AN INTRODUCTION TO

INTERNATIONAL STANDARD

ISO/IEC 12207

SOFTWARE LIFE CYCLE PROCESSES

RAGHU SINGH
FAA, WASHINGTON, DC
April 26, 1999
ISO/IEC 12207
PURPOSE

• To establish a common framework for the life cycle of software
  - To foster mutual understanding among business parties
  - To acquire, supply, develop, operate, and maintain software
  - To manage, control, and improve the framework.

For World Trade in software:
“… facilitating international exchange of goods and services …”
ISO/IEC 12207
SCOPE

LIFE CYCLE: CRADLE ... GRAVE

STAKEHOLDERS: ACQUIRERS, SUPPLIERS, USERS, ...

APPLICATION: CORPORATE PROCESSES

PROJECT PRODUCTS ...

PROJECT SERVICES

DETAILS: PROCESS DEFINITIONS & DESCRIPTIONS

METHODOLOGIES, METHODS & METRICS

PROCEDURES, TECHNIQUES, TOOLS & ENVIRONMENTS

* NOT COVERED IN 12207, BUT SOME GUIDANCE IN ISO/IEC 15271
TOPICS

1. BACKGROUND
   - ISO and IEC
   - History of ISO/IEC 12207

2. BASIC CONCEPTS

3. THE PROCESSES

4. APPLICATION

5. RELATED AREAS

6. SUMMARY

7. FOR YOUR INFORMATION
BACKGROUND - I

ISO
INTERNATIONAL ORGANIZATION FOR STANDARDIZATION

- **ESTABLISHED:** 1947
- **OBJECT:** Promote the development of standardization ... in the world ... to facilitating international exchange of goods and services
- **MEMBERS:** 87 countries (1994)
- **TECHNICAL COMMITTEES (TCs):** Carry out technical work
- **TCs THAT MAY IMPACT SOFTWARE ENGINEERING:**
  - TC 10: Technical Drawings
  - TC 20: Space and aircraft vehicles
  - TC 46: Information and documentation
  - TC 145: Graphical symbols
  - TC 154: Documents and data elements in administration, commerce and industry
  - TC 159: Ergonomics
  - TC 176: Quality management and quality assurance
  - TC 184: Industrial automation systems
BACKGROUND - II

IEC

INTERNATIONAL ELECTROTECHNICAL COMMISSION

• **ESTABLISHED:** 1906

• **OBJECT:** Standardization in electrical and electronic engineering fields

• **TECHNICAL COMMITTEES (TCs):**
  - Carry out technical work

• **TCs THAT MAY IMPACT SOFTWARE ENGINEERING:**
  - TC 45: Nuclear instrumentation
  - TC 56: Dependability and maintainability
  - TC 65: Industrial process measurement/control
BACKGROUND - III

JOINT TECHNICAL COMMITTEE 1
INFORMATION TECHNOLOGY

ESTABLISHED: 1987
OBJECT: TO CARRY ON STANDARDIZATION WORK IN INFORMATION TECHNOLOGY

SC1 - Vocabulary
SC2 - Character sets & information coding
SC6 - Telecommunications & information exchange between systems
SC7 - Software engineering
SC11 - Flexible magnetic media for digital data interchange
SC14 - Representation of data elements
SC15 - Labeling and file structure
SC17 - Identification cards & related devices
SC18 - Document processing and related communication
SC21 - Information retrieval, transfer & management for OSI
SC22 - Programming languages, their environments & systems software interfaces
SC23 - Optical disk cartridges for information interchange
SC24 - Computer graphics and image processing
SC25 - Interconnection of information technology equipment
SC26 - Microprocessor systems
SC27 - IT security techniques
SC28 - Office equipment
SC29 - Coded representation of picture, audio and multimedia/hypermedia information
BACKGROUND - V
ISO/IEC 12207

• **SPONSOR:**
  Joint Technical Committee 1 (JTC1) (Information Technology) of
  International Organization for Standardization (ISO) and
  International Electrotechnical Commission (IEC)
  - Developer: Subcommittee 7 (SC7) (Software Engineering)

• **HISTORY:**
  - Proposed in June 1988
  - 4 Working Drafts; 2 Committee Drafts; 1 DIS
  - Over 6 years and 17000 person-hours expended
  - Published 1 August 1995

• **PARTICIPANTS:**
  - Countries: Australia, Canada, Denmark, Finland, France, Germany,
    Ireland, Italy, Japan, Korea, Netherlands, Spain, Sweden, UK, USA
  - Convener: James Roberts (USA)
  - Editor: Raghu Singh (USA)
TOPICS

1. BACKGROUND

2. BASIC CONCEPTS
   - Principles and assumptions under 12207 development
   - Concepts for understanding the standard

3. THE PROCESSES

4. APPLICATION

5. RELATED AREAS

6. SUMMARY

7. FOR YOUR INFORMATION
BASIC CONCEPTS - I
LIFE CYCLE & ITS ARCHITECTURE

• TITLE OF 12207: *Software Life Cycle Processes*
  - Meaning: The processes in the life cycle of software

• THE ARCHITECTURING OF THE LIFE CYCLE:

```
RULES: MODULARITY; RESPONSIBILITY

RULE: PDCA cycle

LIFE CYCLE
FROM CONCEPTUALIZATION THROUGH RETIREMENT

ACTIVITY 1
TASKS

ACTIVITY N
TASKS

PROCESS 1
PROCESS ...
PROCESS n
```

PROCESS n
ACTIVITY 1
TASKS

ACTIVITY N
TASKS
**BASIC CONCEPTS - II**

**RULES FOR PARTITIONING THE LIFE CYCLE**

- **MODULARITY**
  - Cohesion (Functional): Tasks in a process are functionally related.
  - Coupling (Internal): Linkages among the processes be minimal
  - Association:
    - If a function is used by more than one process, then the function becomes a process in itself
    - If Process X is invoked by Process A and Process A only, then Process X belongs to Process A
  - Exception: Only for potential future application.

- **RESPONSIBILITY**
  - Each process is under a responsibility
  - A function whose parts are under different responsibilities shall not be a process
  - Contrast it with a process for a monolithic subject
    - Example: Quality management

- **Note:** The life cycle itself was not partitioned in time, as the life cycle of software follows its parent system’s life cycle.
BASIC CONCEPTS - III
THE PROCESS TREE

LIFE CYCLE

PRIMARY

SUPPORTING

DOCUMEN(TION
CONFIGURATION MANAGEMENT
QUALITY ASSURANCE
VERIFICATION
VALIDATION
JOINT REVIEW
AUDIT
PROBLEM RESOLUTION

ORGANIZATIONAL

MANAGEMENT
INFRASTRUCTURE
IMPROVEMENT
TRAINING

TAILORING
A PROCESS IS PARTITIONED INTO PDCA ACTIVITIES
- BASED THE PDCA-CYCLE PRINCIPLES

- PLAN
  Tasks, Assignments, Schedule, ...

- DO
  Execution of plans and tasks

- ACT
  Problem resolution, Corrective actions

- CHECK
  Evaluation, Assurance

- CLOSURE

PROCESS

INITIATION
BASIC CONCEPTS - V
ACTIVITY & TASKS

• AN ACTIVITY IS DIVIDED INTO TASKS, WHICH ARE GROUPED INTO SIMILAR ACTIONS

• TASK:
  - A what-to-do action; not a how-to-do action
  - Verbs used:
    
    VERB
    
    WILL (self-declaration)  +
    SHALL (requirement)
    SHOULD (recommendation)
    MAY (permission)
    CAN (possibility, when needed)

    MUST (unavoidable action), not used

    None of the above; present tense  #

# Not a requirement. Used in preamble, assumption, or to complete the context.
BASIC CONCEPTS - VI

• **BASED ON TQM PRINCIPLES**
  - Each party/participant has appropriate responsibility
  - Plan-Do-Check-Act (PDCA) cycle built into processes
  - Consistent with “functional” modularity and internal coupling

• **ESTABLISHES LINK WITH SYSTEM**
  - System activities are the foundation for software activities
    - Needs analysis, development, operation, maintenance, ...
    - Analysis, design, fabrication, integration, testing, ...
  - Software is treated as a part of the system
    - Necessary system context provided
  - System assigns functions to software
    - Software extracted from, developed within, and integrated back into the system
  - Software personnel participate in system activities
BASIC CONCEPTS - VII

• ORGANIZATION & PARTY
  - Organization: An independent body of persons
  - Party: One who enters into an agreement
  - Parties may be from the same or separate organization(s)
  - An organization (party) gets its name from the process it executes, and the name is functional
  - Acquirer executes Acquisition process

• MULTI ROLES:
  - A party may have more than one role
  - Example: A supplier with a sub-contractor is both supplier and acquirer.

• LEVELS OF APPLICATION
  - By a person as a self-imposed standard
  - By an organization internally
  - Between persons within an organization
  - Between two organizations
BASIC CONCEPTS - VIII

- RANGE OF AGREEMENT: From informal to legal contract.
- LANGUAGE
  - General: Introductory; to complete the situation/context; ...
  - Agreement: To facilitate self-imposed and contractual use
  - Active voice when party is clearly known
  - Passive voice when party is not clearly known, or when it is better syntactically
- PROJECT
  - A project may be solo
  - A project may exist in pre-agreement, agreement, or post-agreement phase, or a combination of the above
  - A project may span full or a part of life cycle
- DESIGNED FOR ADAPTATION AND TAILORING
  - Adaptation by an organization
  - Tailoring by a project
  - To fit the needs, size, complexity, cost, schedule, performance
BASIC CONCEPTS - IX
COMPLIANCE & CERTIFICATION

• COMPLIANCE

*Absolute level -- Default (IEC/ISO Directive 3, 4.1.2):*
All “shall” and “will” are performed.
- To claim compliance with the Standard.
- Note that compliance with the full Standard may not be realistic.

*Project level (ISO/IEC 12207, clause 1.4, paragraph 1):*
Parties develop an agreement, with which they comply.
- The agreement includes plans and tasks from 12207 and elsewhere.
- Parties perform in accordance with the agreement.
- 12207 itself then stays on the sidelines, its purpose served.

*Organizational level (ISO/IEC 12207, clause 1.4, paragraph 2):*
Organization declares public a set of clauses with which its suppliers comply.

• CERTIFICATION
- Not addressed in 12207
- Note that certifying an organization to the full Standard may not be realistic.
BASIC CONCEPTS - X
WHAT 12207 IS NOT

• NOT PRESCRIPTIVE; NO HOW-TOs
  - Responsive to evolving technologies
  - No interference in technical decision-making

• NOT A STANDARD FOR METHODS, TECHNIQUES & MODELS:
  - Does not prescribe management and engineering methods
  - Does not prescribe computer languages
  - Does not prescribe software engineering environments
  - Does not prescribe life-cycle or development models
    - Waterfall; incremental; evolutionary; Spiral, reengineering, ...

• NOT A STANDARD FOR PRODUCTS
  - Requires specific output groups be documented
  - But prescribes no formats, explicit contents, or media
    - An organization's product standards usable with 12207

• NOT A STANDARD FOR METRICS
  - Many tasks need metrics and indicators
  - But prescribes no specific metrics/indicators
  - References ISO/IEC 9126 for guidance
• A MANAGEMENT COMPLEMENT
  - 12207 complements institutionalized management
  - 12207 is not a substitute for systematic, disciplined management
  - It provides a powerful and complete but flexible set of
    building blocks of software life cycle for projects and
    organizations to use as appropriate and effective

• PREREQUISITES TO USING THE STANDARD:
  - Understanding of 12207
  - Organization's policies
  - Project's requirements and characteristics
  - Project's life-cycle model(s)
  - Institutionalization of methods, procedures, techniques,
    tools and environment for performing the 12207 and other tasks
  - Trained personnel

Instill life cycle view!
MAJOR ISSUES DURING 12207 DEVELOPMENT

• ARCHITECTURE:
  - Based on responsibilities; fix it for the parties? [U]
  - Acquisition, development, maintenance, ...
  - Monolithic topics; let the parties figure out?
    - Management, contracting, engineering, quality, ...

• SOFTWARE v. SYSTEM:
  - Include necessary system activities? [U]
  - Only software specific?

• LANGUAGE OF CLAUSES:
  - Declarative? [U]
  - Imperative?

• EVALUATIONS:
  - Assign evaluations to all parties appropriately? [C]
    - Do some evaluations become duplicative?
  - Place all evaluations under quality control?

• LEGEND: U- unanimous; C - consensus
TOPICS

1. BACKGROUND
2. BASIC CONCEPTS
3. THE PROCESSES
   - Introductory material
   - Explanation of the processes and their interactions
   - Coverage of special topics in the processes
   Note: In charts, start at if shown

4. APPLICATION
5. RELATED AREAS
6. SUMMARY
7. FOR YOUR INFORMATION
Preamble
- Abstract of the process
- Generic actions at corporate level
- In present tense
- No requirements
- Introduces a list of activities.

Activities & Tasks
- All with shall, will, should, may, can
- Except 5.1.1.1, 5.2.1.1 (present tense)
  - To complement the context
  - For project products and services

a.b.n: Activity n in process a.b
a.b.n.1: Task 1 in activity n, process a.b
... 
a.b.n.m: Task m in activity n, process a.b
DYNAMIC VIEW OF PROCESSES
THE WORKINGS

FUNCTION

PROCESS EMPLOYED

OUTPUT

CORPORATE MANAGEMENT OF PROCESSES
[NOT OF CORPORATION]

ORGANIZATIONAL PROCESSES

TAILORING PROCESS - FOR ADAPTATION

- INSTITUTIONALIZED PROCESSES
- MANAGED PROCESSES
- TAILORED/TOOLED PROCESSES FOR PROJECTS

PROJECT MANAGEMENT WITH PROCESSES

PRIMARY PROCESSES

SUPPORTING PROCESSES

TAILORING PROCESS

MANAGED PROJECT
STRATEGY FOR EXPLAINING THE PROCESSES IN THE TUTORIAL

• 12207 BEGINS THE DESCRIPTION OF EACH PRIMARY AND SUPPORTING PROCESS AT THE CORPORATE LEVEL BY INVOKING:
  - MANAGEMENT PROCESS
  - INFRASTRUCTURE PROCESS
  - IMPROVEMENT PROCESS
  - TRAINING PROCESS
  - TAILORING PROCESS

• 12207 CONTINUES WITH THE DESCRIPTION FURTHER AT THE PROJECT LEVEL -- IN ACTIVITIES AND TASKS.

