Change request	5.4.4, 5.5.1, 6.2.3	Request	6.2	—
Concept of operations description	5.1.1.1	Description	6.3	IEEE 1362, EIA/IEEE J-STD-016 F.2.1. Also see ISO 5806, 5807, 8631, 8790, and 11411 for guidance on use of notations.
Concept/need determination record	5.1.1.1	Record (5.4)	—	IEEE 1062, 1220
Database design description	5.3.5.3, 5.3.6.3, 5.3.7.1	Description	6.4	IEEE 1016, EIA/IEEE J-STD-016 G.2.3
Detailed design evaluation record	5.3.6.7	Record	6.6	—
Development process plan	5.3.1.4	Plan	6.5	ASTM E622, E1340, EIA/IEEE J-STD-016 E.2.1, IEEE 1074, 1074.1

Source: IEEE/EIA 12207.1-1997,

CMMI[SM] SE/SW/IPPD/SS v1.1 Mapping - Process Management



CMMI[SM] SE/SW/IPPD/SS v1.1 Mapping – Project Management

Project Management

- IEEE 1220, Application and Management of the Systems Engineering Process
- IEEE 1058, Software Project Management Plans
- IEEE 1490, A Guide to the Program Management Body of Knowledge
- IEEE 1062, Recommended Practice for Software Acquisition
- IEEE 1540, Risk Management
- IEEE 1028, Software Reviews

- ISO/IEC 15288, System Life Cycle Processes
- IEEE 12207.0, Software Life Cycle Processes
- IEEE 12207.1, Guide to Software Life Cycle Processes—Life Cycle Data
- IEEE 12207.2, Guide to Software Life Cycle Processes—Implementation Considerations

CMMI[SM] SE/SW/IPPD/SS v1.1 Mapping – Engineering

Engineering

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 1362, Guide for Concept of Operations Document
- IEEE 1471, Architectural Description of Software Intensive Systems
- IEEE 830, Software Requirements Specifications
- IEEE 1016, Software Design Descriptions
- IEEE 1012, Software Verification and Validation
- IEEE 1008, Software Unit Testing

- ISO/IEC 15288, System Life Cycle Processes
- IEEE 12207.0, Software Life Cycle Processes
- IEEE 12207.1, Guide to Software Life Cycle Processes—Life Cycle Data
- IEEE 12207.2, Guide to Software Life Cycle Processes—Implementation Considerations

- IEEE 1228, Software Safety Plans
- IEEE 1063, Software User Documentation
- IEEE 1219, Software Maintenance
- IEEE 1320.1,.2, IDEF0, IDEF1X97
- IEEE 1420.1, Data Model for Reuse Library Interoperability

CMMI[SM] SE/SW/IPPD/SS v1.1 Mapping – Support

Support

- IEEE 828, Software Configuration Management Plans
- IEEE 730, Software Quality Assurance Plans
- IEEE 982.1, Dictionary of Measures to Produce Reliable Software
- IEEE 1045, Software Productivity Metrics
- IEEE 1061, Software Quality Metrics Methodology
- IEEE 1219, Software Maintenance

- ISO/IEC 15288, System Life Cycle Processes
- IEEE 12207.0, Software Life Cycle Processes
- IEEE 12207.1, Guide to Software Life Cycle Processes—Life Cycle Data
- IEEE 12207.2, Guide to Software Life Cycle Processes—Implementation Considerations

- IEEE 1465 (ISO/IEC 12119) - Software Packages - Quality Requirements and Testing
- IEEE 14143.1 (ISO/IEC 1443-1) - Functional Size Measurement - Part 1: Definition of Concepts

Requirements Development

SP 2.1-1 Establish Product and Product Component Requirements

- Establish and maintain, from the customer requirements, product and product component requirements essential to product and product component effectiveness and affordability

- ISO/IEC 15288, System Life Cycle Processes
  - Clause 5.5.3 - Requirements Analysis Process
- IEEE/EIA 12207.0, Software Life Cycle Processes
  - Clause 5.3.2 - System Requirements Analysis
  - Clause 5.3.4 - Software requirements analysis

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 830, Software Requirements Specifications

Requirements Development

SP 2.1-1 Establish Product and Product Component Requirements

- Establish and maintain, from th

5.5.3 Requirements Analysis Process
5.5.3.1 Purpose of the Requirements Analysis Process
The purpose of the Requirements Analysis Process is to transform the stakeholder, requirement-driven view of desired services into a technical view of a required product that could deliver those services. This process builds a representation of a future system that will meet stakeholder requirements and that, as far as constraints permit, does not imply any specific implementation. It results in measurable system requirements that specify, from the developer's perspective, what characteristics it is to possess and with what magnitude in order to satisfy stakeholder requirements.
5.5.3.2 Requirements Analysis Process Outcomes
As a result of the successful implementation of the Requirements Analysis Process:
a) The required characteristics, attributes, and functional and performance requirements for a product solution are specified.
b) Constraints that will affect the architectural design of a system and the means to realize it are specified.
c) The integrity and traceability of system requirements to stakeholder requirements is achieved.. . .

- ISO/IEC 15288, System Life Cycle Processes
  - Clause 5.5.3 - Requirements Analysis
    rocess
  - A 12207.0, Software Life Cycle
    s
  - se 5.3.2 - System Requirements
    ysis
  - se 5.3.4 - Software requirements
    sis

- 33, Guide for Developing System
  ents Specifications
- 0, Software Requirements
  tions

Source:
ISO/IEC CD 15288 FDIS, © ISO/IEC2002.
All rights reserved.

evelopment

5.3.2.1 The specific intended use of the system to be developed shall be analyzed to specify system requirements. The system requirements specification shall describe: functions and capabilities of the system; business, organizational and user requirements; safety, security, human-factors engineering (ergonomics), interface, operations, and maintenance requirements; design constraints and qualification requirements. The system requirements specification shall be documented.
5.3.4.1 The developer shall establish and document software requirements, including the quality characteristics specifications, described below. . . .
a) Functional and capability specifications, including performance, physical characteristics, and environmental conditions under which the software item is to perform;
b) Interfaces external to the software item;
c) Qualification requirements;
d) Safety specifications, including those related to methods of operation and maintenance, environmental influences, and personnel injury;
e) Security specifications, including those related to compromise of sensitive information . . .

