Standardization training program
E-40 discipline: Software Engineering

08/06/2021

A collaboration between

http://www.en.intecs.it/page/corsi

Your speaker

Christophe Honvault

christophe.honvault@esa.int

- Software engineer
  - 20 years in industry:
    - 5 years experience in Ground Software
    - 13 years experience in Space Embedded Software
  - 10 years at ESA, ESTEC (Noordwijk)
    - 5 years as Software engineer (Flight Software)
    - 3 years as head of Software System Engineering section
    - 2 years as head of Software Technology section
- Responsible for the On-Board Software Harmonisation Dossier.
- Co-Lead of the Avionics Competence Domain
- Coordinator of ECSS-E-ST-40C standard, chair of ECSS-E-ST-40C Rev1 Working group
- Main research topics at ESA/ESTEC:
  - Avionics, FDIR, Artificial Intelligence, Big Data, Cybersecurity, Formal Methods, Quantum computing, Requirement Engineering, System software co-engineering, Model-Based Software Engineering, Software Architecture, Avionics product lines

Why a Software Standard?

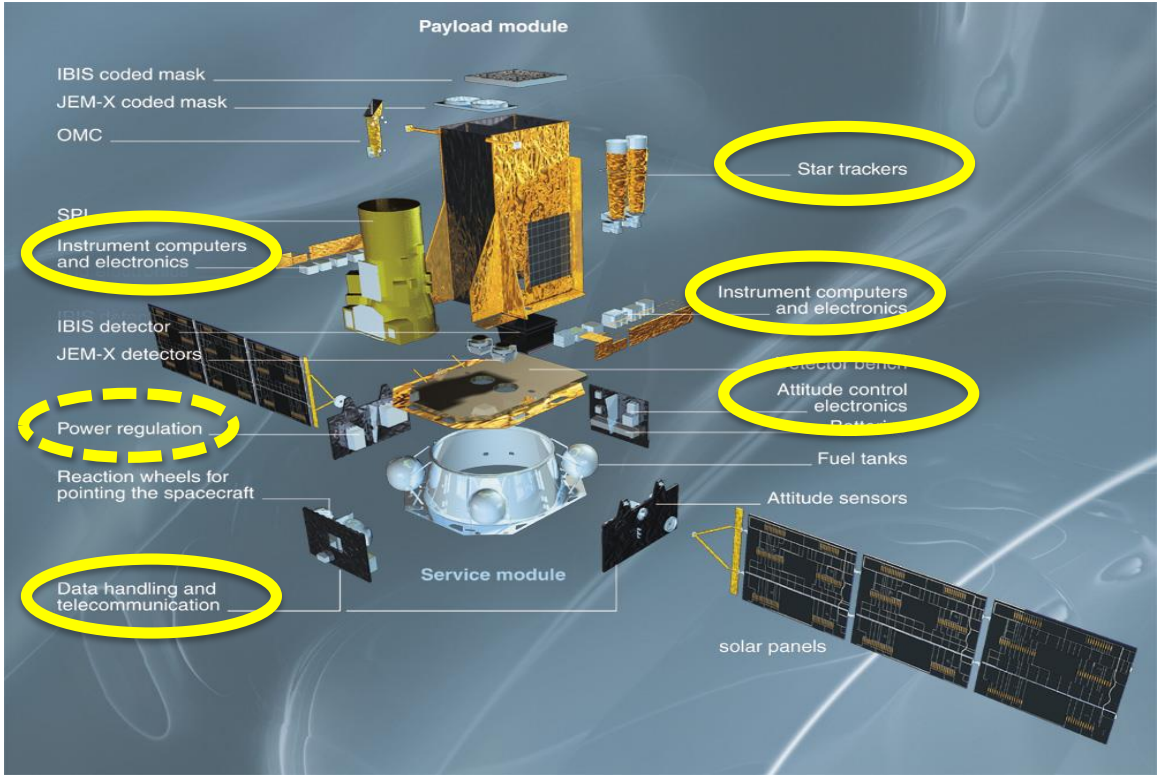
1. To support the business agreement between Customer and Supplier
    - Get a complete view of the software from management standpoint
    - Clarify developments activities
    - Focus the effort
    - Verify the completeness of the Statement of Work
2. The standard is completed with a Statement of Work that adds:
    - Delivery modalities
    - Software Development Environment (SDE)
    - Warranty
    - Intellectual Property Rights (IPR)
    - Customer Furnished Items (CFI)
    - etc.
3. For maturity reason
    - We want successful developments to be reproducible, not successful by coincidence (Spice level 2 of maturity ➔ standards)

PART 1:
Role of Software in the System

Abstract

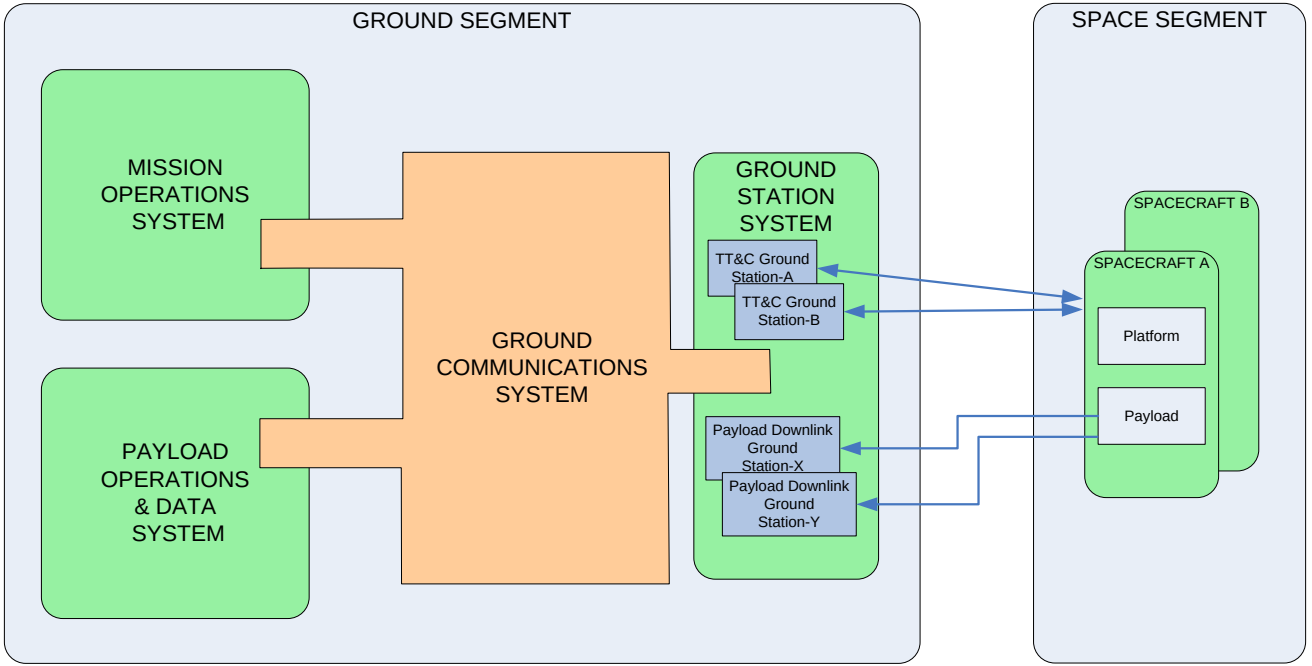
Just as software is one element in the overall engineering system, the ECSS-E-ST-40C standard for space software is one standard within the overall engineering branch of standards.
This module explains the relationship between ECSS-E-ST-40 and other ECSS standards.