• THIS PRESENTATION, FOR SAKE OF SPACE AND CLARITY, WILL EXPLAIN:
  - FIRST, THE PRIMARY AND SUPPORTING PROCESSES AT PROJECT LEVEL
  - THEN, THE ORGANIZATIONAL PROCESSES AT CORPORATE LEVEL.
REQUIREMENTS

Without requirements, there really is no project.

• REQUIREMENT:
  - An expectation/demand as a compliance/obligation/agreement
  - Indicated by a "shall" or a "will"
  - The qualifier to a requirement identifies its source and receiver; no qualifier means in local, immediate context
  - May contain constraints on design, interface, test, or development/test/operation/maintenance environment

• SPECIFICATION:
  - Technical description (form, fit, and function) of a requirement

• DYNAMICS:
  - A step may generate requirements for a neighboring or distant step.
  - Requirements/specifications typically change and expand in time.
  - The challenge is managing the changing requirements.
EVALUATION
It’s an elementary function!

ENTITY
Process, Activity, Task, Inputs, Outputs, Data, Product, Plan, Contract, Report, ...

PURPOSE
Check, Review, Audit, Verify, Validate, Assure, Inspect, Monitor, Control, Improve, ...

MOTIVE
Critique, Defend

FORUM
Diverse, Different, Formal, Informal, Peer, Independent

CRITERIA
At various levels: Requirements, Derived reqmts., Ad hoc conditions, ...

RESULTS; REPORTS

PROCESS 1
INTERNAL EVALUATIONS

BETWEEN PROCESSES

PROCESS 2
INTERNAL EVALUATIONS

1 EVALUATES 2

1 & 2 EVALUATE JOINTLY
PRIMARY PROCESSES

01: START HERE
O1, O2: THE SAME POINTS
E: EXECUTE; T: TASK; U: USE
ACQUISITION PROCESS

• FOR THE ACquirer OF SOFTWARE PRODUCTS AND SERVICES.
• COVERS PRE-CONTRACT AND CONTRACT PERIODS.

<table>
<thead>
<tr>
<th>ACTIVITIES</th>
<th>INTERNAL USE</th>
<th>INVOKED PROCESS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INITIATION</td>
<td>DEVELOPMENT</td>
<td></td>
<td>- SYSTEM REQS.</td>
</tr>
<tr>
<td>RFP [TENDER] PREPARATION</td>
<td></td>
<td></td>
<td>- ACQ. PLAN</td>
</tr>
<tr>
<td>CONTRACT PREPARATION &amp; UPDATE</td>
<td>TAILORING</td>
<td>INTERNAL CONTROL</td>
<td>- ACCEPTANCE CRITERIA</td>
</tr>
<tr>
<td>SUPPLIER MONITORING</td>
<td>JOINT REVIEW</td>
<td>AUDIT</td>
<td>ACQ. REQS. INCL.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VERIF.</td>
<td>- SELECTED TASKS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- REFERENCES TO CONTRACT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- CONTRACT WITH SUPPLIER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- CONTRACT WITH OTHERS</td>
</tr>
<tr>
<td>ACCEPTANCE &amp; COMPLETION</td>
<td></td>
<td>MONITORING &amp; EVALUATION RESULTS</td>
<td>ACCEPTED PRODUCTS &amp; SERVICES</td>
</tr>
</tbody>
</table>
ACQUISITION PROCESS

ACTIVITIES & TASKS

1. INITIATION
   - DESCRIBE THE NEEDS
   - DEFINE SYSTEM REQUIREMENTS
   - DEFINE SOFTWARE REQS. (MAYBE)
   - PREPARE ACQUISITION PLAN
   - DEFINE ACCEPTANCE STRATEGY

2. RFP [TENDER]
   - DOCUMENT ACQUISITION REQS.
   - SELECT ACTIVITIES & TASKS
   - DEFINE REFS. TO CONTRACT
   - SET REVIEW MILESTONES

3. CONTRACT PREPARATION & UPDATE
   - ESTABLISH SUPPLIER SELECTION PROCEDURES
   - SELECT SUPPLIER
   - TAILOR 12207;
     - INVOLVE PARTIES
   - NEGOTIATE CONTRACT

>> CONTRACT UNDERWAY

4. SUPPLIER MONITORING
   - MONITOR IN ACCORDANCE WITH JOINT REVIEW & AUDIT
   - SUPPLEMENT WITH V & V

5. ACCEPTANCE & COMPLETION
   - PREPARE FOR ACCEPTANCE; INCLUDE TESTS
   - CONDUCT ACCEPTANCE REVIEW & TESTING
   - ACCEPT DELIVERABLES
   - ASSUME CM
SUPPLY PROCESS

- FOR THE PROVIDER OF PRODUCTS/SERVICES.
- COVERS PRE-CONTRACT AND CONTRACT PERIODS.
SUPPLY PROCESS
ACTIVITIES & TASKS

1. INITIATION
• REVIEW RFP
• DECIDE TO BID; ACCEPT CONTRACT

2. PREPARATION OF RESPONSE
• PREPARE PROPOSAL

3. CONTRACT
• NEGOTIATE & ENTER INTO CONTRACT WITH ACQUIRER
• REQUEST MODS.

4. PLANNING
• REVIEW ACQ REQS
• SELECT LIFE CYCLE MODEL, AS NEEDED
• ESTABLISH REQS. FOR PLANS
• DEVELOP & DOCUMENT PROJECT MANAGEMENT PLANS [PMP; 15 ITEMS]

5. EXECUTION & CONTROL
• EXECUTE PMPs
• DEVELOP, OPERATE, OR MAINTAIN
• MONITOR/CONTROL PROGRESS/QUALITY
• MANAGE SUBS
• INTERFACE WITH IVVT
• INTERFACE WITH OTHER PARTIES

6. REVIEW & EVALUATION
• COORDINATE WITH ACQUIRER
• JOINT REVIEW
• AUDIT
• V&V
• ACCESS TO ACQUIRER
• QA PER QA PROCESS

7. DELIVERY & COMPLETION
• DELIVER PRODUCT OR SERVICE
• PROVIDE ASSISTANCE

>> CONTRACT UNDERWAY
# DEVELOPMENT PROCESS

- FOR THE DEVELOPER (MODIFIER) OF SOFTWARE PRODUCTS
- MAY PERFORM OR SUPPORT SOME SYSTEM ENGINEERING
- ACTIVITIES NOT NECESSARILY IN TIME ORDER

<table>
<thead>
<tr>
<th>ACTIVITIES</th>
<th>INTERNAL USE</th>
<th>INVOKED PROCESS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS IMPLEMENTATION</td>
<td>DOCUMENTATION</td>
<td>C.M.</td>
<td>DEVELOPMENT PLANS &amp; MODELS</td>
</tr>
<tr>
<td>SYS. REQ. ANALYSIS</td>
<td>EVALUATIONS</td>
<td>ISO/IEC 9126</td>
<td>SYSTEM REQUIREMENTS</td>
</tr>
<tr>
<td>SYS. ARCH. DESIGN</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>SYS. ARCHITECTURE - HW, SW, MO</td>
</tr>
<tr>
<td>SW REQ. ANALYSIS</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>SOFTWARE REQUIREMENTS</td>
</tr>
<tr>
<td>SW ARCH. DESIGN</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>SOFTWARE ARCHITECTURE</td>
</tr>
<tr>
<td>SW DET. DESIGN</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>SOFTWARE DETAILED DESIGN</td>
</tr>
<tr>
<td>SW CODE/TEST</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>SOFTWARE CODE/DATABASE</td>
</tr>
<tr>
<td>SW INTEGRATION</td>
<td>EVALUATIONS</td>
<td>JT. REVIEWS</td>
<td>INTEGRATED SOFTWARE (SCIs)</td>
</tr>
<tr>
<td>SW QUALIF. TEST</td>
<td>EVALUATIONS</td>
<td>AUDITS</td>
<td>QUALIFIED SCIs (DESIGN &amp; CODE)</td>
</tr>
<tr>
<td>SYS. INTEGRATION</td>
<td>EVALUATIONS</td>
<td>AUDITS</td>
<td>INTEGRATED SYSTEM</td>
</tr>
<tr>
<td>SYS. QUALIF. TEST</td>
<td>EVALUATIONS</td>
<td>AUDITS</td>
<td>QUALIFIED SYSTEM</td>
</tr>
<tr>
<td>SW INSTALLATION</td>
<td>EVALUATIONS</td>
<td>Installation Plan Installed SW</td>
<td>DELIVERABLE SOFTWARE</td>
</tr>
<tr>
<td>SW ACCPT. SUPPORT</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• MAY PERFORM OR SUPPORT SOME SYSTEM ENGINEERING
• ACTIVITIES NOT NECESSARILY IN TIME ORDER

- FOR THE DEVELOPER (MODIFIER) OF SOFTWARE PRODUCTS

- ACTIVITIES NOT NECESSARILY IN TIME ORDER

- MAY PERFORM OR SUPPORT SOME SYSTEM ENGINEERING

ISO/IEC 9126
DEVELOPMENT PROCESS
ACTIVITIES & TASKS

1. PROCESS IMPLEMENTATION

• DEFINE/SELECT DEVELOPMENT MODEL(s)
  - The foundation of a project
  - Detailed with iterations/recursions of the
    activities & tasks of Development and
    invoked Supporting processes

• EMPLOY REGULARLY DOC, CM,
  AND PROB. RES. PROCESSES

• SELECT/TAILOR INTERNAL
  METHODS/TOOLS/...

• DEVELOP, DOCUMENT,
  EXECUTE PLANS

• MAY USE NON-DELIVERABLES
  - Avoid dependency of future
    operations & maintenance on these

2,3,10,11. SYSTEM ACTIVITIES

• PERFORM OR SUPPORT

4-9, 12,13. SOFTWARE ACTIVITIES

• PERFORM

3. SYSTEM

ARCHITECTURAL DESIGN

• PRODUCE AN ARCHITECTURE
  OF THE SYSTEM

• IDENTIFY HW, SW AND
  MANUAL OPERATIONS ITEMS

5. SOFTWARE

ARCHITECTURAL DESIGN

• PRODUCE AN ARCHITECTURE
  OF THE SOFTWARE ITEM

• IDENTIFY COMPONENTS
  OF THE SOFTWARE ITEM

7. SOFTWARE CODING & TESTING

• CODE UNITS & DATABASES

• IF NEEDED, CODE SHOULD BE
  COMPILABLE AND TESTABLE

8. SOFTWARE INTEGRATION

10. SYSTEM INTEGRATION

• INTEGRATE IN AGGREGATES

• PARTITION AND INTEGRATION
  PATHS MAY BE DIFFERENT
LEGEND: CI: CONFIGURATION ITEM; HW: HARDWARE; SW: SOFTWARE; FW: FIRMWARE, PW: PEOPLEWARE PROC: PROCESS; SVC: SERVICE.

• LOWER CIs MAY BE PARTITIONED FURTHER UNTIL SUITABLE FOR INDIVIDUAL TREATMENT
• THE CIs MAY BE NOT ALL DISTINCT
• PARTITIONING AND INTEGRATION PATHS CAN BE DIFFERENT

+ CONTINUED TO THE NEXT CHART
SOFTWARE ORGANIZATION
[CI-2222 FROM THE PREVIOUS CHART]

LEGEND:
SC - SOFTWARE COMPONENT; SU - SOFTWARE UNIT

• 12207 ASKS FOR ARCHITECTURE AND DESIGN, BUT DOES NOT IMPLY STYLE, REPRESENTATION OR DERIVATION METHODS
• SU's MAY BE NOT ALL DISTINCT
• IF NEEDED, AN SU MUST BE COMPILABLE AND TESTABLE
• DECOMPOSITION AND INTEGRATION PATHS MAY BE DIFFERENT
**DEVELOPMENT PROCESS**

**THE HORSE**

- **ITERATIONS/RECURSIONS ENCOURAGED:**
  - TO BUILD SPECIFIC MODELS
  - TO OFFSET DEFAULT MODELS

- **ALL ACTIVITIES [TASKS] IN A PROCESS [ACTIVITY] NOT NEEDED IN EACH ITERATION OR RECURSION, BUT SHOULD BE COMPLETED BY THE FINAL ITERATION OR RECURSION**

- **PROJECTS ESTABLISH BASELINES**
  - OF WHAT & WHEN
  - BASELINING AT PRE-DETERMINED REVIEWS/AUDITS
  - FORUM TO INVOLVE KEY PARTIES
  - BASELINES INHIBIT UNPLANNED OR EASY CHANGES
  - AT LEAST 3 BASELINED PRODUCTS:
    - REQUIREMENTS, DESIGN, CODE
OPERATION PROCESS

• FOR THE OPERATOR OF A SYSTEM CONTAINING SOFTWARE

ACTIVITIES | INTERNAL USE | INVOKED PROCESS | OUTPUTS
--- | --- | --- | ---
PROCESS IMPLEMENTATION | PROBLEM RESOLUTION | MAINTENANCE | - OPERATION PLAN
- OPERATION PROCEDURES
OPERATIONAL TESTING | INTERNAL TESTING & ENSURANCE | | RELEASED OPERATIONAL SOFTWARE
SYSTEM OPERATION | | | [FUNCTIONS PERFORMED]
USER SUPPORT | | | - USER REQUESTS
- PROBLEM RESOLUTIONS
OPMATION PROCESS
ACTIVITIES & TASKS