SO/IEC 15288, System Life Cycle rocesses

- Clause 5.5.3 - Requirements Analysis Process

EE/EIA 12207.0, Software Life Cycle ocesses

- Clause 5.3.2 - System Requirements Analysis
- Clause 5.3.4 - Software requirements analysis

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 830, Software Requirements Specifications

Source:
IEEE/EIA 12207.0-1997,
© IEEE 2001. All rights reserved.

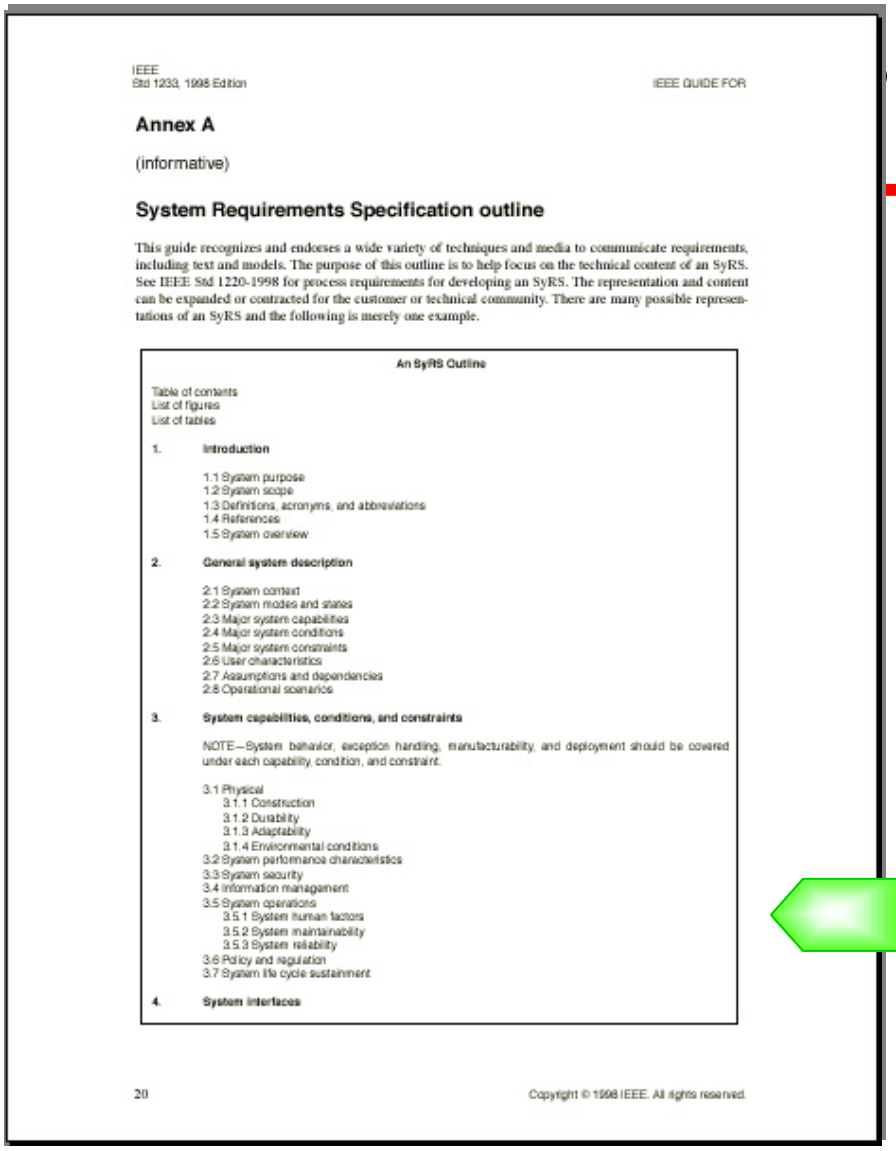
velopment

7.2 Build a well-formed requirement
The analysts carry out this subphase by doing the following:
a) Ensuring that each requirement is a necessary, short, definitive statement of need (capability, constraints);
b) Defining the appropriate conditions (quantitative or qualitative measures) for each requirement and avoiding adjectives such as "resistant" or "industry wide;"
c) Avoiding requirements pitfalls (see 6.4);
d) Ensuring the readability of requirements, which entails the following:
1) Simple words/phrases/concepts;
2) Uniform arrangement and relationship;
3) Definition of unique words, symbols, and notations;
4) The use of grammatically correct language and symbology.
e) Ensuring testability.
Example:
Capability: Move people between Los Angeles and New York
Condition: Cruising speed of 200 km/hr
Constraint: Maximum speed of 300 km/hr
Well-formed requirement: This system should move people between Los Angeles and New York at an optimal cruising speed of 200 km/hr with a maximum speed of 300 km/hr.

- EC 15288, System Life Cycle sses
  - lause 5.5.3 - Requirements Analysis rocess
- /EIA 12207.0, Software Life Cycle sses
  - lause 5.3.2 - System Requirements nalysis
  - lause 5.3.4 - Software requirements nalysis

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 830, Software Requirements Specifications

Source:
IEEE 1233-1998, © IEEE 1998. All rights reserved.



IEEE
Std 1233, 1998 Edition IEEE GUIDE FOR

Annex A

(informative)

System Requirements Specification outline

This guide recognizes and endorses a wide variety of techniques and media to communicate requirements, including text and models. The purpose of this outline is to help focus on the technical content of an SyRS. See IEEE Std 1220-1998 for process requirements for developing an SyRS. The representation and content can be expanded or contracted for the customer or technical community. There are many possible representations of an SyRS and the following is merely one example.