Space software environments (I)



Space software environments (II)

Ground and space segment



Importance of software in the system

1. Software implements (more and more of) the **system behaviour**
2. System **complexity** increases ➔ software size increases
3. **Software schedule** is squeezed within the system schedule
4. Software is the last **flexibility** of the system at the end of the life cycle *(but reconfigurable FPGAs are coming)*
5. Software is a candidate for **subcontracting** policies
6. Software touches many parts of the system. It has **interface** everywhere (ground – hardware – avionics – payloads – sensors – actuators – EGSE – security)
7. Software uses a **lot of data** from various system functional chains (centre of gravity, temperature, health status, voltage)
8. Software has several **users** (system – AIT – operation)

IMPORTANCE OF:

specifying **requirements** (and **interface**)

validating software

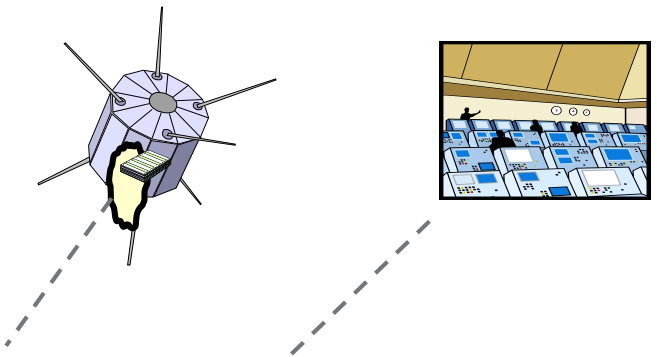
agreeing on a **development approach**

managing the **configuration**

Software and Space System Engineering

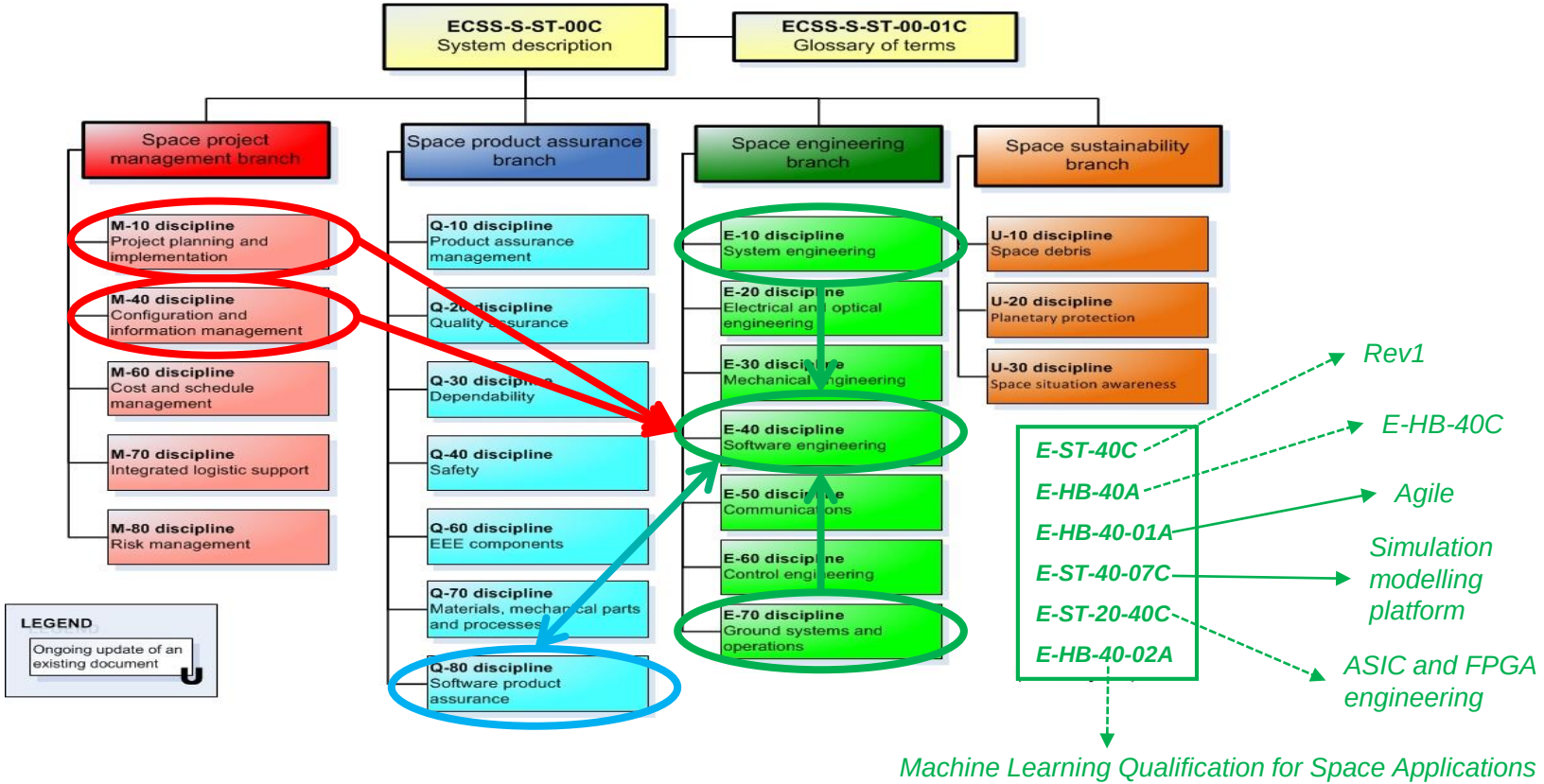
1. The software components of a space system play a role alongside the other engineering components such as mechanical and electrical