1. PROCESS IMPLEMENTATION
   • DEVELOP OPERATIONAL PLAN
   • SET OPERATIONAL STANDARDS
   • DOCUMENT AND EXECUTE PLAN
   • ESTABLISH PROCEDURES FOR/WITH PROBLEM RESOLUTIONS
   • ESTABLISH PROCEDURES FOR OPERATIONAL TESTING
   • ESTABLISH PROCEDURES FOR INTERFACING WITH MAINTENANCE PROCESS
   • ESTABLISH PROCEDURES FOR RELEASING PRODUCTS FOR OPERATIONAL USE

2. OPERATIONAL TESTING
   • PERFORM OPERATIONAL TESTING FOR EACH RELEASE
   • RELEASE AFTER CRITERIA MET
   • ENSURE CODE/DATABASE PERFORM AS PLANNED

3. SYSTEM OPERATION
   • OPERATE IN ENVIRONMENT

4. USER SUPPORT
   • PROVIDE ASSISTANCE TO USERS
   • FORWARD USER REQUESTS TO MAINTENANCE AS NEEDED
   • FOR TEMPORARY FIXES, PROVIDE OPTION TO USE IT
MAINTENANCE PROCESS

- FOR THE MAINTAINER OF SOFTWARE PRODUCTS

<table>
<thead>
<tr>
<th>ACTIVITIES</th>
<th>INTERNAL USE</th>
<th>INVOKED PROCESS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS IMPLEMENT</td>
<td></td>
<td>PROBLEM RESOLUTION</td>
<td>MAINTENANCE PLANS/PROCS.</td>
</tr>
<tr>
<td>PROB/MOD. ANALYSIS</td>
<td></td>
<td>CM</td>
<td>PROB./MOD. ANAL/SOLN.</td>
</tr>
<tr>
<td>MOD. IMPLEMENT.</td>
<td></td>
<td>DEVELOPMENT</td>
<td>MODIFIED SOFTWARE</td>
</tr>
<tr>
<td>MAINT. REVIEW/ACCEPT.</td>
<td>INTERNAL REVIEWS</td>
<td></td>
<td>REVIEW RESULTS</td>
</tr>
<tr>
<td>MIGRATION</td>
<td>INTERNAL REVIEWS</td>
<td></td>
<td>- MIGRATION PLANS/REPORTS</td>
</tr>
<tr>
<td>SOFTWARE RETIREMENT</td>
<td></td>
<td></td>
<td>- MIGRATED SYS.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- RETIREMENT PLANS</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- ARCHIVES</td>
</tr>
</tbody>
</table>

FOR THE MAINTAINER OF SOFTWARE PRODUCTS

- INVOKED PROCESS
  - RETIREMENT PLANS
  - ARCHIVES
MAINTENANCE PROCESS ACTIVITIES & TASKS

1. PROCESS IMPLEMENTATION
   - Develop, document and execute plan
   - Establish procedures for problem reports and modifications requests
   - Manage modifications

2. PROBLEM/MODIFICATION ANALYSIS
   - Analyze modifications for impacts
   - Replicate/verify problems
   - Implement modifications
   - Document and get approval

3. MODIFICATION IMPLEMENTATION
   - Determine targets of modifications
   - Use Development Process for mods.
   - Supplement with testing to ensure modified and unmodified parts are correctly done

4. MAINTENANCE REVIEW/ACCEPTANCE
   - Review with authorizing organization

5. MIGRATION
   - Develop/document/execute plan
   - Notify users, etc.
   - Do parallel operations
   - Do post-operations for impact

6. SOFTWARE RETIREMENT
   - Develop/document/execute plan
   - Notify users, etc.
   - Do parallel operations
   - Provide for access to retired data/products
SUPPORTING PROCESSES

• TO SUPPORT ANOTHER PROCESS BY PERFORMING A SPECIFIC FUNCTION

ARROWS: EMPLOY/INVOKE
DOCUMENTATION PROCESS

- FOR ESTABLISHING DOCUMENTATION STANDARDS:
  - MEDIA, FORMAT, OUTLINE, CONTENT, FILING, DISTRIBUTION, ...
  - EXAMPLES: YOUR ORGANIZATION'S USER'S MANUALS;
    - US DOD DIDs; IEEE SRS, ...
- THIS PROCESS DOES NOT PRESCRIBE ANY OF ABOVE;
  - THE INVOKING PROCESS DOES

ACTIVITIES | INTERNAL USE | INVOKED PROCESS | OUTPUTS
---|---|---|---
PROCESS IMPLEMENTATION | DOCUMENTATION PLAN
DESIGN AND DEVELOPMENT | "PREPARED" DOCUMENTS
PRODUCTION | PRODUCED DOCUMENTS
MAINTENANCE | MODIFIED DOCUMENTS

CONFIGURATION MANAGEMENT
CONFIGURATION MANAGEMENT PROCESS

- FOR CM OF PRODUCTS AND TASKS
- INTERNAL OR EXTERNAL TO THE INVOKING PROCESS
- THIS PROCESS IDENTIFIES BASELINED PRODUCTS; THE INVOKING PROCESS ESTABLISHES BASELINES

ACTIVITIES

- PROCESS IMPLEMENTATION
- CONFIGURATION IDENTIFICATION
- CONFIGURATION CONTROL
- CONFIGURATION STATUS ACCOUNTING
- CONFIGURATION EVALUATION
- RELEASE MANAGEMENT & DELIVERY

INTERNAL USE

- INTERNAL ACCESS CONTROL & AUDIT
- INTERNAL EVALUATION

INVOKED PROCESS

- CONFIGURATION MANAGEMENT PLAN
- IDENTIFICATION SCHEMA
- BASELINING DOCUMENT
- CONFIGURATION CONTROL RESULTS
- CONFIGURATION STATUS REPORTS
- EVALUATION REPORTS
- DELIVERABLE PRODUCTS

OUTPUTS
QUALITY ASSURANCE PROCESS

• FOR ASSURING CONFORMANCE OF PRODUCTS/SERVICES WITH REQUIREMENTS AND ADHERENCE TO PLANS
• EXTERNAL, WITH ORGANIZATIONAL FREEDOM
• USES THE TERM "ASSURE" INSTEAD OF "EVALUATE"

ACTIVITIES

INTERNAL USE

INVOKED PROCESS

OUTPUTS

PROCESS IMPLEMENTATION

V&V, JT. REVIEW, AUDIT AS TECHNIQUES

PROBLEM RESOLUTION

QUALITY ASSURANCE PLAN

PRODUCT ASSURANCE

INCLUDE RESULTS OF V&V, JT. REVIEW, AUDIT, AND INTERNAL EVALUATIONS

PRODUCTS ASSURED

PROCESS ASSURANCE

ASSURANCE OF QUALITY SYSTEMS

ISO 9001

PROCESSSES ASSURED

AS SPECIFIED IN THE CONTRACT
# QUALITY ASSURANCE PROCESS ACTIVITIES AND TASKS

## 1. PROCESS IMPLEMENTATION

- Establish QA process for the project
- Develop/document/execute QA plan
- Coordinate with verification, validation, joint review, audit processes

## 2. PRODUCT ASSURANCE

Assure that:
- Plans are documented/compliant/executed
- Products/documentation are compliant & adherent
- Products are deliverable/acceptable to acquirer

## 3. PROCESS ASSURANCE

Assure that:
- Employed processes are compliant & adherent
- Internal engineering practices are compliant
- Prime requirements are passed down to subcontractors
- Other parties are provided support
- Trained staff and training are in place

## 4. ASSURANCE OF QUALITY SYSTEM

- Additional quality management per ISO 9001
VERIFICATION PROCESS

- FOR VERIFICATION OF REQUIREMENTS IN AN ACTIVITY AGAINST THOSE IN PREVIOUS ACTIVITY
- INTERNAL OR INDEPENDENT
- USES THE TERM "VERIFY" INSTEAD OF "EVALUATE"
- PRIMARILY FOR DEVELOPMENT PROCESS;
  - OPERATION AND MAINTENANCE NOT COVERED.

<table>
<thead>
<tr>
<th>ACTIVITIES</th>
<th>INTERNAL USE</th>
<th>INVOKED PROCESS</th>
<th>OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS IMPLEMENTATION</td>
<td>PROBLEM RESOLUTION</td>
<td>VERIFICATION PLAN</td>
<td></td>
</tr>
</tbody>
</table>

VERIFICATION
- CONTRACT
- PROCESS
- REQUIREMENTS
- DESIGN
- CODE
- INTEGRATION
- DOCUMENTATION

EACH WITH OWN CRITERIA

VERIFIED PRODUCTS AND SERVICES
VERIFICATION PROCESS
ACTIVITIES AND TASKS

1. PROCESS IMPLEMENTATION
   • DETERMINE IF AND HOW MUCH NEEDED
     - USE CRITICALITY FACTORS
   • DETERMINE DEGREE OF INDEPENDENCE

2. CONTRACT VERIFICATION
   • SUPPLIER IS CAPABLE
   • USER NEEDS ARE COVERED
   • HANDLING REQS CHANGES ADEQUATE
   • PARTIES' INTERFACES STIPULATED

3. PROCESS VERIFICATION
   • PLANNING ADEQUATE/TIMELY
   • PROCESSES ADEQUATE/IMPLEMENTED BEING EXECUTED/COMPLIANT
   • STANDARDS/PROCEDURES/ENVIRONMENTS ADEQUATE
   • PERSONNEL STAFFED AND TRAINED

4. REQS. VERIFICATION
   • CONSISTENT/FEASIBLE/TESTABLE
   • ALLOCATIONS APPROPRIATE
   • CRITICALITY REQS. CORRECT BY RIGOROUS METHODS

5. DESIGN VERIFICATION
   • CORRECT/CONSISTENT/TRACEABLE
   • PROPER SEQUENCE/ALLOCATION OF EVENTS, I/O, INTERFACES, LOGIC, TIMING, SIZING, RECOVERY, ...
   • DESIGN IMPLEMENTS CRITICALITY REQS. CORRECTLY
     [SHOW BY RIGOROUS METHODS]

6. CODE VERIFICATION
   • CORRECT/TESTABLE/TRACEABLE
   • SIMILAR TO ABOVE

7. INTEGRATION VERIFICATION
   • COMPONENTS/UNITS INTEGRATED COMPLETELY/CORRECTLY
   • ITEMS INTEGRATED INTO SYSTEM COMPLETELY AND CORRECTLY
   • PERFORMED PER PLANS

8. DOC. VERIFICATION
   • ADEQUATE/COMPLETE/CONSISTENT
   • TIMELY
   • follows CM
VALIDATION PROCESS

- FOR VALIDATION OF AS-BUILT PRODUCTS AGAINST SPECIFIED CRITERIA
- INTERNAL OR INDEPENDENT
- USES THE TERM "VALIDATE" INSTEAD OF "EVALUATE"
- CONFIDENCE IN VALIDATION: THROUGH TESTING

ACTIVITIES

PROCESS IMPLEMENTATION

VALIDATION
4/5 TASKS TESTING
1 TASK: INTENDED USE

PROBLEM RESOLUTION

VALIDATION PLAN

INTERNAL USE

INVOKED PROCESS

OUTPUTS

VALIDATED PRODUCTS AND SERVICES
JOINT REVIEW PROCESS

• FOR JOINT REVIEWS BETWEEN REVIEWER AND REVIEWEE
  - TYPICALLY BY SUPPLIER WITH ACQUIRER
  - BOTH TECHNICAL AND MANAGEMENT

• REVIEW OF PROJECT STATUS, PRODUCTS AND TASKS
  FOR COMPLETENESS, COMPLIANCE AND ADHERENCE

ACTIVITIES  INTERNAL USE  INVOKED PROCESS  OUTPUTS

PROCESS IMPLEMENTATION  PROBLEM RESOLUTION  AGENDA, SCOPE, FORUM, ETC.

PROJECT MANAGEMENT REVIEWS

TECHNICAL REVIEWS  PROJECT STATUS RESULTS

REVIEW RESULTS
AUDIT PROCESS

• FOR AUDITS BETWEEN AUDITOR AND AUDITEE
  - TYPICALLY BY ACQUIRER WITH SUPPLIER

• FOR COMPLIANCE WITH REQUIREMENTS/PLANS/CONTRACT

![Diagram of Audit Process]

ACTIVITIES  INTERNAL USE  INVOKED PROCESS  OUTPUTS

PROCESS IMPLEMENTATION  PROBLEM RESOLUTION  AGENDA, SCOPE, FORUM, ETC.

AUDIT

AUDIT RESULTS
PROBLEM RESOLUTION PROCESS

• FOR ANALYZING AND RESOLVING PROBLEMS, TAKING CORRECTIVE ACTIONS, AND DETECTING TRENDS - THE “A” OF THE PDCA CYCLE.