An SyRS Outline

Table of contents
List of figures
List of tables

1. **Introduction**
   - 1.1 System purpose
   - 1.2 System scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 System overview
2. **General system description**
   - 2.1 System context
   - 2.2 System modes and states
   - 2.3 Major system capabilities
   - 2.4 Major system conditions
   - 2.5 Major system constraints
   - 2.6 User characteristics
   - 2.7 Assumptions and dependencies
   - 2.8 Operational scenarios
3. **System capabilities, conditions, and constraints**

   NOTE—System behavior, exception handling, manufacturability, and deployment should be covered under each capability, condition, and constraint.

   - 3.1 Physical
     - 3.1.1 Construction
     - 3.1.2 Durability
     - 3.1.3 Adaptability
     - 3.1.4 Environmental conditions
   - 3.2 System performance characteristics
   - 3.3 System security
   - 3.4 Information management
   - 3.5 System operations
     - 3.5.1 System human factors
     - 3.5.2 System maintainability
     - 3.5.3 System reliability
   - 3.6 Policy and regulation
   - 3.7 System life cycle sustainment
4. **System interfaces**

20 Copyright © 1998 IEEE. All rights reserved.

Development

- ISO/IEC 15288, System Life Cycle Processes
  - Clause 5.5.3 - Requirements Analysis Process
- IEEE/EIA 12207.0, Software Life Cycle Processes
  - Clause 5.3.2 - System Requirements Analysis
  - Clause 5.3.4 - Software requirements analysis

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 830, Software Requirements Specifications

Source:
IEEE 1233-1998, © IEEE 1998. All rights reserved.

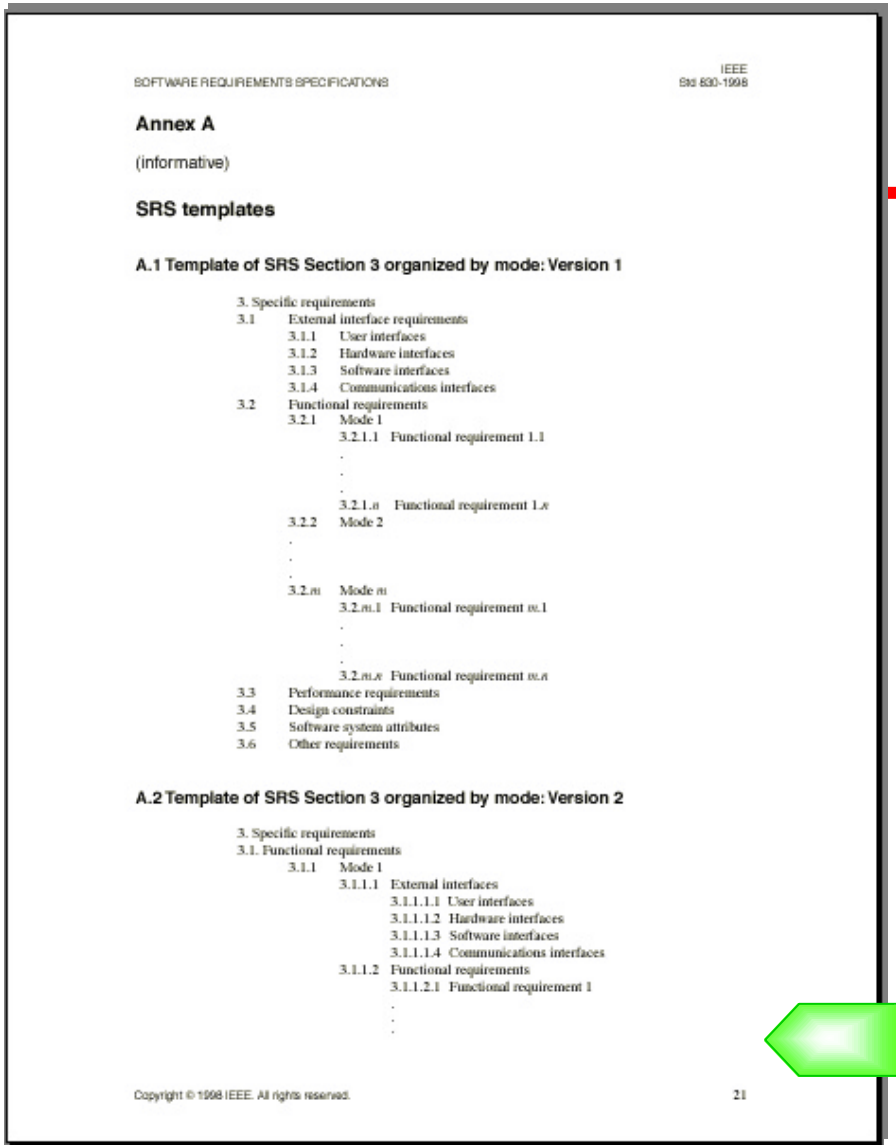
elopment

5.3.2 Functions
Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as “shall” statements starting with “The system shall”
These include:
a) Validity checks on the inputs
b) Exact sequence of operations
c) Responses to abnormal situations, including:
1) Overflow
2) Communication facilities
3) Error handling and recovery
d) Effect of parameters
e) Relationship of outputs to inputs . . .
1) It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does
not imply that the software design will also be partitioned that way.
5.3.3 Performance requirements
This subsection should specify both the static and the dynamic numerical requirements placed on the soft-
ware or on human interaction with the software as a whole. Static numerical requirements may include the
following:
a) The number of terminals to be supported;
b) The number of simultaneous users to be supported;
c) Amount and type of information to be handled.

Source:
IEEE 830-1998, © IEEE 1998. All rights reserved.