2. All of these various engineering components (including software) are governed by the overall discipline known as **space system engineering**



Software components are part of the overall mission system, together with other engineering components

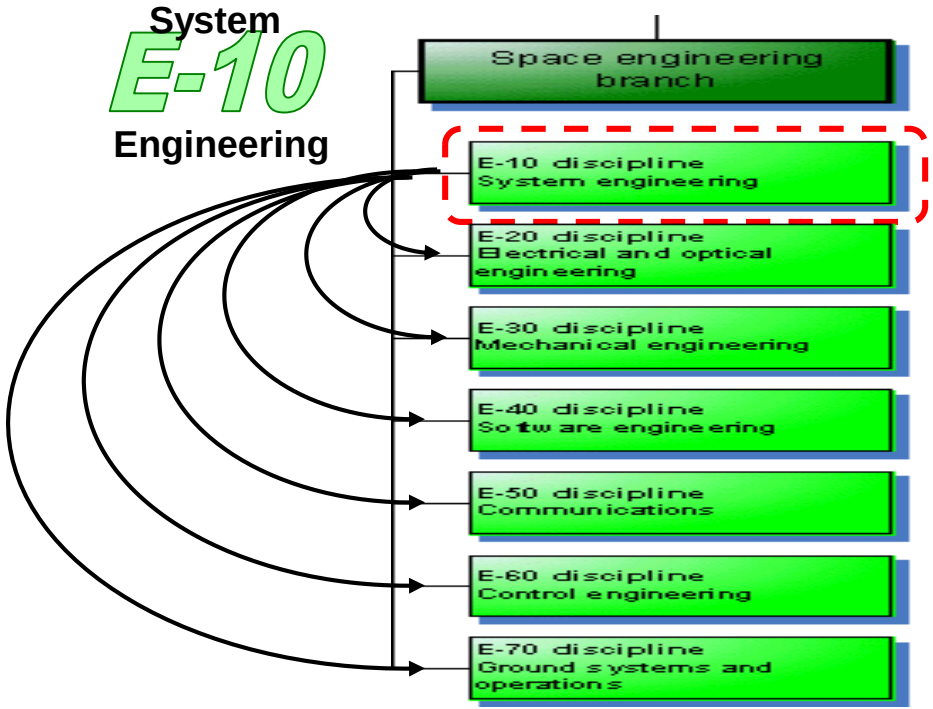
Software in the ECSS System



Software and system, E40 and E10

The E-10 Standard for System Engineering

1. The **ECSS-E-10** standard is special in that it is relevant to **all** the engineering disciplines, **including software**
    a. It is intended to guide the development of systems including H/W, S/W, man-in-the-loop, facilities & services for space applications
2. It specifies implementation requirements for the responsible **system engineering organization**



The Five System Engineering Functions

1. **Requirement engineering**
   - Translates customer needs to input for design
2. **Analysis**
   - Supports all other activities with various modeling, simulation, test activities
3. **Design and configuration**
   - creates the physical architecture
4. **Verification**
   - Checks compliance with requirements
5. **Integration and control**
   - Overall management of the activities

It is important to be aware how the overall system engineering process is organized

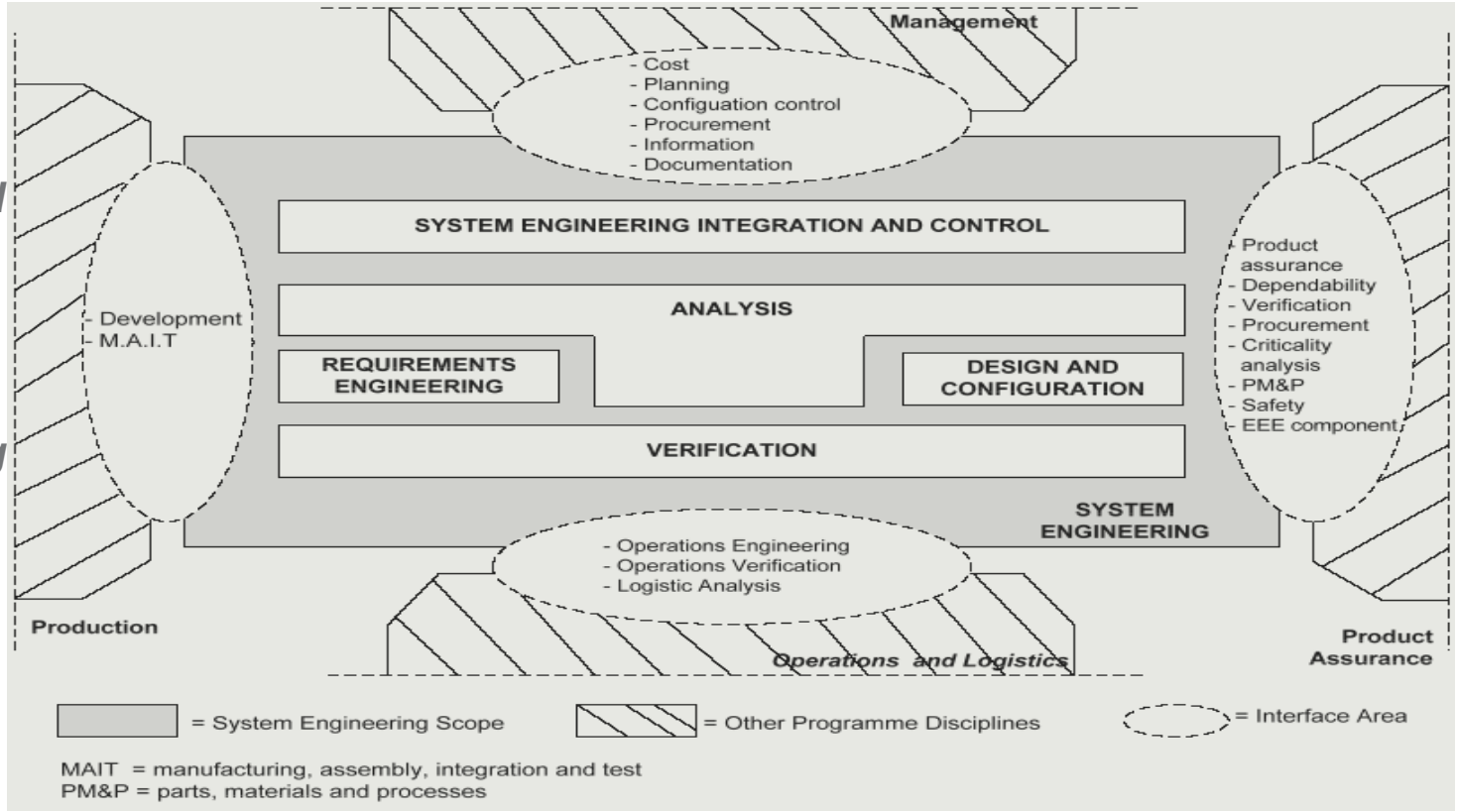
Although the E-40 standard defines its own processes, they "echo" this overall organisation and terminology