• A CLOSED LOOP PROCESS:
  - PROBLEMS REPORTED/ENTERED
  - ACTION TAKEN
  - CAUSES IDENTIFIED/ELIMINATED
  - RESOLUTION/DISPOSITION ACHIEVED/RECORDED
  - TRENDS DETECTED

• NOTE: NOT EVERY PROBLEM NEEDS CORRECTIVE ACTION

ACTIVITIES

PROCESS IMPLEMENTATION

PROBLEM RESOLUTION

INTERNAL USE

INVOKED PROCESS

OUTPUTS

RESOLVED PROBLEMS
ORGANIZATIONAL PROCESSES

- FOR AN ORGANIZATION TO MANAGE AND IMPROVE ITS PROCESSES AT CORPORATE LEVEL
- THE ORGANIZATION IS RESPONSIBLE FOR ESTABLISHING AND INSTITUTIONALIZING THE LIFE CYCLE PROCESSES

1: MANAGE FOLLOWING MANAGEMENT PROCESS
2: ESTABLISH INFRASTRUCTURE FOLLOWING INFRASTRUCTURE PROCESS
3: IMPROVE FOLLOWING IMPROVEMENT PROCESS
4: TRAIN PERSONNEL FOLLOWING TRAINING PROCESS
MANAGEMENT PROCESS

- FOR GENERAL MANAGEMENT OF PROCESSES
- IT IS THE PDCA CYCLE
- IT IS INSTANTIATED IN OTHER PROCESSES
INFRASTRUCTURE PROCESS

• FOR ESTABLISHING AND MAINTAINING INFRASTRUCTURE OF A LIFE CYCLE PROCESS

• INFRASTRUCTURE: PROCEDURES, STANDARDS, TOOLS, EQUIPMENT, SPACE

ACTIVITIES  INTERNAL USE  INVOKED PROCESS  OUTPUTS

PROCESS IMPLEMENTATION

ESTABLISHMENT OF THE INFRASTRUCTURE

MAINTENANCE OF THE INFRASTRUCTURE

INFRASTRUCTURE

CONFIGURATION OF THE INFRASTRUCTURE

[RECORDS]
TRAINING PROCESS

• FOR PROVIDING AND MAINTAINING TRAINED PERSONNEL

ACTIVITIES

INTERNAL USE

INVOKED PROCESS

OUTPUTS

PROCESS IMPLEMENTATION

TRAINING MATERIAL DEVELOPMENT

TRAINING PLAN IMPLEMENTATION

TRAINING PLAN

TRAINING MANUALS

TRAINING RECORDS [TRAINED PERSONNEL]
IMPROVEMENT PROCESS

• FOR ESTABLISHING, ASSESSING, MEASURING, CONTROLLING, AND IMPROVING A LIFE CYCLE PROCESS

[RESTATED PROCESS(ES)]

[ASSESSMENT PROCEDURES AND PLANS]

[evalu水中, historical, quality cost records]
TAILORING PROCESS
A SPECIAL PROCESS

- FOR BASIC TAILORING OF THE STANDARD FOR PROJECTS
  - ADDITIONS IN AGREEMENT
  - ONLY 5 “SHALLS”
- TAILORING OF THIS PROCESS NOT ALLOWED

**ACTIVITIES**

1. **IDENTIFYING PROJECT ENVIRONMENT**
2. **SOLICITING INPUTS**
3. **SELECTING PROCESSES, ACTIVITIES, & TASKS**
4. **DOCUMENTING TAILORING DECISIONS & RATIONALE**

**INTERNAL USE**

**INVOKED PROCESS**

**OUTPUTS**

- PROJECT'S CHARACTERISTICS
- INPUTS FROM ORGANIZATIONS
- SELECTED PROCESSES, ACTIVITIES, & TASKS
- TAILORING DECISIONS & RATIONALE
EVALUATION-BASED PROCESSES
THE “C” OF THE PDCA CYCLE

- EVALUATION: BASIC TO ALL EVALUATION-BASED TASKS
- PROCESS-INTERNAL EVALUATIONS: AGAINST SPECIFIED CRITERIA
- VERIFICATION: AGAINST PREVIOUS ACTIVITIES
- VALIDATION: AGAINST INTENDED USE
- QA: ASSURANCE WITH RESPECT TO REQUIREMENTS/PLANS
- JOINT REVIEW: EVALUATIONS OF STATUS & PRODUCTS
- AUDIT: EVALUATIONS FOR COMPLIANCE WITH REQUIREMENTS/PLANS/CONTRACT
# REQUIREMENTS

- **AN EXPECTATION/DEMAND**  
  AS A COMPLIANCE/OBLIGATION/AGREEMENT
- **A QUALIFIER IDENTIFIES THE SOURCE & RECEIVER; OTHERWISE IN LOCAL CONTEXT**

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>TERM &quot;REQUIREMENTS&quot; and QUALIFIER USED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACQUISITION</strong></td>
<td>• ACQUISITION REQS; SYSTEM REQS; SOFTWARE REQS</td>
</tr>
</tbody>
</table>
| **SUPPLY**    | • ACQUISITION REQS  
  • PLANNING REQS; CONTRACTUAL REQS; PRIME-CONTRACT REQS |
| **DEVELOPMENT** | • SYSTEM REQS & SPECS  
- ORGANIZATIONAL, USER, SAFETY, INTERFACE, QUALIFICATION, ...  
  • SYSTEM REQS ALLOCATED TO ITEMS: HARDWARE, SOFTWARE, MANUAL OPERATIONS  
  • SOFTWARE REQS FOR: ITEMS; COMPONENTS; UNITS  
  • TEST REQS |
| **MAINTENANCE** | • NEW AND MODIFIED REQS |
| **OTHER**     | • FOR LOCAL ACTION ON INCOMING REQS |
### CRITICAL FUNCTIONS

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>CRITICAL-FUNCTION RELATED TASKS</th>
</tr>
</thead>
</table>
| **ACQUISITION**| • Define Safety/Security/Criticality Requirements.  
                • Include Related Design/Testing/Compliance Standards/Procedures. |
| **SUPPLY**     | • Address in Project Plans, Management of:  
                - Safety/Security/Criticality Requirements  
                - Related Policy/Regulation/Certification  
                • Separate Plans Encouraged. |
| **DEVELOPMENT**| • Address Planning, Analysis, Design, and Qualification of Safety, Security, and Criticality Requirements, Including Ergonomic. |
| **MAINTENANCE**| • Analyze Impact of Modifications on:  
                Safety/Security/Criticality Functions. |
| **DOCUMENTATION**| • Produce/Store Documents Per Security Policies. |
| **C. M. PROCESS**| • Control/Audit Access To Software Processing Safety/Security Critical Functions. |
| **VERIFICATION**| • Determine Verification Effort Per Criticality Reqsmts.  
                • Verify By Rigorous Methods Safety/Security/Criticality Functions Are Analyzed/Designed/Coded Correctly. |
# TESTING

- TESTING SHARED AMONG THE PARTIES

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>TESTING RELATED TASKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACQUISITION</td>
<td>• DEFINE ACCEPTANCE STRATEGY &amp; CRITERIA.</td>
</tr>
<tr>
<td></td>
<td>• PREPARE TESTS AND TEST ENVIRONMENT FOR ACCEPTANCE.</td>
</tr>
<tr>
<td></td>
<td>• IDENTIFY TEST STANDARDS &amp; PROCEDURES FOR CRITICAL REQS.</td>
</tr>
<tr>
<td></td>
<td>• CONDUCT VALIDATION &amp; ACCEPTANCE TESTING.</td>
</tr>
<tr>
<td>SUPPLY</td>
<td>• PLAN TEST ENVIRONMENT.</td>
</tr>
<tr>
<td></td>
<td>• INTERFACE WITH IVV&amp;T AGENT.</td>
</tr>
<tr>
<td></td>
<td>• SUPPORT ACCEPTANCE &amp; VALIDATION TESTING.</td>
</tr>
<tr>
<td>DEVELOPMENT</td>
<td>• DEFINE TESTS; PLAN AND DO TESTING:</td>
</tr>
<tr>
<td></td>
<td>- UNITS, DATABASES, AGGREGATES</td>
</tr>
<tr>
<td></td>
<td>- INTERNAL, INTEGRATION, QUALIFICATION, CONFORMANCE, INSTALLATION</td>
</tr>
<tr>
<td></td>
<td>- INCLUDE STRESS TESTS AND TESTING</td>
</tr>
<tr>
<td></td>
<td>• EVALUATE FOR TESTABILITY, TEST COVERAGE, TEST FEASIBILITY.</td>
</tr>
<tr>
<td></td>
<td>• SUPPORT ACCEPTANCE TESTING.</td>
</tr>
<tr>
<td>OPERATION</td>
<td>• DEFINE OPERATIONAL TESTS.</td>
</tr>
<tr>
<td></td>
<td>• TEST IN OPERATIONAL ENVIRONMENT.</td>
</tr>
<tr>
<td></td>
<td>• CONDUCT OPERATIONAL TESTING FOR EACH RELEASE.</td>
</tr>
<tr>
<td>MAINTENANCE</td>
<td>• WHEN MODIFICATION, DO DEVELOPMENT TESTING.</td>
</tr>
<tr>
<td></td>
<td>• TEST MODIFIED AND UNMODIFIED PARTS.</td>
</tr>
<tr>
<td></td>
<td>• DO MIGRATION TESTING.</td>
</tr>
<tr>
<td>VALIDATION</td>
<td>• DEFINE AND CONDUCT VALIDATION TESTS.</td>
</tr>
<tr>
<td></td>
<td>• INCLUDE STRESS TESTING.</td>
</tr>
</tbody>
</table>
# Off-the-Shelf Software (OTSS) & Non-Deliverable Items (NDI)

## Process

<table>
<thead>
<tr>
<th>Process</th>
<th>OTSS &amp; NDI Related Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scope</strong></td>
<td>- 12207 is not intended for OTSS, unless incorporated into deliverables</td>
</tr>
<tr>
<td></td>
<td>- Consider OTSS as an option in acquisition or parts of acquisition</td>
</tr>
<tr>
<td></td>
<td>- Ensure the following if acquiring OTSS:</td>
</tr>
<tr>
<td></td>
<td>- Requirements are satisfied</td>
</tr>
<tr>
<td></td>
<td>- Documentation available</td>
</tr>
<tr>
<td></td>
<td>- Rights are satisfied</td>
</tr>
<tr>
<td></td>
<td>- Future support planned</td>
</tr>
<tr>
<td><strong>Acquisition</strong></td>
<td>- Consider OTSS in development</td>
</tr>
<tr>
<td><strong>Supply</strong></td>
<td>- Consider OTSS in development [via supply process].</td>
</tr>
<tr>
<td></td>
<td>- NDI may be used in development,</td>
</tr>
<tr>
<td></td>
<td>- Ensure operation &amp; maintenance independent of NDI</td>
</tr>
<tr>
<td></td>
<td>- Otherwise NDI should become deliverable</td>
</tr>
</tbody>
</table>
# METRICS & INDICATORS - I

- 12207 NEEDS THESE, BUT DOES NOT DEFINE OR PRESCRIBE THEM

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>METRIC/INDICATOR NEEDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACQUISITION</td>
<td>• PROCESS MONITORING</td>
</tr>
<tr>
<td></td>
<td>- Cost, Schedule, Technical</td>
</tr>
<tr>
<td></td>
<td>• SUPPLIER SELECTION</td>
</tr>
<tr>
<td></td>
<td>- Capability, Past performance, ...</td>
</tr>
<tr>
<td></td>
<td>• PROPOSAL EVALUATION</td>
</tr>
<tr>
<td></td>
<td>- Technical, Cost, Schedule, Personnel, ...</td>
</tr>
<tr>
<td></td>
<td>• AGREEMENT CHANGES</td>
</tr>
<tr>
<td></td>
<td>- No., Rate, Impact, ...</td>
</tr>
<tr>
<td></td>
<td>• ACCEPTANCE PROGRESS</td>
</tr>
<tr>
<td></td>
<td>- Acceptance criteria, Conformance, Releasability, ...</td>
</tr>
<tr>
<td></td>
<td>• JOINT ACTION ITEMS STATUS</td>
</tr>
<tr>
<td>SUPPLY</td>
<td>• BID DECISION</td>
</tr>
<tr>
<td></td>
<td>• PROCESS MONITORING</td>
</tr>
<tr>
<td></td>
<td>- Cost, Schedule, Technical</td>
</tr>
<tr>
<td></td>
<td>• PROBLEM STATUS</td>
</tr>
<tr>
<td></td>
<td>- By Activity/Task/Source, Trend, ...</td>
</tr>
<tr>
<td></td>
<td>• ACCEPTANCE PROGRESS</td>
</tr>
<tr>
<td></td>
<td>- Acceptance criteria, Conformance, Releasability, ...</td>
</tr>
<tr>
<td></td>
<td>• JOINT ACTION ITEMS STATUS</td>
</tr>
</tbody>
</table>
## METRICS & INDICATORS - II

- 12207 NEEDS THESE, BUT DOES NOT DEFINE OR PRESCRIBE THEM

<table>
<thead>
<tr>
<th>PROCESS</th>
<th>METRIC/INDICATOR NEEDED</th>
</tr>
</thead>
</table>
| DEVELOPMENT            | • CHANGE STATUS: By Activity/Task, Source, Trend, ...
                          | • PROBLEM STATUS: BY Activity/Task, Source, Trend, ...
<pre><code>                      | • JOINT ACTION ITEMS STATUS                                                                                                                                 |
                      | • TRACEABILITY:                                                                                                                                                                                                 |
                      |   - System Requirements to Acquisition Needs                                                                                                             |
                      |   - System Architectural Design to System Requirements                                                                                                  |
                      |   - Software Requirements to System Requirements &amp; Design                                                                                               |
                      |   - Software Architectural Design to Software Requirements                                                                                              |
                      |   - Software Detailed Design to Software Requirements                                                                                                  |
                      |   - Software Unit to Software Requirements &amp; Design                                                                                                     |
                      |   - Software Design &amp; Unit to System Requirements                                                                                                     |
                      | • QUALITY CHARACTERISTICS [ISO/IEC 9126]                                                                                                               |
                      |   - Functionality, Reliability, Usability, Efficiency, Maintainability, Portability                                                                  |
                      |   - Plus their sub-characteristics                                                                                                                      |
                      | • REQUIREMENTS TESTABILITY STATUS                                                                                                                      |
                      | • TEST COVERAGE                                                                                                                                   |
                      | • CONSISTENCY: INTERNAL &amp; EXTERNAL                                                                                                                      |
                      | • CONFORMANCE TO EXPECTED RESULTS                                                                                                                     |
                      | • FEASIBILITY OF NEXT ACTIVITY                                                                                                                         |
                      | • FEASIBILITY OF OPERATIONS                                                                                                                             |
                      | • FEASIBILITY OF MAINTENANCE                                                                          |
</code></pre>
<table>
<thead>
<tr>
<th>PROCESS</th>
<th>METRIC/INDICATOR NEEDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPERATION</td>
<td>• OPERATIONAL CHARACTERISTICS</td>
</tr>
<tr>
<td></td>
<td>- Run time, Throughput, Availability, Responsiveness, ...</td>
</tr>
<tr>
<td></td>
<td>• OPERATIONAL TESTING</td>
</tr>
<tr>
<td></td>
<td>- Coverage, Releasability, ...</td>
</tr>
<tr>
<td></td>
<td>• USER SUPPORT</td>
</tr>
<tr>
<td></td>
<td>- Status of requests, support, releases, ...</td>
</tr>
<tr>
<td>MAINTENANCE</td>
<td>• STATUS: PROBLEM REPORTS &amp; MODIFICATION REQUESTS</td>
</tr>
<tr>
<td></td>
<td>- Measure of classification, size, criticality, closure, ...</td>
</tr>
<tr>
<td></td>
<td>- Impact on operations &amp; maintenance</td>
</tr>
<tr>
<td></td>
<td>• TEST COVERAGE OF</td>
</tr>
<tr>
<td></td>
<td>- Modified parts</td>
</tr>
<tr>
<td></td>
<td>- Unmodified parts</td>
</tr>
<tr>
<td></td>
<td>• IMPACT ON UNMODIFIED PARTS</td>
</tr>
<tr>
<td></td>
<td>• MIGRATION PORTABILITY</td>
</tr>
<tr>
<td></td>
<td>• USER SUPPORT DURING MIGRATION</td>
</tr>
<tr>
<td></td>
<td>• POST-OPERATION IMPACT OF MIGRATION</td>
</tr>
<tr>
<td></td>
<td>• USER SUPPORT DURING RETIREMENT</td>
</tr>
</tbody>
</table>