EC 15288, System Life Cycle
sses
lause 5.5.3 - Requirements Analysis
rocess
/EIA 12207.0, Software Life Cycle
sses
lause 5.3.2 - System Requirements
nalysis
lause 5.3.4 - Software requirements
nalysis

E 1233, Guide for Developing System
equirements Specifications
IEEE 830, Software Requirements Specifications



SOFTWARE REQUIREMENTS SPECIFICATIONS

IEEE Std 830-1998

Annex A

(informative)

SRS templates

A.1 Template of SRS Section 3 organized by mode: Version 1

3. Specific requirements
3.1 External interface requirements
3.1.1 User interfaces
3.1.2 Hardware interfaces
3.1.3 Software interfaces
3.1.4 Communications interfaces
3.2 Functional requirements
3.2.1 Mode 1
3.2.1.1 Functional requirement 1.1
.
.
.
3.2.1.*n* Functional requirement 1.*n*
3.2.2 Mode 2
.
.
.
3.2.*m* Mode *m*
3.2.*m*.1 Functional requirement *m*.1
.
.
.
3.2.*m*.*n* Functional requirement *m*.*n*
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.2 Template of SRS Section 3 organized by mode: Version 2

3. Specific requirements
3.1. Functional requirements
3.1.1 Mode 1
3.1.1.1 External interfaces
3.1.1.1.1 User interfaces
3.1.1.1.2 Hardware interfaces
3.1.1.1.3 Software interfaces
3.1.1.1.4 Communications interfaces
3.1.1.2 Functional requirements
3.1.1.2.1 Functional requirement 1
.
.
.

Development

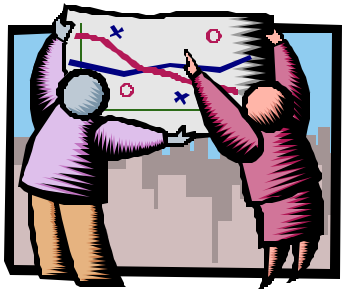
- ISO/IEC 15288, System Life Cycle Processes
  - Clause 5.5.3 - Requirements Analysis Process
- IEEE/EIA 12207.0, Software Life Cycle Processes
  - Clause 5.3.2 - System Requirements Analysis
  - Clause 5.3.4 - Software requirements analysis

- IEEE 1233, Guide for Developing System Requirements Specifications
- IEEE 830, Software Requirements Specifications

Source:
IEEE 830-1998, © IEEE 1998. All rights reserved.

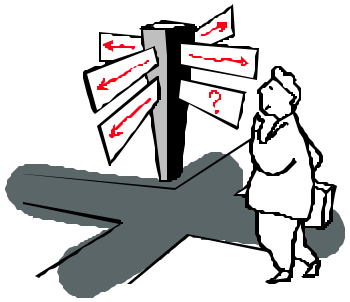
8 Steps to Success With Best Practice Information Aids

1. Understand your business processes



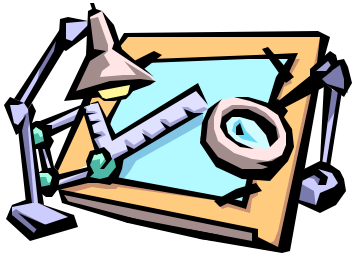
2. Look to the CMMI[SM] for Process Completeness