E-10

4.1

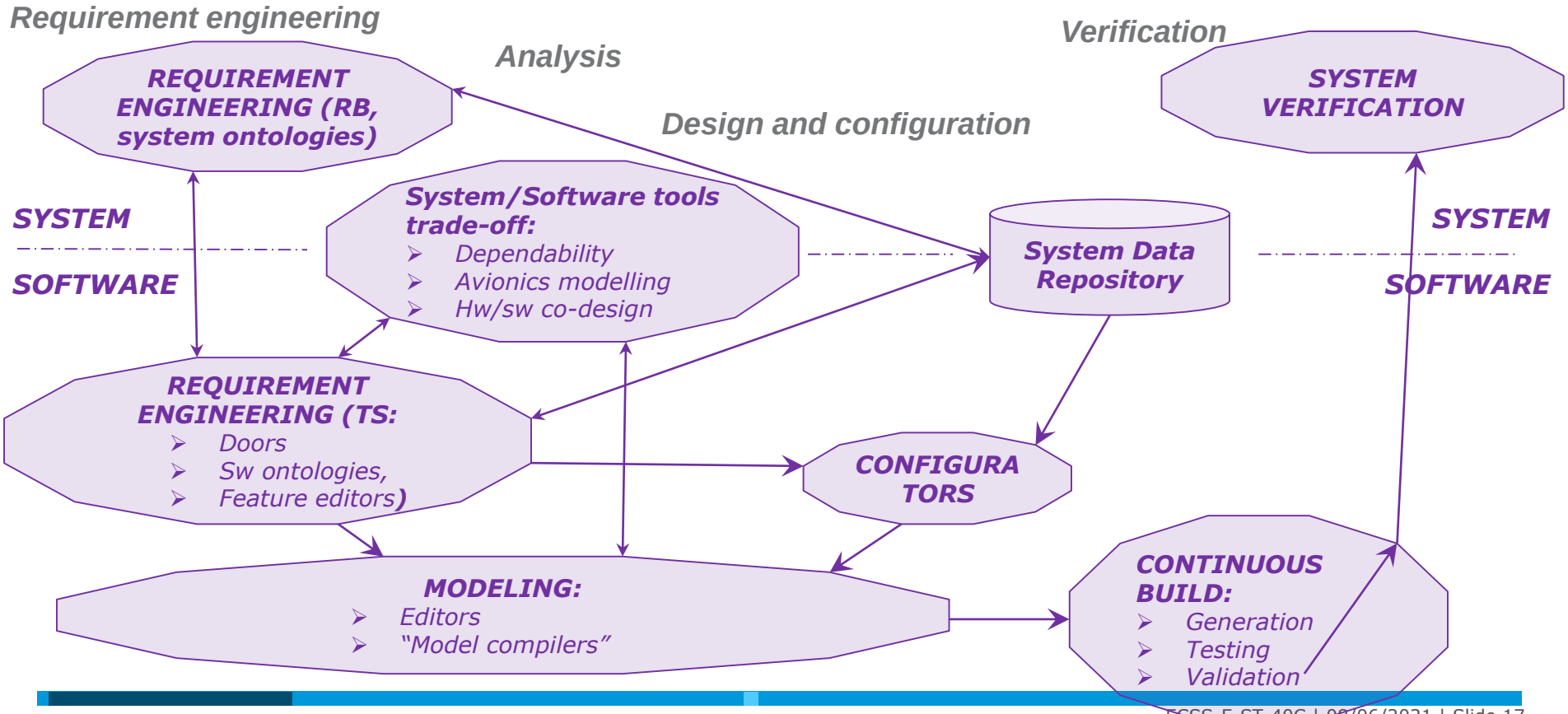
Overview of the System Engineering

A simplified view in which the five main system engineering functions are identified

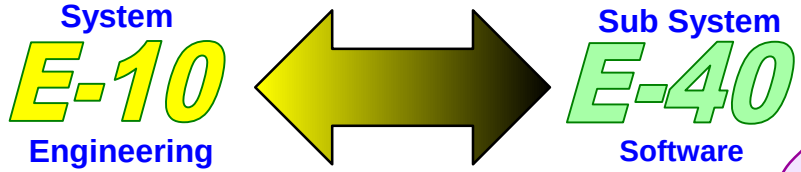


E-10

System - Software relationship



The Link Between E-10 and E-40



Space System Engineering

1. ***Requirements engineering***
2. ***Analysis***
3. ***Design and configuration***
4. ***Verification***
5. ***Integration and control***