• 12207 NEEDS THESE, BUT DOES NOT DEFINE OR PRESCRIBE THEM
TOPICS

1. BACKGROUND

2. BASIC CONCEPTS

3. THE PROCESSES

4. APPLICATION
   - Business practices under debate
   - How to use 12207 in an organization and on a project
   - Pertinent advice and notes
Note: No rules; only salient guidelines -- this presenter’s
Note: In charts, start at if shown

5. RELATED AREAS

6. SUMMARY

7. FOR YOUR INFORMATION
BUSINESS PRACTICES UNDER DEBATE

MARKET

PUBLIC

SUPPLIER (VENDOR)

ESTABLISH AN ENVIRONMENT WITH METHODS, TECHNIQUES AND TOOLS FOR 12207 AND OTHER STANDARDS AND FOR THE PRODUCT AND SERVICE LINE

DECLARE/ DEMONSTRATE/ CERTIFY ESTABLISHED ENVIRONMENT

USE THE ENVIRONMENT TO PROVIDE THE PRODUCT / SERVICE

+ TO BE DISCUSSED AT THE NEXT CHART

INPUT

ACQUIRER

DEVELOP PERFORMANCE SPECIFICATIONS FOR THE PRODUCT/SERVICE

VERIFY SUPPLIER’S PERFORMANCE HISTORY AND ESTABLISHED ENVIRONMENT

DEVELOP SOW; INCLUDE: PERFORMANCE SPECIFICATIONS & THE VERIFIED ENVIRONMENT

DEVELOP SOW; INCLUDE: PERFORMANCE SPECIFICATIONS & 12207 AND OTHER TASKS

ONE OR BOTH

INPUT
A total environment or the standard is general, complex, and large.
- Outside of a project/service context, it would appear abstract.
- Using all of it would be neither cost-effective nor feasible.

Therefore, the environment or the standard should be tailored for the specific product/service. That is,
- Selecting the activities, tasks, inputs, and outputs
- Selecting methods, techniques, and tools for the above

The following charts will discuss the factors [determinants] that should be helpful in:
- Selecting the activities and tasks for the product or service
- Selecting methods, techniques, and tools for the above
  is left to down-the-road decision.

12207 will be used as a backdrop for selecting [tailoring] processes, activities, tasks, and outputs.
GETTING STARTED

1 IDENTIFY YOUR PRIMARY ROLE(s):
   - PROVIDING PRODUCTS & SERVICES
   - MANAGING PROCESSES

2 ORGANIZATIONAL MODE
   - MANAGE THE PROCESSES - INSTITUTIONALIZED ENVIRONMENT
   - IMPROVE THE PROCESSES

3 PRODUCT MODE SERVICE MODE
   DETERMINE WHICH 12207 PROCESSES TO EMPLOY
   DETERMINE WHICH OTHER PROCESSES TO EMPLOY
   SELECT ACTIVITIES & TASKS FROM THOSE PROCESSES
   TUNE, TAILOR THE INSTITUTIONALIZED ENVIRONMENT
   DEVELOP MODELS BASED ON THE PROCESSES, ACTIVITIES, AND TASKS
   COMPLETE/DELIVER THE PRODUCT OR SERVICE

NOTE USAGE:
- ONE PERSON (SOLO)
- WITHIN ORGANIZATION
- AMONG ORGANIZATIONS

ARE YOU THE ACQUIRER?
   Y
   CLARIFY TASKS THAT REFERENCE THE CONTRACT
   NEGOTIATE/SIGN CONTRACT WITH THE PARTIES
   PERFORM THE ACTIVITIES & TASKS RESPONSIBLE FOR

N

• THE SHADED BOXES WILL BE EXPLAINED IN THE ORDER THEY ARE NUMBERED.
APPLICATION STEPS AND FACTORS
[The shaded boxes 1, 2, 3 in Getting Started]

1. PRIMARY ROLES:
   Determine and identify

2. IN ORGANIZATIONAL MODE:
   2.1 Process Management: Establish the processes (and resources)
   2.2 Process Improvement: Improve the processes

3. IN PROJECT MODE:
   3.1 Application Concepts: Familiarize and understand
   3.2 Policies: Determine and identify applicables
   3.3 Project Characteristics: Determine and identify
   3.4 System Context: Select, determine, construct
   3.5 Life cycle Models: Determine and identify
   3.6 Specialty Areas: Identify and supplement
   3.7 Types of Software: Determine and identify outputs
   3.8 Documentation: Determine and identify
   3.9 Evaluation Categories: Determine and identify

• THE STEPS AND FACTORS ARE EXPLAINED NEXT AS OUTLINED ABOVE.
1. PRIMARY ROLES
DETERMINE AND IDENTIFY

- ACQUISITION ROLE
  - ACQUIRER
    - ACQUISITION PROCESS

- SUPPLY ROLE
  - SUPPLIER
    - SUPPLY PROCESS

- OPERATING ROLE
  - • OPERATOR
  - • USER
    - OPERATION PROCESS

- ENGINEERING ROLE
  - • DEVELOPER
  - • MAINTAINER
    - MAINTENANCE PROCESS

- SUPPORTING ROLE
  - EMPLOYER OF SUPPORTING PROCESSES
    - DEVELOPMENT PROCESS

- ORGANIZATIONAL ROLE
  - MANAGER
    - ORGANIZATIONAL PROCESSES
      - • Management
      - • Infrastructure
      - • Improvement
      - • Training

- ROLE?

- Supporting Processes
  - Documentation
  - Configuration management
  - Quality assurance
  - Verification
  - Validation
  - Joint review
  - Audit
  - Problem resolution
2.1 PROCESS MANAGEMENT
ESTABLISH THE PROCESSES

MANAGE A PROCESS?

IS THE PROCESS DEFINED IN 12207?

N

EXECUTE MANAGEMENT PROCESS
CUSTOMIZE/ SPECIALIZE/ INSTANTIATE THE PROCESS

EXECUTE INFRASTRUCTURE PROCESS
ESTABLISH/ INSTITUTIONALIZE THE PROCESS AS AN ENVIRONMENT

EXECUTE TRAINING PROCESS FOR PERSONNEL;
TAILORING PROCESS FOR PROJECTS

EXECUTE IMPROVEMENT PROCESS
SEE THE NEXT CHART

Y

MANAGE THE 12207 PROCESS
THIS PROCESS IS DEFINED IN 12207 AND SHOULD BE AN INSTANTIATION OF THE MANAGEMENT PROCESS
2.2 PROCESS IMPROVEMENT
IMPROVE THE PROCESSES

PROCESS IMPROVEMENT [Procedures ...]

PROCESS ASSESSMENT [Procedures ...]

ASSSESSMENT DATA
- Effectiveness
- Suitability
- Capability
- Cost of quality

PROCESS BASELINES

PROJECT 1
PROJECT n

EXPERIENCE DATA BASE
- Historical
- Technical
- Evaluation
- Quality cost data

ISO/IEC 12207 PROCESSES

PROCESS DEFINITIONS

PROCESS IMPROVEMENT DATA
- Process improvement
- Project's direction
- Technology advancement
- Organizational improvement

PROCESS ESTABLISHMENT [Environment, control ...]
3.1 APPLICATION CONCEPTS
FAMILIARIZE AND UNDERSTAND

• DUAL USE OF 12207:
  - To develop, operate, and maintain application products
  - To use to analyze, model, or study a system
    - Whether or not the system would contain software.

• LIFE CYCLE MODEL:
  - To organize and manage the steps/phases of a life cycle in the desired order(s)
  - Incorporates developmental, operational, maintenance, ... models

• PROTOTYPING IN 12207:
  - Not listed as an activity of the Development or Maintenance
  - Treated as a method/technique for:
    - Performing studies, requirements analysis, design, etc.
    - Developing prototypes and mock-ups

• BUILD:
  - An instance of a product that meets a specified subset
    of the total requirements.
  - A build may be a prototype
  - A period of time during which the build is developed.

• Iteration: Iteration across activities.
  Recursion: "Iteration" across tasks in an activity.
  Note: Not every activity (task) needs to be executed in every iteration (recursion).
3.2 POLICIES
DETERMINE AND IDENTIFY APPLICABLES

• CHECK POLICIES & STANDARDS AFFECTING TAILORING:
  - Contract type(s): Fixed price; cost plus fee; etc.
  - Operations and support strategies
  - Documentation/data/interface standards
  - Safety, security, risk management
  - Programming language(s)
  - Reserve requirements
  - Measurement/metrics
  - Reuse
  - Proprietary, usage, warranty, escrow rights
  - Use of IV&V agents
  - ...

• CHECK LAWS ON PUBLIC SAFETY, SECURITY, ENVIRONMENT, ...
  - Consider, interpret, and incorporate clauses related to the above items

• ADD CLAUSES IN AGREEMENT, IF NOT FOUND IN 12207
3.3 PROJECT CHARACTERISTICS
DETERMINE AND IDENTIFY

• SIZE: of software product, of software service, number of personnel

• CRITICALITY:
  - Impact of a defect/malfunction on business/mission
  - Liability due to failure
  - Immaturity or unprecedentedness of technology employed
  - Growth/changes in technology
  - Unprecedentedness of products under consideration
  - Unavailability of needed resources/schedule

• LIFE: Duration and extent of use and maintenance

• AS SIZE OR CRITICALITY INCREASES, SO DOES:
  - Extent of 12207's engineering tasks
  - Caution: Don't delete an engineering task unless sure
  - Extent of technical insight: [I]V&V
  - Extent of management visibility: Joint Review, Audit, QA

• AS OBJECTIVITY BECOMES MORE IMPORTANT, SO DOES:
  - Degree of independence of V&V Processes
  - Degree of organizational freedom for QA Process
  - Degree of independence in peer evaluations

• AS LIFE OR EXPECTED CHANGE INCREASES, SO DOES:
  - Extent of documentation
  - Caution: Consult with operators, users, and maintainers

• NOTE: Even for lesser size or criticality, independent or peer evaluations are beneficial
3.4 SYSTEM CONTEXT
DETERMINE AND IDENTIFY

• A system is moved from cradle to grave through steps/phases

• Software is one of many components in the system

• Software follows the system in each step/phase

• A project puts the steps/phases together:
  - In the sequence
  - In the recursions
  - In the iterations
  - To the desired extend
3.4.1 SYSTEM LIFE CYCLE
[GENERIC; STATIC VIEW]

- NEEDS DETERMINATION
- CONCEPT DEFINITION
- DEMONSTRATION
- DEVELOPMENT
- PRODUCTIONS; MANUFACTURING
- DEPLOYMENT; SALES
- OPERATIONS
- MAINTENANCE & SUPPORT
- RETIREMENT

- DEFINE/DETERMINE NEEDS
- DEFINE CONCEPTS & SOLUTIONS
- DEMONSTRATE FEASIBILITY OF IMPLEMENTING SOLUTIONS
- DEVELOP PRODUCIBLE PRODUCTS
- PRODUCE; MANUFACTURE
- DEPLOY AT SITES; SELL
- OPERATE; USE PRODUCTS
- MAINTAIN/SUPPORT PRODUCTS
- RETIRE PRODUCTS
3.4.2 VIEWS OF LIFE CYCLE

SYSTEM’s LIFE CYCLE

<table>
<thead>
<tr>
<th>PARTY</th>
<th>ND</th>
<th>CE</th>
<th>Dem</th>
<th>Dev</th>
<th>Prod</th>
<th>D/S</th>
<th>Opn</th>
<th>M,S&amp;R</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACQUISITION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUPPLY</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEVELOPMENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPERATION</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAINTENANCE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUPPORTING</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PARTY LIFE CYCLE

ND: NEEDS DETERMINATION
CD: CONCEPT DEFINITION
Dem: DEMONSTRATION
Dev: DEVELOPMENT
Prod: PRODUCTIONS
D/S: DEPLOYMENT/SALES
Opn: OPERATIONS
M,S&R: MAINTENANCE, SUPPORT & RETIREMENT

NOT TO SCALE
3.4.3 LIFE CYCLE DYNAMICS [MODELS]

LC DECISION POINTS:
- DO A NEXT PHASE
- HOLD PROJECT
- TERMINATE PROJECT
- CONTINUE CURRENT PHASE
- GO BACK TO A PREVIOUS PHASE

LC DECISION FACTORS:
- FUNDING & RESOURCES
- SCHEDULE
- FEASIBILITY
- TECHNOLOGY
- MARKET NEED
3.4.4 SYSTEM & SOFTWARE BUILDS

SYB = SYSTEM BUILD
SWB = SOFTWARE BUILD
3.4.5 12207 IN SYSTEM LIFE CYCLE

**NEEDS DETERMINATION**
- Use Acquisition Process to decide technical/operational/ economical feasibility.

**CONCEPT DEFINITION**
- Use [Acquisition, Supply &] Development Process to define draft system requirements, develop prototypes, and analyze user feedback to get proposed solutions.