3. Look to Framework Standards for Life Cycle Definition

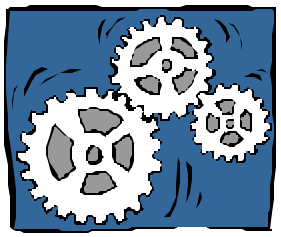


4. Look to Supporting Standards for Process Detail

5. Build or Refine Your Process Architecture

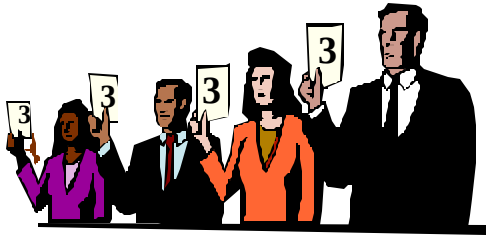


6. Execute Your Processes

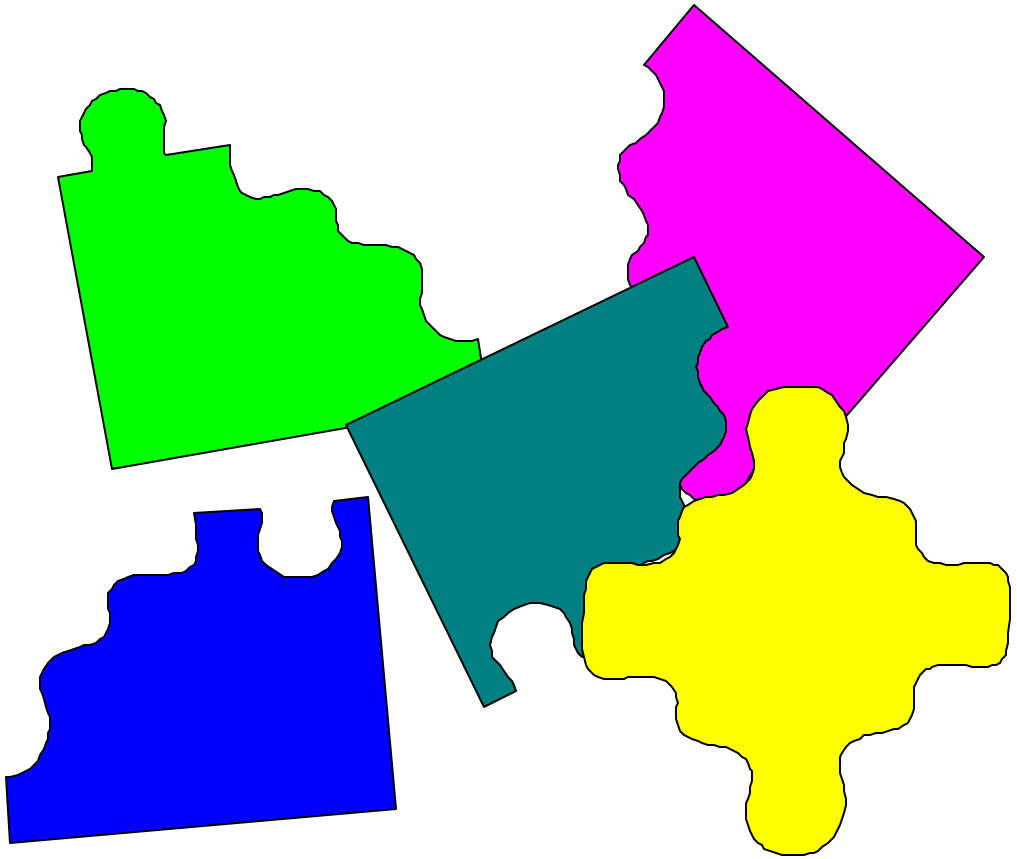


7. Measure Your Results - Modify Processes as Necessary

8. Confirm Your Status With Independent Appraisals



The IEEE CMMI[SM] Mapping Project



The IEEE CMMI[SM] Mapping Project – Background

- Study Group on External Standards Coordination formed
  - Initial Task: Map the IEEE Software Engineering Standards Collection to the CMMI[SM].
- Software Productivity Consortium's Quagmap® selected as the mapping tool.
- License agreements for distribution of Standards and the Quagmap® tool to Study Group members negotiated and signed.
- Results to be published on the SESC web site.

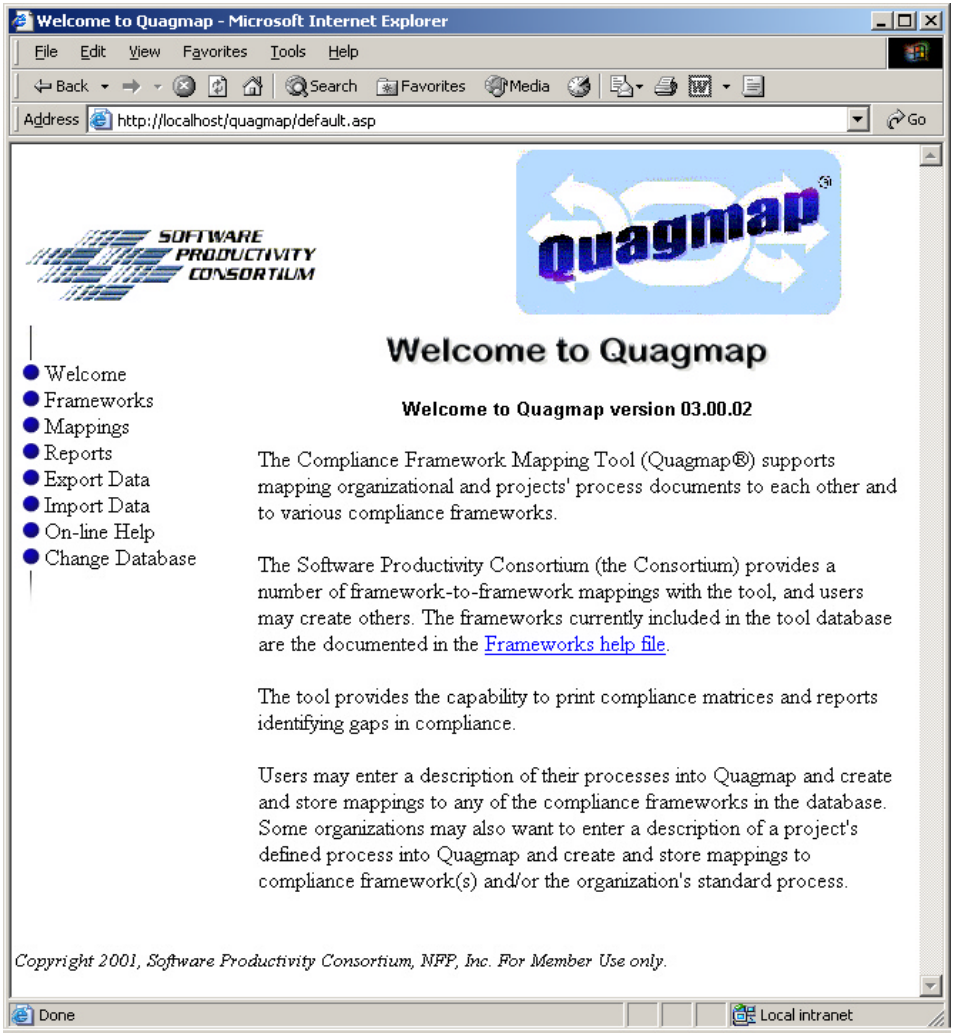
Best Practice Information Aids for CMMI[SM]-Compliant Process Engineering