Software related system requirement process (E-40 Section 5.2)

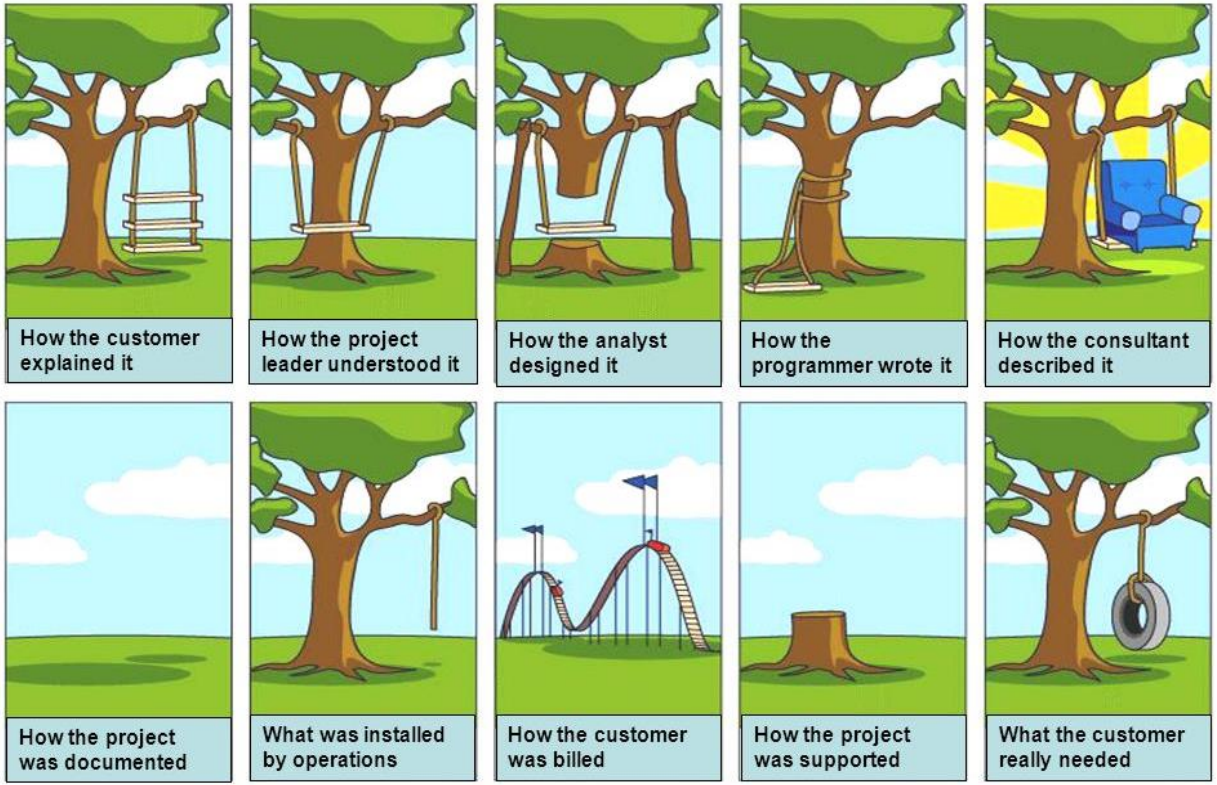
Space Software Engineering

Software related:

1. ***Requirements analysis***
2. ***Verification***
3. ***Integration and control***

This clause (5.2) of E-40 complements ECSS-E-10 for the specific software activities to be performed at system level by the customer

Importance of good requirements



System software : THE projects' critical issue…

1. System requirements related to software are normally done by the system entity (customer)
2. Software requirements are normally done by the software entity (supplier)
3. However, system requirements related to software may be:
    a. Delegated by the customer to the supplier.
        The customer may have initiated a software RB and ask for consolidation.
        The system requirements may be distributed in many (hardware) subsystem requirements.
    b. Merged with the software requirements

    The system is "software intensive", no value added in 2 documents, however incremental approach.

System software requirements weaknesses are the root of a lot of project troubles: integration issues, late change, delays, …

E10 and E40 Relationship

ECSS-E-10

b. The system engineering organisation shall derive, generate, control and maintain the set of requirements for the lower level elements, defining their design and operational constraints and the parameters of functionality, performance, and verification necessary to meet the system requirements issued by the customer.

Document title	ECSS document	DRD ref.	Phase 0	Phase A	Phase B	
			MDR	PRR	SRR	PDR
Specifications						
Preliminary technical requirements specification	ECSS-E-ST-10-06	Annex A	+	+		
Technical requirements specification	ECSS-E-ST-10-06	Annex A			+	
Interface requirements document	ECSS-E-ST-10	Annex M		+	+	+
Preliminary technical requirements specifications for next lower level	ECSS-E-ST-10-06			+	+	
Technical requirements specifications for next lower level	ECSS-E-ST-10-06				+	+
Design definition file for next lower level						+
Interface control document	ECSS-E-ST-10-24	Annex A			+	+

(system)