**DEMONSTRATION**
- Use [Acquisition, Supply &] Development Process to define system requirements, system architecture, draft software requirements for software items.

**DEVELOPMENT**
- Use Acquisition & Supply Processes to trigger Development Process.
- Use Development Process [fully] to build, test, and integrate the product.

**PRODUCTIONS; MANUFACTURING**
- As no manufacturing is done for software, 12207 is of little use.

**DEPLOYMENT/SALES**
- Use [Acquisition, Supply &] Development Process to install and check-out products.

**OPERATIONS**
- Use [Acquisition, Supply &] Operation Process to provide operation services.

**MAINTENANCE; SUPPORT; RETIREMENT**
- Use [Acquisition, Supply &] Maintenance for maintenance, support, and retirement.
3.5 LIFE CYCLE MODELS
SELECT, DETERMINE, CONSTRUCT

- LIFE CYCLE MODELING IS A TOOL TO ORGANIZE/MANAGE THE STEPS/PHASES IN THE DESIRED ORDER
- A STEP/PHASE MAY HAVE ITS OWN MODELING
  - Examples: Development, Operation, Maintenance, ...

- ONLY [GENERIC] DEVELOPMENT MODELS WILL BE DISCUSSED NEXT.
- GENERIC OPERATIONS OR MAINTENANCE MODELS ARE NOT KNOWN.
### 3.5.1 BASIC DEVELOPMENT MODELS

<table>
<thead>
<tr>
<th>BASIC MODEL</th>
<th>ALL REQS. DEFINED FIRST?</th>
<th>MULTIPLE BUILDS?</th>
<th>USE INTERIM PRODUCTS?</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>WATERFALL</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
<td>BUILD N = BUILD 1</td>
</tr>
<tr>
<td>INCREMENTAL</td>
<td>YES</td>
<td>YES</td>
<td>MAYBE</td>
<td>BUILD N = BUILD (N-1) + MORE CAPABILITIES</td>
</tr>
<tr>
<td>EVOLUTIONARY</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>BUILD N = BUILD (N-1) + Refined specs.</td>
</tr>
</tbody>
</table>

- The basic models may be combined to create a hybrid model.
- Iterations and recursions are presumed.
## 3.5.2 BASIC DEVELOPMENT MODELS

### OPPORTUNITIES AND RISKS

<table>
<thead>
<tr>
<th>FACTORS</th>
<th>OPPORTUNITY</th>
<th>RISK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Requirements not well clear</td>
<td>E</td>
<td>W, I</td>
</tr>
<tr>
<td>2. System too large to do once</td>
<td>I, E</td>
<td>W</td>
</tr>
<tr>
<td>3. Full capability needed at once</td>
<td>W</td>
<td>I, E</td>
</tr>
<tr>
<td>4. Part capability needed early</td>
<td>I, E</td>
<td>W</td>
</tr>
<tr>
<td>5. Phase out of old system to be gradual</td>
<td>I, E</td>
<td>W</td>
</tr>
<tr>
<td>6. Rapid changes in requirements anticipated</td>
<td>E</td>
<td>W, I</td>
</tr>
<tr>
<td>7. Rapid changes in technologies anticipated</td>
<td>E</td>
<td>W, I</td>
</tr>
<tr>
<td>8. Long-run staff/funds commitment doubtful</td>
<td>W</td>
<td>I, E</td>
</tr>
</tbody>
</table>

W= WATERFALL; I= INCREMENTAL; E= EVOLUTIONARY

**NOTE:** A PROJECT MAY USE MORE THAN ONE MODEL.
3.5.4 DEVELOPMENT MODEL: INCREMENTAL

R: REQUIREMENTS  C/T: CODING & TESTING
D: DESIGN  I/AS: INSTALLATION & ACCEPTANCE SUPPORT
3.5.5 DEVELOPMENT MODEL: EVOLUTIONARY

--- REFINEMENTS ---

R: REQUIREMENTS  C/T: CODING & TESTING
D: DESIGN  I/AS: INSTALLATION & ACCEPTANCE SUPPORT
3.5.6 DEVELOPMENT MODEL: SPIRAL

- A: FEASIBILITY STUDY, PROTOTYPING, RISK ANALYSIS
- THIS MODEL ADOPTED FROM BOEHM's
- THE SECTORS MAY BE ALLOCATED TO THE 13 ACTIVITIES AS APPROPRIATE
- THIS MODEL IS REDUCIBLE TO THE BASIC MODELS
- MAY BE USED AS A HYBRID INCREMENTAL-EVOLUTIONARY MODEL
3.5.7 DEVELOPMENT MODEL: RE-ENGINEERING

GOAL (EXAMPLE):
TRANSLATE TO Ada (SI 1),
RE-DESIGN TO OOD (SI 2),
RETARGET TO NEW COMPUTER (SI 3).

Determine reqs. for reeng'd system
Analyze existing products; derive design; derive reqs.
Design reeng'd system
### 3.6 SPECIALTY AREAS

**IDENTIFY & SUPPLEMENT**

- **SPECIALTY-AREA STANDARDS SHOULD BE USED WITH 12207**
  - Examples of specialty standards: safety, security, ergonomics, documentation, ...
  - Correlate terminology and concepts
    - Examples: coding v. implementation; architecture v. top-level design; ...
  - Determine supplementary tasks from the specialty standards for each 12207 process
    - Example: formal methods for design and verification, ...
  - Add the supplementary tasks to the 12207 tasks in respective clauses

- **EXAMPLE: SAFETY IN THE DEVELOPMENT PROCESS OF 12207**

<table>
<thead>
<tr>
<th>TASK IN 12207</th>
<th>SUPPLEMENTARY TASKS FROM A SAFETY SPECIALTY STANDARD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop system architecture, ...</td>
<td>• Isolate safety specifications in separate configuration items.</td>
</tr>
<tr>
<td>Develop software architecture, ...</td>
<td>• Use structured design.</td>
</tr>
<tr>
<td></td>
<td>• Isolate safety specifications in separate components/modules.</td>
</tr>
<tr>
<td></td>
<td>• Use information hiding.</td>
</tr>
<tr>
<td></td>
<td>• Derive design mathematically.</td>
</tr>
<tr>
<td>Develop code, ...</td>
<td>• Derive code by mathematically.</td>
</tr>
<tr>
<td></td>
<td>• Use N-version coding resulting in voting modules.</td>
</tr>
<tr>
<td>Test units and aggregates, ...</td>
<td>• Exercise structural coverage [of branches and paths].</td>
</tr>
<tr>
<td>Document software requirements</td>
<td>• Document ... in Software Requirements Specification (SRS).</td>
</tr>
</tbody>
</table>
## 3.7 TYPES OF SOFTWARE
**DETERMINE AND IDENTIFY**

- DIFFERENT TYPES OF SOFTWARE NEED DIFFERENT TREATMENT
- A PROJECT MAY HAVE MORE THAN ONE TYPE OF SOFTWARE

<table>
<thead>
<tr>
<th>CLASSIFICATION SCHEME - I</th>
<th>CLASSIFICATION SCHEME - II</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>APPLICATION:</strong></td>
<td>NEW DEVELOPMENT</td>
</tr>
<tr>
<td>To monitor and control system functions</td>
<td></td>
</tr>
<tr>
<td>- ATC, fire control, ...</td>
<td></td>
</tr>
<tr>
<td><strong>SUPPORT:</strong></td>
<td>EMBEDDED</td>
</tr>
<tr>
<td>To support users</td>
<td></td>
</tr>
<tr>
<td>- Word processors, graphics, test generator, ...</td>
<td></td>
</tr>
<tr>
<td><strong>SYSTEM:</strong></td>
<td>INTEGRAL</td>
</tr>
<tr>
<td>To support computer operations</td>
<td>Connected within/to a system</td>
</tr>
<tr>
<td>- OS, Compiler, ...</td>
<td></td>
</tr>
</tbody>
</table>

|                       | STAND-ALONE                                  |
|                       | Self-contained (viz., payroll)               |
|                       | OFF-THE-SHELF                                |
|                       | Exists but not developed under the current project |
|                       | NON-DELIVERABLE                              |
|                       | Employed in the development/maintenance    |

- NO DIFFERENCES IN APPLICATIONS OF 12207 FOR CLASSIFICATION-I
- APPLICATION OF 12207 FOR CLASSIFICATION-II TO FOLLOW NEXT
3.7.1 TYPES OF SOFTWARE
OFF-THE-SHELF SOFTWARE

- OFF-THE-SHELF SOFTWARE (OTSS):
  - Software that is available from a source, but was not developed under the current application.
  - Source: In-house, another part of the organization, another organization, acquirer, market, …

- HELPFUL POINTS:
  - Consider the OTSS as a new software for the current application.
  - OTSS may or may not save costs.
  - OTSS may enhance or compromise design and performance.
  - Decide which activity the OTSS needs to enter.
  - Ensure the following, as applicable:
    - The OTSS satisfies its requirements & specifications
    - The documentation is available
    - Rights, warranty, licensing, and escrow are addressed
    - Future support for the OTSS is available.
3.7.2 TYPES OF SOFTWARE
NOTES ON REUSABILITY & PORTABILITY

- **REUSABILITY**: Extent of use in another application
- **PORTABILITY**: Extent of use in another computer system
- **REUSABILITY & PORTABILITY**:
  - Quality characteristics
  - Specified and designed
  - May have inherent conflict with other quality characteristics
  - Universal programming language ideal
- **A PROGRAM FROM ONE COMPUTER SYSTEM RARELY RUNS FIRST TIME ON ANOTHER COMPUTER SYSTEM.**
  - Few compilers adhere to standards exactly (have extra, improved features)
  - Word length varies from machine to machine
    - The largest integer varies. Different rounding off of numbers
  - Smallest positive floating point number depends on the machine
    - Drastic effect on "iterations until $x < e""
  - 1's complement machine may give different results from 2's complement machine
  - Instruction length depends on OS
  - The program may not fit in the machine's memory; ...

“Never ever quite the same; nor ever quite another.”
3.7.3  TYPES OF SOFTWARE NON-DELIVERABLE ITEMS

• **NON-DELIVERABLE ITEM (NDI):**
  - A hardware or a software item that is not provided with the current application software, but is employed in its development or maintenance
  - Source: In-house, another part of the organization, another organization, acquirer, market

• **HELPFUL POINTS:**
  - For the suppliers: Approve and control all NDIs.
  - For the acquirers:
    - Decide whether the future operation or maintenance of the application software will depend on the NDI
    - Decide whether to "acquire" the NDI
    - Address rights, warranty, license, and escrow.
### TYPES OF SOFTWARE

**ENTRY POINTS, etc.**

- ENTRY POINT (SUGGESTED) REFERS TO DEVELOPMENT PROCESS ACTIVITY
  - A PRIOR ENTRY POINT POSSIBLE
- TYPES TYPICALLY DETERMINED DURING REQUIREMENTS ANALYSIS AND DESIGN

<table>
<thead>
<tr>
<th>TYPE</th>
<th>ENTRY POINT</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW DEVELOPMENT</td>
<td>SYS. REQ. ANALYSIS</td>
<td>• Consider all activities.</td>
</tr>
<tr>
<td>EMPLOYING OFF-THE-SHELF AS IS</td>
<td>SW QUAL. TESTING</td>
<td>• Full Development Process may be excessive.</td>
</tr>
<tr>
<td>INCORPORATING OFF-THE-SHELF WITHOUT MODIFICATIONS</td>
<td>SW UNIT TESTING</td>
<td>• Evaluate documentation and data rights needs.</td>
</tr>
<tr>
<td>INCORPORATING OFF-THE-SHELF WITHOUT MODIFICATIONS</td>
<td>SW UNIT TESTING</td>
<td>• As in above</td>
</tr>
<tr>
<td>INCORPORATING OFF-THE-SHELF WITHOUT MODIFICATIONS</td>
<td>SW UNIT TESTING</td>
<td>• Based on performance record</td>
</tr>
<tr>
<td>ADAPTING/MODIFYING OFF-THE-SHELF</td>
<td>SW DESIGN</td>
<td>• Documentation may be unavailable/insufficient.</td>
</tr>
<tr>
<td>SOFTWARE OR FIRMWARE EMBEDDED IN OR INTEGRAL TO A SYSTEM</td>
<td>SYS. ARCH. DESIGN</td>
<td>• Decide if developer performs or supports system activities.</td>
</tr>
<tr>
<td>STAND-ALONE SOFTWARE</td>
<td>SW REQ. ANALYSIS</td>
<td>• Determine support life and documentation needs.</td>
</tr>
<tr>
<td>NON-DELIVERABLE ITEM</td>
<td>CLAUSE 5.3.1.5</td>
<td>• Beware of impact on the operations and maintenance.</td>
</tr>
</tbody>
</table>

**ENTRY POINT (SUGGESTED) REFERS TO DEVELOPMENT PROCESS ACTIVITY**

- A PRIOR ENTRY POINT POSSIBLE

**TYPES TYPICALLY DETERMINED DURING REQUIREMENTS ANALYSIS AND DESIGN**

- Beware of impact on the operations and maintenance.
3.8 DOCUMENTATION
DETERMINE AND IDENTIFY OUTPUTS

- REASONS FOR DOCUMENTATION (in 12207):
  - Use in or across the same activity or process
  - Examples: Development plan; design;
  - Examples: Acquisition requirements; User manuals
  - Use in other projects
  - Examples: From development to maintenance; Reuse
  - Delivery to the acquirer
  - Examples: Operations manual; code

- DETERMINE PRODUCTS/DOCUMENTATION:
  - Which ones? -- Specifications; Design; ...
  - See the next chart for 12207's output categories
  - Format, Content, Media, ... ?
  - Use the Documentation Process of 12207

- HELPFUL POINTS:
  - Existing product standards usable with 12207
  - Correlate 12207 outputs with your product standards
  - Involve affected persons: especially users, operators, maintainers
  - Life cycle cost more critical than development cost
  - Over 70% of life cycle cost in maintenance
## 3.8.1 OUTPUT CATEGORIES