610.12-1990	IEEE Standard Glossary of Software Engineering Terminology
730-1998	IEEE Standard for Software Quality Assurance Plans
828-1998	IEEE Standard for Software Configuration Management Plans
829-1998	IEEE Standard for Software Test Documentation
830-1998	IEEE Recommended Practice for Software Requirements Specifications
982.1-1988	IEEE Standard Dictionary of Measures to Produce Reliable Software
1008-1987	IEEE Standard for Software Unit Testing
1012-1998	IEEE Standard for Software Verification and Validation
1012a-1998	IEEE Standard for Software Verification and Validation - Supplement to 1012-1998 - Content Map to IEEE 12207.1
1016-1998	IEEE Recommended Practice for Software Design Descriptions
1028-1997	IEEE Standard for Software Reviews
1044-1993	IEEE Standard Classification for Software Anomalies
1045-1992	IEEE Standard for Software Productivity Metrics
1058-1998	IEEE Standard for Software Project Management Plans
1058.1-1987	IEEE Standard for Software Project Management Plans
1061-1998	IEEE Standard for Software Quality Metrics Methodology
1062, 1998	IEEE Recommended Practice for Software Acquisition (includes IEEE 1062a)
1063-2001	IEEE Standard for Software User Documentation
1074-1997	IEEE Standard for Developing Software Life Cycle Processes
1219-1998	IEEE Standard for Software Maintenance
1220-1998	IEEE Standard for the Application and Management of the Systems Engineering Process
1228-1994	IEEE Standard for Software Safety Plans
1233, 1998	IEEE Guide for Developing System Requirements Specifications (including IEEE 1233a)
1320.1-1998	IEEE Standard for Functional Modeling Language - Syntax and Semantics for IDEF0
1320.2-1998	IEEE Standard for Conceptual Modeling Language Syntax and Semantics for IDEF1X97 (IDEFobject)
1362-1998	IEEE Guide for Information Technology-System Definition-Concept of Operation Document
1420.1-1995	IEEE Standard for Information Technology--Software Reuse--Data Model for Reuse Library Interoperability: Basic Interoperability Data Model (BIDM)
1420.1a-1996	IEEE Supplement to Standard for Information Technology--Software Reuse--Data Model for Reuse Library Interoperability: Asset Certification Framework
1420.1b-1999	IEEE Trial-use Supplement to IEEE Standard for Information Technology--Software Reuse--Data Model for Reuse Library Interoperability: Intellectual Property Rights Framework
1465-1998	(12119:1998 ISO/IEC) Information Technology - Software Packages - Quality Requirements and Testing
1471-2000	IEEE Recommended Practice for Architectural Description of Software Incentive Systems
1490-1998	IEEE Guide (©IEEE) - Adoption of PMI Standard - A Guide to the Project Management Body of Knowledge(©PMI)
1517-1999	IEEE Standard for Information Technology - Software Life Cycle Processes - Reuse Processes
1540-2001	IEEE Standard for Software Life Cycle Processes-Risk Management
12207.0-1996	IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology--Software Life Cycle Processes
12207.1-1997	IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology- - Software Life Cycle Processes --Life cycle data
12207.2-1997	IEEE/EIA Standard: Industry Implementation of International Standard ISO/IEC 12207:1995 Standard for Information Technology- - Software Life Cycle Processes --Implementation considerations
14143.1-2000	Implementation Note for IEEE Adoption of ISO/IEC 14143-1:1998, Information Technology-Software Measurement-Functional Size Measurement-Part 1: Definition of Concepts

[SM] CMMI is a service mark of Carnegie Mellon University

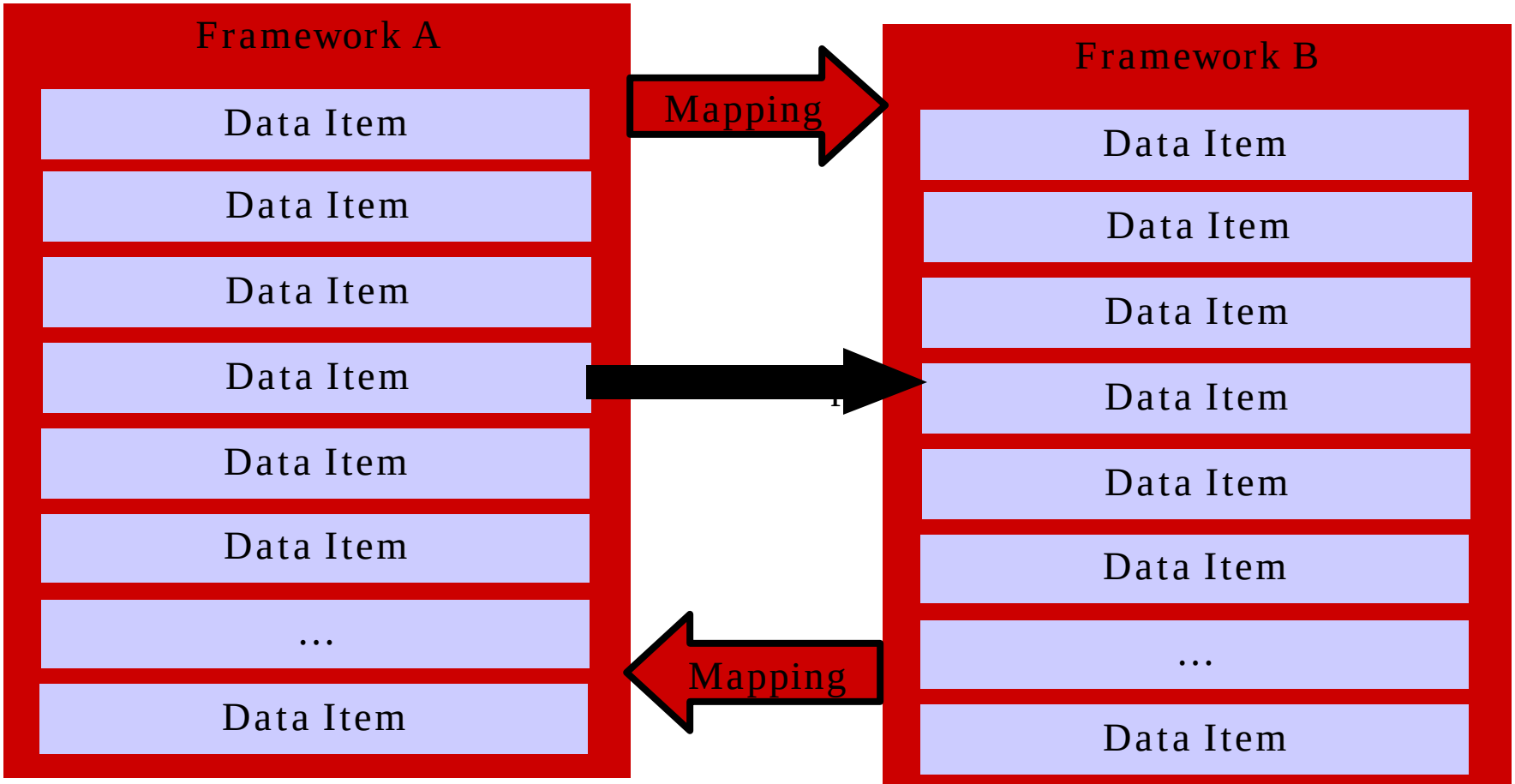
For more information:
http://standards.computer.org/sesc/