(system) Functional Specification for Software

(software) Requirement Baseline

ECSS-E-40

Specification of system requirements allocated to software

5.2.2

Evaluation of system baseline

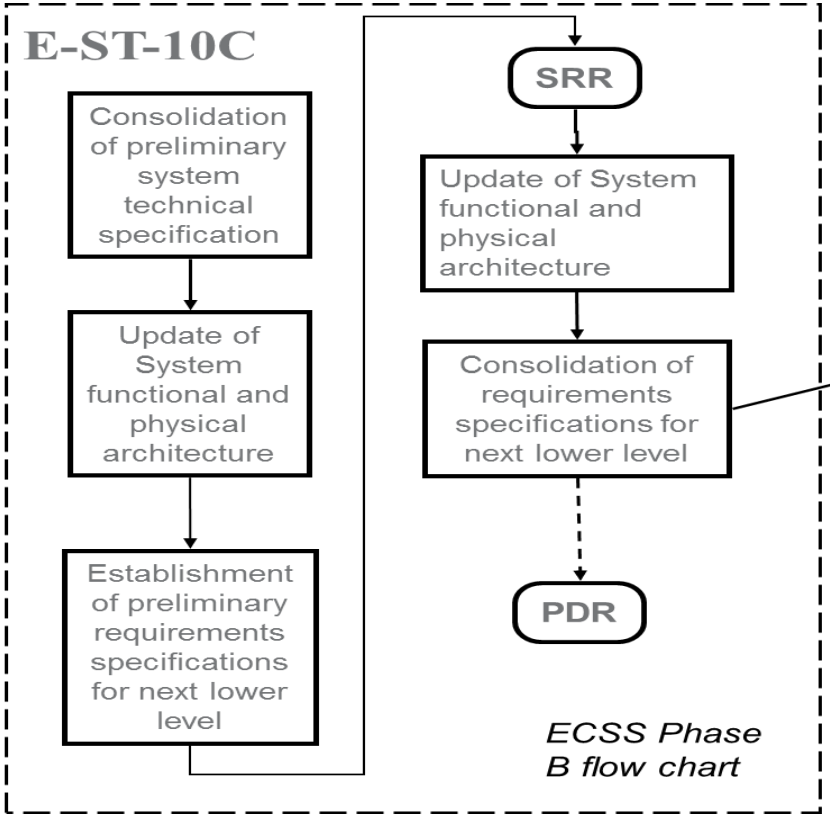
5.8.3.1 (software)

SRR

Establishment of the software Technical Specification [TS]

PDR

E-10 and E-40 Relationship



(software) Requirement Baseline

(system) Functional Specification for Software

Specification of system requirements allocated to software

Evaluation of system baseline

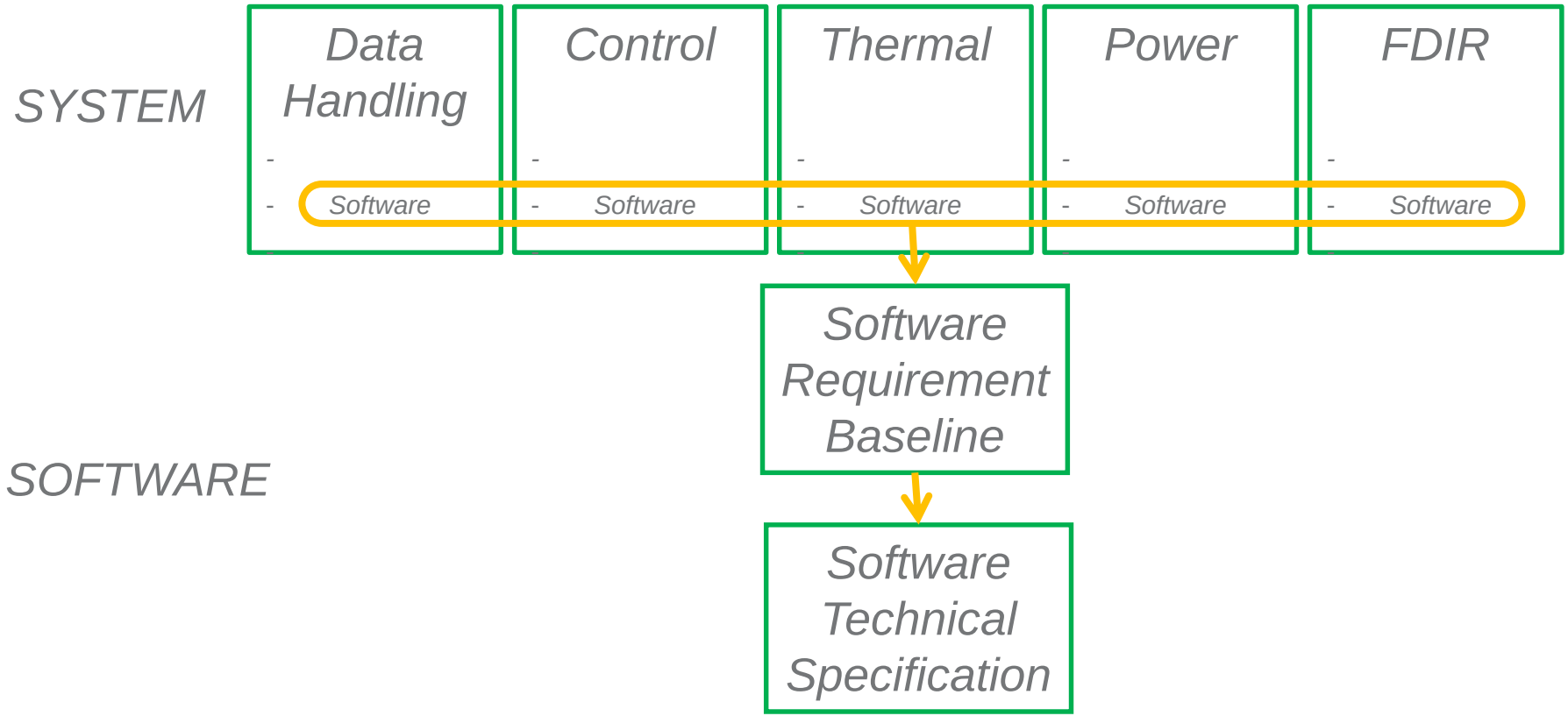
E-ST-40C

SRR

Establishment of the software Technical Specification [TS]

PDR

Software Requirements



What is Verification for System and Software?

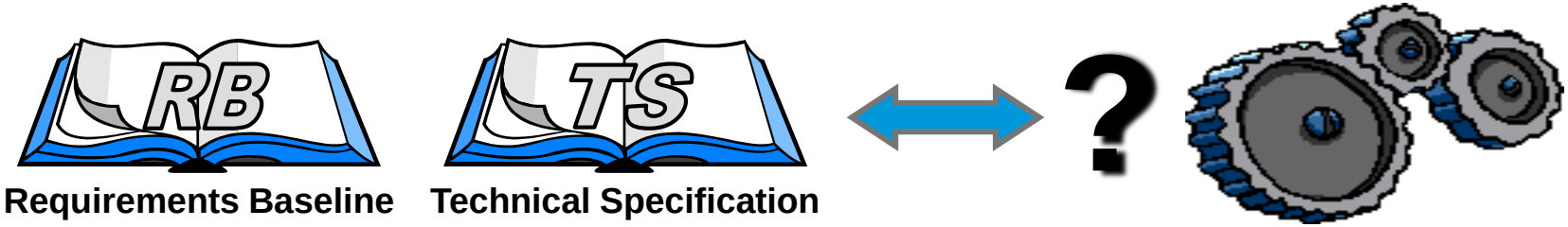
1. The **software verification** activities confirm that adequate specifications and inputs exist for any activity and that the outputs of the activities are correct and consistent with the specifications and inputs
    a. "Are we doing the thing right?"

correct & consistent?