<table>
<thead>
<tr>
<th>OUTPUT CATEGORY</th>
<th>12207 OUTPUTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLAN</td>
<td>ACQUISITION, PROJECT MANAGEMENT, DEVELOPMENT, TEST, INTEGRATION, INSTALLATION, OPERATION, MAINTENANCE, MIGRATION, RETIREMENT, CM, QA, VERIFICATION, VALIDATION, MANAGEMENT, INFRASTRUCTURE, TRAINING</td>
</tr>
<tr>
<td>REQUIREMENTS &amp; SPECIFICATIONS</td>
<td>SYSTEM, HARDWARE, SOFTWARE, MANUAL OPERATIONS, TEST, SYSTEM QUALIFICATION, SOFTWARE QUALIFICATION</td>
</tr>
<tr>
<td>DESIGN</td>
<td>SYSTEM ARCHITECTURAL, SOFTWARE ARCHITECTURAL, SOFTWARE DETAILED, INTERFACE, DATA BASE</td>
</tr>
<tr>
<td>SOFTWARE CODE &amp; DATABASE</td>
<td>SOFTWARE UNIT, DATABASE</td>
</tr>
<tr>
<td>TEST</td>
<td>TEST REQUIREMENTS &amp; TEST RESULTS FOR UNITS, DATA BASE, SYSTEM/SOFTWARE QUALIFICATION, ACCEPTANCE, MAINTENANCE</td>
</tr>
<tr>
<td>MANUAL (USER’S)</td>
<td>FOR OPERATIONS AND MAINTENANCE</td>
</tr>
<tr>
<td>EVALUATION RESULT/REPORT</td>
<td>INTERNAL, QA, V&amp;V, JOINT REVIEW, AUDIT, CM, PROBLEM RESOLUTION</td>
</tr>
<tr>
<td>SPECIAL</td>
<td>TAILORING DECISIONS</td>
</tr>
</tbody>
</table>

Outputs in a category may be combined into one or more sets. Example: Development plan includes test, integration, installation and CM.
### 3.9 EVALUATION CATEGORIES
**DETERMINE AND IDENTIFY**

<table>
<thead>
<tr>
<th>EVALUATION</th>
<th>OBJECTIVE</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROCESS INTERNAL</td>
<td>EXTENT TO WHICH MEETS SPECIFIED CRITERIA</td>
</tr>
<tr>
<td>VERIFICATION</td>
<td>VERIFY PRODUCTS IN AN ACTIVITY AGAINST PREVIOUS ACTIVITIES [DOING IT RIGHT]</td>
</tr>
<tr>
<td></td>
<td>- Depth depends on criticality</td>
</tr>
<tr>
<td></td>
<td>- Independence depends on objectivity</td>
</tr>
<tr>
<td>VALIDATION</td>
<td>VALIDATE AS-BUILT PRODUCTS FOR SPECIFIC INTENDED USE [DONE IT RIGHT]</td>
</tr>
<tr>
<td></td>
<td>- Depth depends on criticality</td>
</tr>
<tr>
<td></td>
<td>- Independence depends on objectivity</td>
</tr>
<tr>
<td>QUALITY ASSURANCE</td>
<td>ASSURE PRODUCTS &amp; PROCESSES CONFORM TO REQUIREMENTS AND ADHERE TO PLANS</td>
</tr>
<tr>
<td></td>
<td>- External</td>
</tr>
<tr>
<td></td>
<td>- Organizational freedom and authority</td>
</tr>
<tr>
<td>JOINT REVIEW</td>
<td>REVIEW OF STATUS &amp; PRODUCTS</td>
</tr>
<tr>
<td></td>
<td>- Inter-party [Typically acquirer-supplier]</td>
</tr>
<tr>
<td>AUDIT</td>
<td>COMPLIANCE WITH REQUIREMENTS, PLANS AND CONTRACT</td>
</tr>
<tr>
<td></td>
<td>- By authorized persons [acquirer or representative]</td>
</tr>
<tr>
<td>IMPROVEMENT</td>
<td>SELF-IMPROVEMENT OF PROCESS(es)</td>
</tr>
</tbody>
</table>
APPLICATION -- RECAP

• WE DISCUSSED:
  - Role determination in a life cycle
  - Factors and determinants for selecting (and supplementing) processes, activities, and tasks
    - At project level
    - At organizational level.

• SUITABLE FOR EITHER BUSINESS PRACTICE OPTION:
  - Direct use of 12207 (including with specialty standards)
  - Use of an established environment (with 12207 as backdrop)

• HOPEFULLY, ACCOMMODATES DIVERSE PROJECTS, ORGANIZATIONS, AND COUNTRIES.

• NEXT, SOME ADVICE WILL BE GIVEN.
ADVICE ON SPECIFICATION PRACTICES - I

• Remember:
  - Without specifications, there is no further work.
  - Specification mistakes are of commission and omission types.
  - A mistake in requirements/specifications that costs $1 to fix now would cost about $75 to fix during testing and more afterwards.

• Software specifications are begun during system design
  - There may be a “phase lag” between system and software.

• Software [requirements and] specifications will change
  - The challenge is in managing those changes.
  - Some changes can create an “earthquake” in software development.
  - Baselining [builds] is one way to control changes and impacts.
  - Incremental, evolutionary, and Spiral models may be helpful tools.
**ADVICE ON SPECIFICATION PRACTICES - II**

- **Make sure quality characteristics and lower characteristics are as completely specified as possible:**
  - Pay special attention to safety, security, and privacy.
  - Some quality characteristics may conflict with each other.
  - Prioritize the quality characteristics.
  - Caution: Software engineering may not be mature enough to design in several quality characteristics.

- **Besides what the system needs to do, specify the following:**
  - What the software must not do.
  - Execution behavior at non-normal inputs
  - Specifications of interfaces
  - Test cases and associated specifications, as needed
  - Constraints on design, interfaces, and test environment
  - Target environment, as much as possible

- **Use specification language, techniques, and tools**
QUALITY FACTORS [CHARACTERISTICS]

C: CONFLICTING;  S: SUPPORTING;  BLANK: NO RELATIONSHIP

Source: James McCall
SAFETY, SECURITY, RELIABILITY

SAFETY
PREVENTING THE SYSTEM FROM DAMAGING ITS EXTERNAL WORLD

SECURITY
PROTECTING THE SYSTEM FROM DAMAGE BY ITS EXTERNAL WORLD

RELIABILITY
ASSURANCE THAT THE SYSTEM WILL PERFORM AS INTENDED.

ENGINEERING EQUIPS THE SYSTEM WITH THE ABOVE FEATURES.
FACTORS AFFECTING WORKING HUMANS (EXAMPLES):
- ENVIRONMENTS -- NATURAL, ARTIFICIAL, WORK
- CAPABILITY: PHYSICAL AND MENTAL
- FELLOW WORKERS; EQUIPMENT; AUTOMATION
- LIGHTING, CONSOLE, SCREEN, DESK, CHAIR, TIME, FOOD, et.al.
ADVICE ON TESTING - I
CONTEXT

• A programmer describes a system function to a digital computer in “precise” instructions.

• What a programmer thinks s/he instructed the computer to do and what the computer understands it has been instructed to do are rarely the same.

• A program works in digital space, which is discrete and not finite.

• Completeness of testing:
  - It is impractical, if not impossible, to cover the digital space completely by testing.

• Adequacy of testing:
  - The magic is in covering the digital space adequately, but judiciously and cost-effectively.

• Do not depend on testing only to gain confidence. Use analysis, inspection, and demonstration as well.
ADVICE ON TESTING - II
ADEQUACY OF TESTING

• Modularize
  - Follow the rules and schema of modularization.
  - Ensure correct order and passing of parameters.

• For each module and aggregate, compare “line-by-line” outputs with manual calculations for certain input values:
  - Normal -- educated, representative points
  - Non-normal: Boundaries; Discontinuities; Singularities; +/- delta
  - Smallest and largest numbers allowed by the computer.

• Note: The above must be accommodated by design and in code.

• Execute with global initializations: Zero; Infinity.

• Cover McCabe’s basic paths.

• Compare runs with special/deduced/known cases.

• Remember: A modification may introduce new errors.

• When a module is modified, ensure other modules are not affected.

• Employ a suitable reliability growth model to determine releasability.

• Include information on the computer, OS, language, and compiler.
TOPICS

1. BACKGROUND
2. BASIC CONCEPTS
3. THE PROCESSES
4. APPLICATION

5. RELATED AREAS

6. SUMMARY

7. FOR YOUR INFORMATION
# REFERENCED STANDARDS

<table>
<thead>
<tr>
<th>REFERENCED STANDARD</th>
<th>REFERENCED AS</th>
<th>REASON FOR REFERENCING</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFNOR: Dictionary of Computer Science</td>
<td>Normative</td>
<td>Definitions</td>
</tr>
<tr>
<td>ISO/IEC 2382-1</td>
<td>Normative</td>
<td>Definitions</td>
</tr>
<tr>
<td>ISO/IEC 2382-20</td>
<td>Normative</td>
<td>Definitions</td>
</tr>
<tr>
<td>ISO 8402</td>
<td>Normative</td>
<td>Definitions</td>
</tr>
<tr>
<td>ISO 9001</td>
<td>Normative</td>
<td>Additional guidance on quality systems</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[see clause 6.3]</td>
</tr>
<tr>
<td>ISO/IEC 9126</td>
<td>Normative</td>
<td>Guidance on quality characteristics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[see clause 5.3.4]</td>
</tr>
<tr>
<td>ISO/IEC 12119</td>
<td>Bibliography</td>
<td>Software packages</td>
</tr>
</tbody>
</table>
### SUPPLEMENTARY STANDARDS

<table>
<thead>
<tr>
<th>STANDARD</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Technical Reports (TR):</strong>&lt;br&gt;- ISO/IEC 15271, 12207 Guidebook</td>
<td>- Under publication</td>
</tr>
<tr>
<td>- ISO/IEC 14759, Mockup &amp; Prototype Guide</td>
<td>- TR Ballot Passes</td>
</tr>
<tr>
<td><strong>Standard:</strong>&lt;br&gt;ISO/IEC 14764, Software Maintenance</td>
<td>Under FDIS BALLOT</td>
</tr>
<tr>
<td><strong>Standard:</strong>&lt;br&gt;ISO/IEC 15846, Software CM</td>
<td>Published</td>
</tr>
<tr>
<td><strong>Guide:</strong>&lt;br&gt;ISO/IEC 16326 Software Project Management</td>
<td>CD</td>
</tr>
<tr>
<td><strong>Guide:</strong>&lt;br&gt;Software QA</td>
<td>Canceled in favor of ISO 9000-3</td>
</tr>
<tr>
<td><strong>Guide:</strong>&lt;br&gt;Software V&amp;V</td>
<td>On hold</td>
</tr>
<tr>
<td><strong>Guide:</strong>&lt;br&gt;Reviews &amp; Audits</td>
<td>On hold</td>
</tr>
</tbody>
</table>
RELATIONSHIP TO SYSTEM AREAS

SYSTEMS ENGINEERING

CONVENTIONAL ENGINEERING

SOFTWARE ENGINEERING

INFORMATION TECHNOLOGY

SOFTWARE ENGINEERING

NOT TO SCALE; NOT EXHAUSTIVE
RELATIONSHIP TO QUALITY AREAS

QUALITY

SYSTEM: ISO 9001; 8402
SOFTWARE: ISO 9000-3

QUALITY SYSTEM REQUIREMENTS

ASSESSMENT

SYSTEM: ?
SOFTWARE: ISO/IEC 15504

QUALITY SYSTEM REQUIREMENTS

REQUIREMENT TO MANAGE PROCESSES

SYSTEM: ISO/IEC 15288
SOFTWARE: ISO/IEC 12207

QUALITY SYSTEM REQUIREMENTS

REQUIREMENT TO ASSESS AND IMPROVE PROCESSES

LIFE CYCLE PRACTICES
COMPARISON WITH ISO 9001

ISO 9001 ← ISO 8402 ← ISO/IEC 12207

CORPORATE VIEW OF QUALITY
QUALITY SYSTEM -- CONSOLIDATED
GENERAL PROCESS COMPLIANCE
SUPPLIER QUALITY SYSTEM CAPABILITY

FUNCTIONAL VIEW OF LIFE CYCLE
QUALITY FUNCTIONS -- DELEGATED
SPECIFIC PROCESS COMPLIANCE
PRODUCT DEVELOPMENT; SERVICE

12207 INVOKES 9001 FOR FURTHER QUALITY SYSTEM
TOPICS

1. BACKGROUND
2. BASIC CONCEPTS
3. THE PROCESSES
4. APPLICATION
5. RELATED AREAS

6. SUMMARY

7. FOR YOUR INFORMATION
SUMMARY

• **12207 IS THE TOP-LEVEL ARCHITECTURE OF SOFTWARE LIFE CYCLE**
  - Architecture built with processes
  - Processes have tasks and outcomes.