The SPC Quagmap® Tool



Source: Software Productivity Consortium www.software.org

Quagmap® Information Structure



Source: Software Productivity Consortium, www.software.org

Quagmap® Mappings and Relationships

- Mappings define how well one framework (the Source) meets the intent of the second (the Target)
- Relationships define how well each source framework data item meets the intent of the associated target framework item[s]
  - [none]
  - Partial
  - Complete
- These are M:M mappings and relationships
- Standards mappings are generally done in both directions

Source: Software Productivity Consortium, www.software.org

Quagmap® Reports

- Framework: Description and all data items
- Relationship: Source framework items and related data items from one or more target frameworks
- Coverage: Target framework items and the source data items that meet them
- Gap Identification: Target items not fully satisfied by the source data items
- Inferred Mapping: Given a mapping from A:B, and B:C; what is a likely set of relationships from A:C?

Source: Software Productivity Consortium, www.software.org

Quagmap® Relationship Report

Relationship Report - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Relationship Report

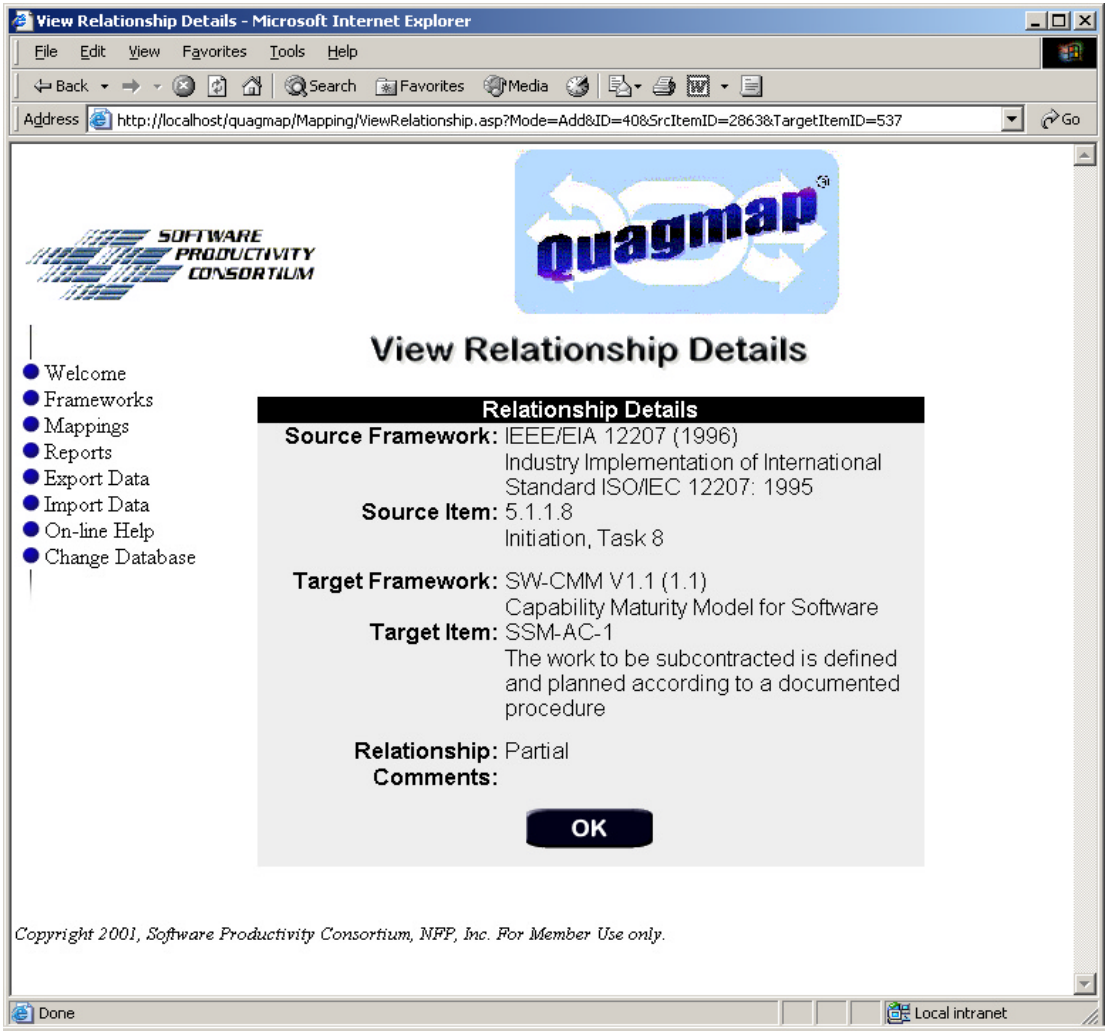
Source: IEEE/EIA 12207 (1996)
Notes:

Printed: 11/12/2002
Format to Save or Print

Source Item Name	Source Item Description	Target Framework	Target Item Name	Target Item Description	Relationship	Comment
				according to a documented procedure		
5.1.1.9	Initiation, Task 9	SW-CMM V1.1 (1.1)	SSM-AC-3	The contractual agreement between the prime contractor and the software subcontractor is used as the basis for managing the subcontract	Partial	
5.1.1.9	Initiation, Task 9	SW-CMM V1.1 (1.1)	SSM-AC-12	The prime contractor conducts acceptance testing as part of the delivery of the subcontractor's software products according to a documented procedure	Partial	
5.1.2.1	Request-for-proposal [-tender] preparation, Task 1	SW-CMM V1.1 (1.1)	SSM-AC-1	The work to be subcontracted is defined and planned according to a documented procedure	Partial	
5.1.2.3	Request-for-proposal [-tender] preparation, Task 3	SW-CMM V1.1 (1.1)	SSM-AC-3	The contractual agreement between the prime contractor and the software subcontractor is used as the basis for managing the subcontract	Partial	
5.1.2.4	Request-for-proposal [-	SW-CMM V1.1	SSM-AC-1	The work to be	Partial	

Source: Software Productivity Consortium www.software.org

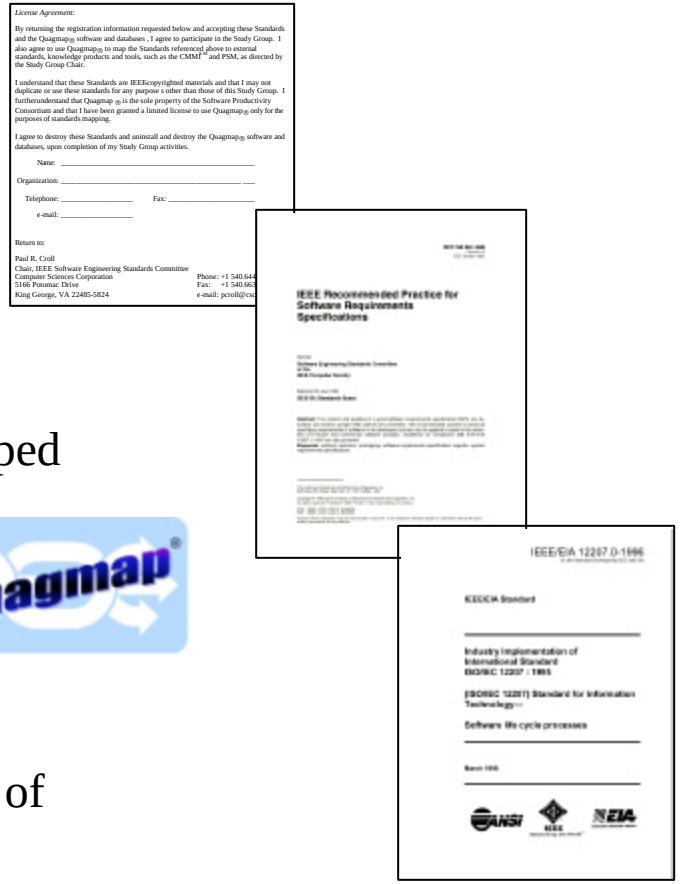
Quagmap® Relationship Details



Source: Software Productivity Consortium www.software.org

How You Can Help . . .

- Get involved in the mapping effort
  - Initial mapping
  - Peer reviewer
- What you will receive
  - A Copyright Notice and License Agreement
  - An electronic copy of the standard(s) to be mapped the associated framework standard (s)
  - A copy of the Quagmap® tool
  - A mapping guide
- How to get started
  - Contact me at: pcroll@csc.com to let me know of your interest



Questions?

For More Information . . .

Paul R. Croll
Computer Sciences Corporation
5166 Potomac Drive
King George, VA 22485-5824

Phone: +1 540.644.6224
Fax: +1 540.663.0276
e-mail: pcroll@csc.com

For IEEE Standards:

http://computer.org/standards/sesc/
http://computer.org/cspress/CATALOG/st01110.htm

For ISO/IEC Standards:

http://saturne.info.uqam.ca/Labo_Recherche/Lrgl/sc7/

References

CMMI[SM] -SE/SW/IPPD/SS, V1.1, *CMMI for Systems Engineering/Software Engineering/Integrated Product and Process Development, and Supplier Sourcing Version 1.1, CMMI[SM] -SE/SW/IPPD/SS, V1.1, Continuous Representation.* CMU/SEI-CMU/SEI-2002-TR-011, ESC-TR-2002-011, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, PA, March 2002.

Guide for ISO/IEC 15288 (System Life Cycle Processes), PDTR, ISO/IEC JTC1/SC7, 2002.

IEEE Standard 830-1998, *Recommended Practice for Software Requirements Specifications*, Institute of Electrical and Electronics Engineers, Inc. New York, NY, 1998.

References - 2

IEEE Standard 1233-1998, *Guide for Developing System Requirements Specifications*, Institute of Electrical and Electronics Engineers, Inc. New York, NY, 1998.

IEEE Standard 1540-2001, *IEEE Standard for Software Life Cycle Processes — Risk Management*, Institute of Electrical and Electronics Engineers, Inc. New York, NY, 2001.

IEEE/EIA Standard 12207.0-1996, *Industry Implementation of International Standard ISO/IEC12207:1995 — (ISO/IEC 12207) Standard for Information Technology —Software life cycle processes*, Institute of Electrical and Electronics Engineers, Inc. New York, NY, 1998.

References - 3

IEEE/EIA Standard 12207.1-1997, *Industry Implementation of International Standard ISO/IEC12207:1995 — (ISO/IEC 12207) Standard for Information Technology —Software life cycle processes - Implementation Considerations*, Institute of Electrical and Electronics Engineers, Inc. New York, NY, 1998.

ISO/IEC CD 15288 FDIS:2002, *Systems Engineering — System Life Cycle Processes*, ISO/IEC JTC1/SC7, 2002.

[Singh97] Raghu Singh, *An Introduction to International Standards ISO/IEC 12207, Software Life Cycle Processes*, 1997.