- *The system verification activities are more concerned with ensuring that the requested functionality has been implemented*

What is Validation for System and Software?

1. The **software validation** activities ensure that the functionality of the developed system really corresponds to what was specified in the Requirements Baseline and further detailed in the Technical Specification
    a. "Are we doing the right thing?"
    b. "Does the running system actually implement the promised functionality?"



Requirements Baseline Technical Specification

- *The system validation activities are more concerned with the way the system is used.*

Software and [ground] system, E40 and E70

Organisation of E70

5 ***Operations engineering***
5.1 General
5.2 Requirements analysis and concept development
5.3 Mission operations data preparation
5.4 Mission operations data validation
5.5 Operations teams build–up and training
5.6 Operational validation
5.7 Operations execution
5.8 Space segment disposal

6 ***Ground segment engineering***
6.1 General
6.2 Ground segment definition
6.3 Ground segment production
6.4 Ground segment AIT and verification
6.5 Ground segment maintenance
6.6 Ground segment disposal

7 ***Ground segment and operations lifecycle***
7.1 General
7.2 Phase A: Mission and operational analysis, feasibility studies and conceptual design
7.3 Phase B: Preliminary design
7.4 Phase C: Detailed design
7.5 Phase D: Production, AIT and verification
7.6 Phase E: Mission operations
7.7 Phase F: Disposal
7.8 Summary of key documents and reviews

The Link Between E-70 and E-40

Ground Segment Engineering

System **E-70** Engineering

Sub System **E-40** Software

Space Software Engineering

1. ***Requirements engineering (GSRD/SURD)***
2. ***Analysis + Design ~~and configuration~~***
3. ***Verification***
4. ***Integration and control*** *(production)*

Software related system requirement process (E-40 Section 5.2)

Software related:

1. ***Requirements analysis***
2. ***Verification***
3. ***Integration and control***

This clause (5.2) of E-40 complements ECSS-E-70 for the specific software activities to be performed at system level by the customer