• **PROVIDES A COMMON FRAMEWORK FOR:**
  - Acquiring & supplying products & services
  - Managing & improving the processes
  - *World trade in software*
EPILOGUE

• 12207 IS NOT A SUBSTITUTE FOR DISCIPLINED MANAGEMENT AND ENGINEERING

• 12207 PROVIDES MERELY THE BUILDING BLOCKS FOR CONSTRUCTING MODELS, STRATEGIES, AND PLANS FOR PROJECTS AND ORGANIZATIONS
  - You need to proceduralize and automate the building blocks using your organizational knowledge and experience for that extra quality, competitiveness, and success

• 12207 SHOULD BE USED BY TRAINED PERSONNEL

• PLEASE READ THE STANDARD IN YOUR SPECIFIC CONTEXT: LEARNING, PROJECT, ORGANIZATION, ...
  - Otherwise, it may be misinterpreted
TOPICS

1. BACKGROUND
2. BASIC CONCEPTS
3. THE PROCESSES
4. APPLICATION
5. RELATED AREAS
6. SUMMARY

7. FOR YOUR INFORMATION
COST AND BENEFIT OF USING 12207

• **ASSUMPTION:** Basic software engineering environment instituted

• **COST/TIME OF IMPLEMENTING 12207 IN ORGANIZATION**
  - Unknown, but one-time cost
  - Factors:
    - Size/extent of software engineering environment
    - Size/diversity of the organization

• **COST OF 12207 TO A PROJECT**
  - Unknown, but savings through maintenance significant
  - Project-to-project cost should be lower
  - Factors:
    - Size/complexity of the project
    - Size of the personnel
    - Schedule and quality
    - Maturity levels of the acquirer and supplier

• **BENEFITS:**
  - The discipline
  - Reduced risk to cost, schedule and performance
ADAPTATION OF 12207 IN COUNTRIES

- Most participating countries have issued their National versions
- The USA developed its version in 3 parts -- under ANSI’s sponsorship:

ISO/IEC 12207

IEEE/EIA 12207.0 (STANDARD)
- ISO/IEC 12207, AS IS
- A FEW CHANGES
- ADDITIONAL ANNEXES:
  - BASIC CONCEPTS
  - COMPLIANCE
  - ERRATA

IEEE/EIA 12207.1 (GUIDE)
- LIFE CYCLE DATA

IEEE/EIA 12207.2 (GUIDE)
- IMPLEMENTATION GUIDELINES
COPIES OF 12207

• **ISO:**
  1, rue de Varembe
  CH-1211 Geneve 20
  Switzerland (Suisse)

• **IEC:**
  3, rue de Varembe
  CH-1211 Geneve 20
  Switzerland (Suisse)

• **ANSI:**
  American National Standards Institute
  Customer Services
  11 West 42nd St,
  New York, NY 10036
  USA
  [Tel: +1 212 642 4900; Fax: +1 212 302 1286]
ISO/IEC 15288
System Life Cycle Processes

• BACKGROUND:
  - Jun 94: SC7 study group on software-system relationship
  - Feb 95: Study group report to SC7 (Doc # SC7 N1331)
  - Mar 95: US ANSI New Work Item proposal to SC7
  - Jun 95: SC7 acceptance of the proposal
  - Apr 96: JTC1 approval of the project
  - May 96: Work started
  - Dec 00: Completion

• STUDY REPORT:
  - The software-system relationship is poor; needs immediate attention
  - Software is always part of system
  - Hardware & software are inherently different
  - Hardware & software are marching almost separately
  - Hardware terms are often unrealistic in software
  - 12207 has limited effectiveness without a system context
  - Recommended a standard on life cycle processes for
    modern systems containing hardware, software, and humans

• A CHALLENGE:
  Analog, digital, and manual functions together!
THANK YOU for LISTENING

PLEASE PROVIDE COMMENTS/SUGGESTIONS TO:

RAGHU SINGH
FEDERAL AVIATION ADMINISTRATION
AIR-200, ROOM-815
800 INDEPENDENCE AVE, S.W.
WASHINGTON, DC 20591
U.S.A.

Phone: +1 202 267-3976; Fax: +1 202 267-5580
E-Mail: RAGHUBANSH.SINGH@FAA.DOT.GOV
BACKUPS
COMING TO TERMS - I

- **ARCHITECTURE:**
  - The art and science of planning and building structures
  - A unifying or coherent form or structure

- **ASSESS:** Fix the value, size, importance of ...

- **AUDIT:** Check the authenticity of ... against a baseline, especially officially

- **EVALUATE:** Determine the worth, amount, value, or condition of ...

- **COMPATIBILITY:** Usable together in harmony

- **COMPLY** [with]: follow/obey [rules/laws]
  - The builder’s construction practices comply with the state’s building code.

- **CONFORM** [to]: Be in accordance with [specs.]
  - The house conforms to the official specs. of the building code.

- **CONSISTENCY:** In agreement with; does not contradict.

- **CRITERION:** A "standard" on which judgment is based

- **DESIGN:**
  - The arrangement of elements or details in a product or work
  - An underlying schema that governs functioning, developing, or unfolding

- **ENVIRONMENT:** An organization of manual and automated methods, techniques, and tools and personnel employed to produce products and provide services.

- **EXAMINE:** Test by questioning.
FORM, FIT & FUNCTION (3F):
- Form: physical configuration (shape) for interchangeability and compatibility
- Fit: I/O characteristics for interoperability
- Function: performance characteristics for “capable of doing its job.”

FRAMEWORK:
- A skeletal structure to hold or support something constructed or stretched over it
- Work done in a frame.

INSPECT: Check critically

PROTOTYPE: A [primitive] model that exhibits essential operational features ...

REQUIREMENT:
- Something obligatory, demanded as a condition
- Something needed

REVIEW: A general survey of ...

SYSTEMATIC: Definite scheme/method of procedure/classification

VALIDATE:
- Confirm the validity of ...
- Declare legally valid

VERIFY:
- Check the correctness of ... by comparison
- Prove to be true.
INDEPENDENCE & ORGANIZATIONAL FREEDOM

I: INDEPENDENCE:
- FROM THOSE PERFORMING THE TASK OR DEVELOPING THE PRODUCT

O: ORGANIZATIONAL FREEDOM:
- FROM THOSE WHO HAVE MANAGEMENT RESPONSIBILITY FOR THE TASK/PRODUCT
“FAULTY” TERMS

Error: Difference between computed and observed
Fault: Incorrect step/process/data definition ...
Failure: Incorrect result
Mistake: Action that produces an incorrect result

Cause of Error (a result, a manifestation):
- Fault in the process or the product
- Mistake by the system or human
“4.1.2  A standard does not in itself impose any obligation upon anyone to follow it. However, such an obligation may be imposed, for example, by legislation or by a contract. In order to be able to claim compliance with a standard, the user needs to be able to identify the requirements he is obliged to satisfy. He needs also to be able to distinguish these requirements from other provisions where he has a certain freedom of choice.

Clear rules for the use of verbal forms (including modal auxiliaries) are therefore essential.

Annex C gives, in the first column of each table, the verbal form that shall be used to express each kind of provision. The equivalent expressions given in the second column shall be used only in exceptional cases when the form given in the first column cannot be used for linguistic reasons.”

[See the next chart for specific verbs.]
### Verbal Forms - II


| REQUIREMENT
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VERB</strong></td>
</tr>
<tr>
<td>shall</td>
</tr>
<tr>
<td>shall not</td>
</tr>
<tr>
<td>must</td>
</tr>
</tbody>
</table>

| RECOMMENDATION
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VERB</strong></td>
</tr>
<tr>
<td>should</td>
</tr>
<tr>
<td>should not</td>
</tr>
</tbody>
</table>

| POSSIBILITY
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VERB</strong></td>
</tr>
<tr>
<td>can</td>
</tr>
<tr>
<td>cannot</td>
</tr>
</tbody>
</table>

| PERMISSION
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>VERB</strong></td>
</tr>
<tr>
<td>may</td>
</tr>
<tr>
<td>need not</td>
</tr>
</tbody>
</table>

- 12207 defines and uses “will,” which is not yet defined by ISO/IEC Directives.
- “Will” in 12207 means self-declaration and requirement.
12207 & MIL-STD-498

12207

ACQUISITION
SUPPLY
OPERATION
MAINTENANCE
DOCUMENTATION
VERIFICATION
VALIDATION
MANAGEMENT
INFRASTRUCTURE
TRAINING
IMPROVEMENT

498

DEVELOPMENT
QUALITY ASSURANCE
CONFIGURATION MANAGEMENT
PROBLEM RESOLUTION
TAILORING

JOINT REVIEW
AUDIT
DIDs
CLAUSES:
SAFETY
SECURITY
PRIVACY
REUSE
METRICS
ACCESS
12207 & MIL-STD-498 DIDs

- 498 is a development & documentation standard from acquisition perspective.
- 498 is a family of standards, under DOD Directives and MIL-STD-499B (Systems).
- 12207 addresses the life cycle and stakeholders.

<table>
<thead>
<tr>
<th>12207 DEVELOPMENT DOCUMENTATION</th>
<th>DIDs of MIL-STD-498</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acquisition process (5.1.1.1, 5.1.1.8)</td>
<td>OCD; PMP (Gov’t)</td>
</tr>
<tr>
<td>Supply process (5.2.4.5)</td>
<td>SDP; CM; QA; [SEMP/499B]</td>
</tr>
<tr>
<td>Plans for the dev. [versions in CM] (5.3.1.4)</td>
<td>SDP [Includes CM, QA, ...]; SVD</td>
</tr>
<tr>
<td>System requirements specification (5.3.2.1)</td>
<td>SSS</td>
</tr>
<tr>
<td>System architecture (5.3.3.1)</td>
<td>SSDD</td>
</tr>
<tr>
<td>Software requirements/specs. (5.3.4.1)</td>
<td>SRS; IRS</td>
</tr>
<tr>
<td>Architecture of software item (5.3.5.1)</td>
<td>SDD; SPS</td>
</tr>
<tr>
<td>Detailed design (5.3.6.1)</td>
<td>SDD; IDD; SPS</td>
</tr>
<tr>
<td>Database design (5.3.6.3)</td>
<td>DBDD</td>
</tr>
<tr>
<td>Software units and databases (5.3.7.1)</td>
<td>SPS; DBDD</td>
</tr>
<tr>
<td>Test reqs. (5.3.5.5, 5.3.6.5, 5.3.7.4, 5.3.8.1)</td>
<td>STP; STD; STR</td>
</tr>
<tr>
<td>Installation plan (5.3.12.1)</td>
<td>SIP</td>
</tr>
<tr>
<td>..</td>
<td>STrP (Transition)</td>
</tr>
<tr>
<td>User’s manuals (5.3.5.4, ..., 5.3.9.2)</td>
<td>SUM; SIOM; SCOM; COM; CPM; FSM</td>
</tr>
<tr>
<td>Evaluation records (5.3.2.2, ..., 5.3.11.2)</td>
<td>Internal records</td>
</tr>
</tbody>
</table>
• Tailor at two levels:

  1. Adaptation at organization level
     - Environment with methods, tools & techniques
  2. Tailoring the “adaptation” for each project

• Involve potential contractors and the post-development personnel

• Avoid fixed-price contract for software
12207: ADAPTATION TIPS

• ADAPT ISO/IEC 12207
  - Begin with IEEE/EIA 12207 -- or NATO AQAP 150
  - Incorporate changes and additions for your business sector
  - Institutionalize the Acquisition process (12207/5.1)
    - Ensure all tasks are supportable with personnel and procedures

• INVOKE SPECIALTY “STANDARDS”
  - Invoke RTCA DO-178B for the software requirements that impact system safety.
  - Similarly for security and human factors

• DEVELOP “DOCUMENTATION STANDARDS”:
  - Similar to US DoD’s DIDs
    - Demilitarize (use Jt-Std 016’s)
    - No prescription on format, media, etc.
  - To manage software development
  - For post-development personnel (operations, maintenance, et. al.)
12207: TAILORING TIPS

• DEVELOP YOUR OWN TAILORING GUIDANCE:

- Similar to MIL-HDBK-287 (Tailoring guide for 2167A)
- Base on tailoring criteria in 12207’s Annexes A and B
- Automate tailoring like 2167A Tailor

- Use a combination of the following:
  - The contractor has implemented a 12207 environment:
    - Develop guidance on assessing the environment
  - 12207 is directly used in contract:
    - Develop tailoring guidance for types of projects

- Do not compromise your project for off-the-shelf software
  - Ensure clause 5.1.1.7 is FULLY satisfied

- Avoid non-developmental item trap; Address clause 5.3.1.5.

- Consider evolutionary models -- with baselines and builds
WHAT IS A STANDARD?

• A COMMON PRODUCT
  - Identified as a preferred item in a situation

• A WRITTEN SET OF REQUIREMENTS FOR PRODUCTS (3F):
  - Form:
    - Physical configuration for interchangeability/compatibility
  - Fit:
    - Dimensional description of i/o for interoperability
  - Function:
    - Performance characteristics for job

• AN ACCEPTED PROCESS OR PROCEDURE
  - A series of actions or operations

• EXCEPTIONS:
  - Natural phenomena
  - Laws or regulations mandated by governments
    - Health, environment, safety, security, ...
## CONFORMITY & CERTIFICATION

### SOFTWARE PRODUCT & PROCESS

<table>
<thead>
<tr>
<th>STANDARDIZATION</th>
<th>ADAPTATION TAILORING</th>
<th>CONFORMANCE COMPLIANCE</th>
<th>CERTIFICATION</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PRODUCT</strong></td>
<td>3F specs.</td>
<td>Applicable 3F specs.</td>
<td>Meets the applicable 3F specs.</td>
</tr>
</tbody>
</table>
| **PROCESS**      | Activities/tasks for the services in the life cycle | Applicable activities & tasks | Satisfies the applicable activities & tasks | ? - Activities/tasks performed as planned
|                  |                      |                         | ? - Provides the products/services as specified |

3F: Form, Fit, Function

CONFORMITY SHOULD BE ADDRESSED SEPARATELY FOR PRODUCTS AND PROCESSES -- SO SHOULD BE CERTIFICATION
HUMOR IN CODE

Once God decided to find out what the Earthlings were up to. S/he descends upon the Earth. Finds a little girl crying over a broken doll, fixes it, and consoles her. Moves on. So s/he helps other people.

Just before departing, God sees a haggard man brooding over a piece of paper. S/he asked who he was and what was the problem. The man said he was a programmer, but asked whether s/he had any programming experience. S/he humbly admitted to have programmed things from Big Bangs to Black Holes. Impressed, the man showed his paper.

God reviewed the paper and, after a pause, said, "Son, I think this one is yours."

Anonymous
Q. How to have a standard powed?
A. Utter one of the following:

• **Prescriptive**
  "It's prescriptive."

• **OOD (and Ada)**
  "I can't use OOD (and Ada) with it."

• **Waterfall**
  "It's a Waterfall model."

• **Expensive**
  "It's expensive to use."

• **Documentation**
  "It's documentation dependent."