Software Requirements

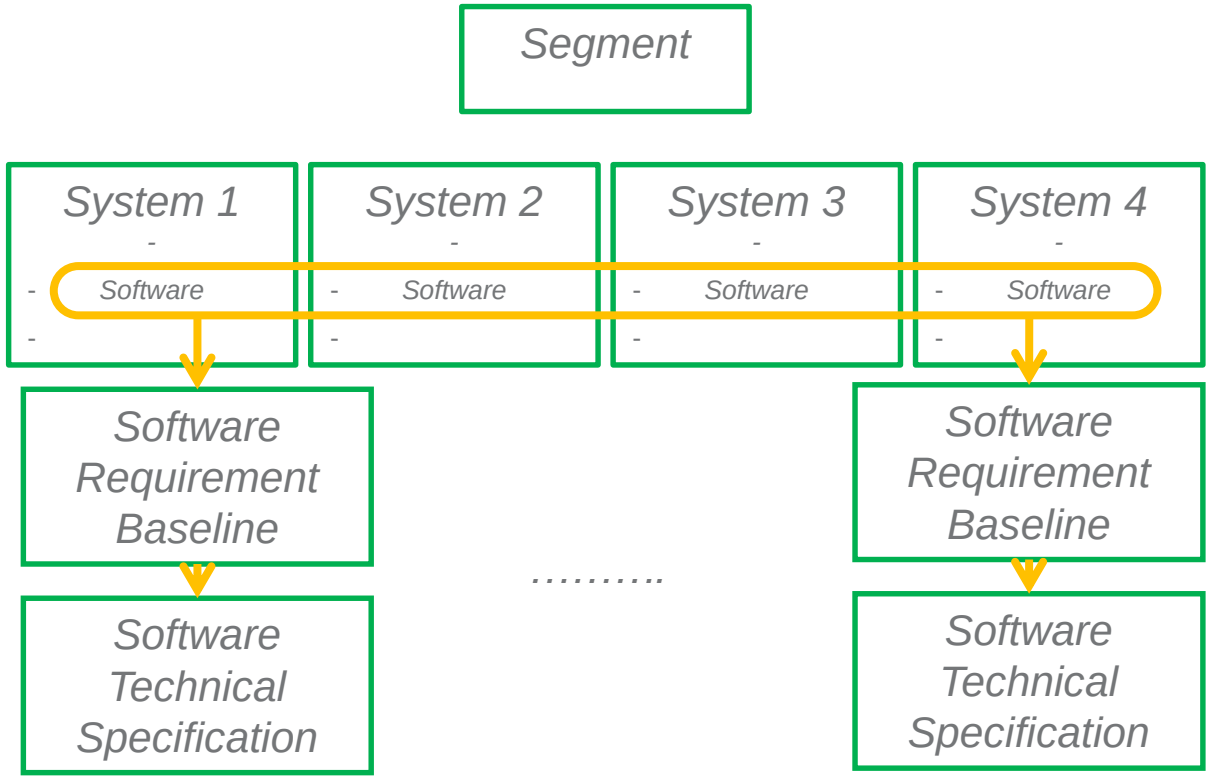
SYSTEM

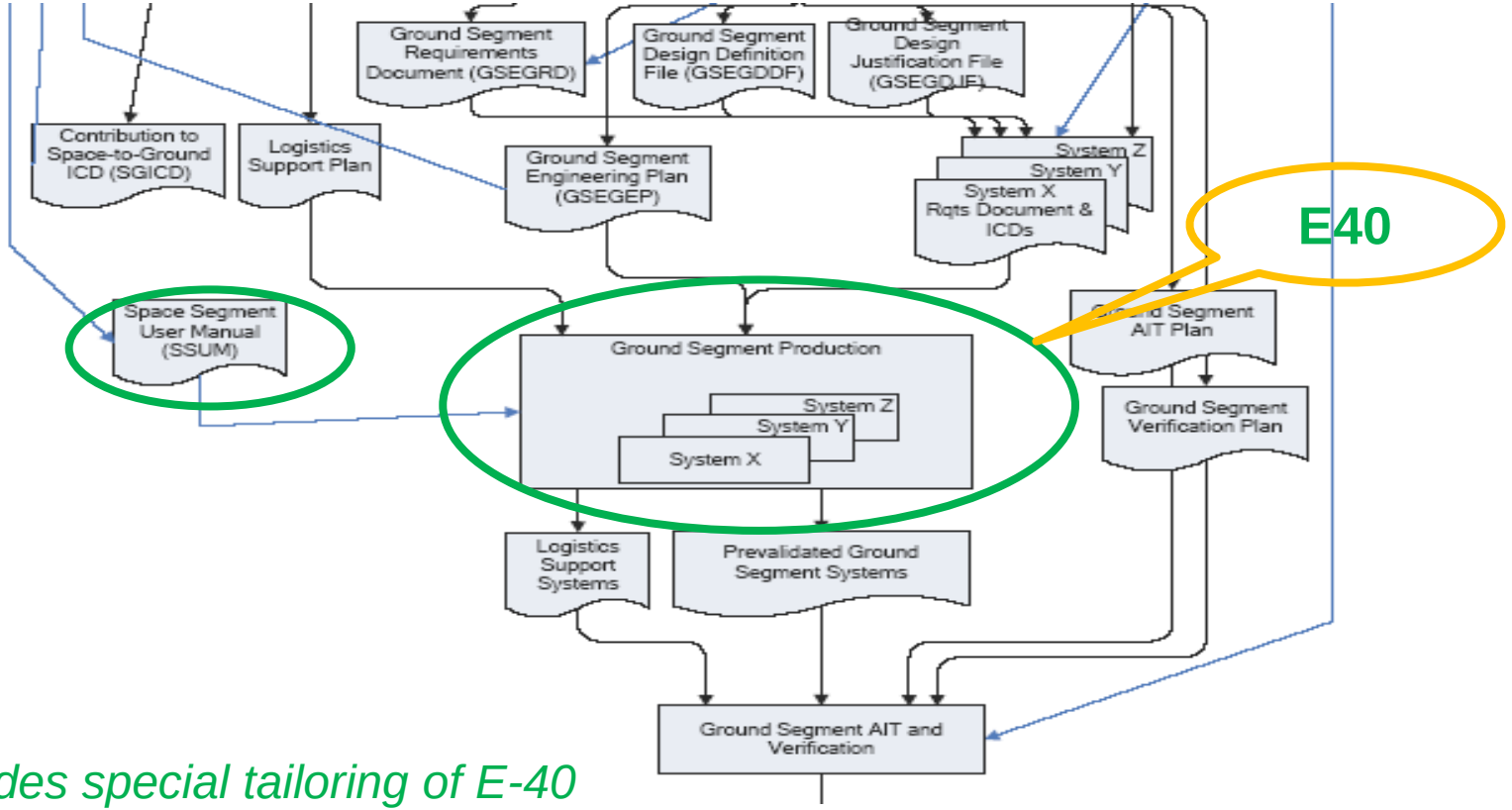
(SUB-)SYSTEM

Here E70 calls E10

SOFTWARE

Here E70 calls E40/Q80



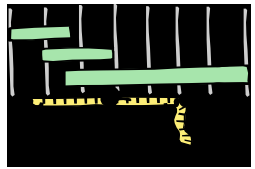


ESOC provides special tailoring of E-40

Project Planning and Implementation, E40 and M10

Project Planning and Implementation

1. Project planning and implementation is the project function, encompassing a coherent set of processes for all aspects of project management and control.
2. The E-40 software management process tailors M-10 for software to:
    a. define phases and formal milestones enabling the progress of the software project to be controlled
    b. define the software project breakdown structures to:
        - identify the tasks and responsibilities of each actor;
        - facilitate the coherence between all activities of the whole project;
        - perform scheduling and costing activities.
    c. set up the software project organization to perform all necessary activities on the project



M-10

Software Development Plan

Annex O of ECSS-E-ST-40C: SDP DRD

- Management Approach (can be also in the project's SDP)
    - Objectives, priorities, master schedule, assumptions, dependencies, constraints, WBS, risk management, monitoring & control mechanisms, staffing plan, software procurement process, supplier management
- Software development approach (strategy, **development life cycle** [identification, relation with system life cycle, reviews and milestones and their documentation])
- Standards and Techniques (requirement analysis, design method, autocode, HMI standard, delivery format)
- **Development environment, testing environment** (requirement tool, design tool, compiler/linker, conf management, static analysis, test scripting language, testing tools)
- Documentation plan

Software Validation Plan

Annex J of ECSS-E-ST-40C: SValP DRD

- Management Approach (can be also in the project's SDP)
    - Approach, effort, independence, organisation, schedule, resource, responsibilities, tool, techniques, methods, [independent]] personnel, risks
- Validation tasks identification (description, item under tests, success criteria, resuming after interrupt, input, output, resources)
- **Validation approach** (requirements on testing activities, kind of tests to be executed; inspection/analysis/review of design approach; regression testing)
- **Validation testing facilities** (test environment, configuration [software, hardware, test equipments, comms, testing data, simulators, etc]
- Control procedures (problem reporting, problem resolution, deviation, waiver, configuration management)

Software configuration management, E40 and M40

Software Configuration Management

1. The **product tree** is the breakdown of the project into successive levels of hardware and software products or elements called Configuration Items (CI) [M-10]
2. For each software product CI, a software configuration file (SCF) is prepared to provide the configuration status of the software CI
    a. It controls its evolution during the programme or project life cycle
3. The SCF is a constituent of the design definition file and is called from M-40, requirement 5.3.3.2b and from E-40 and Q-80

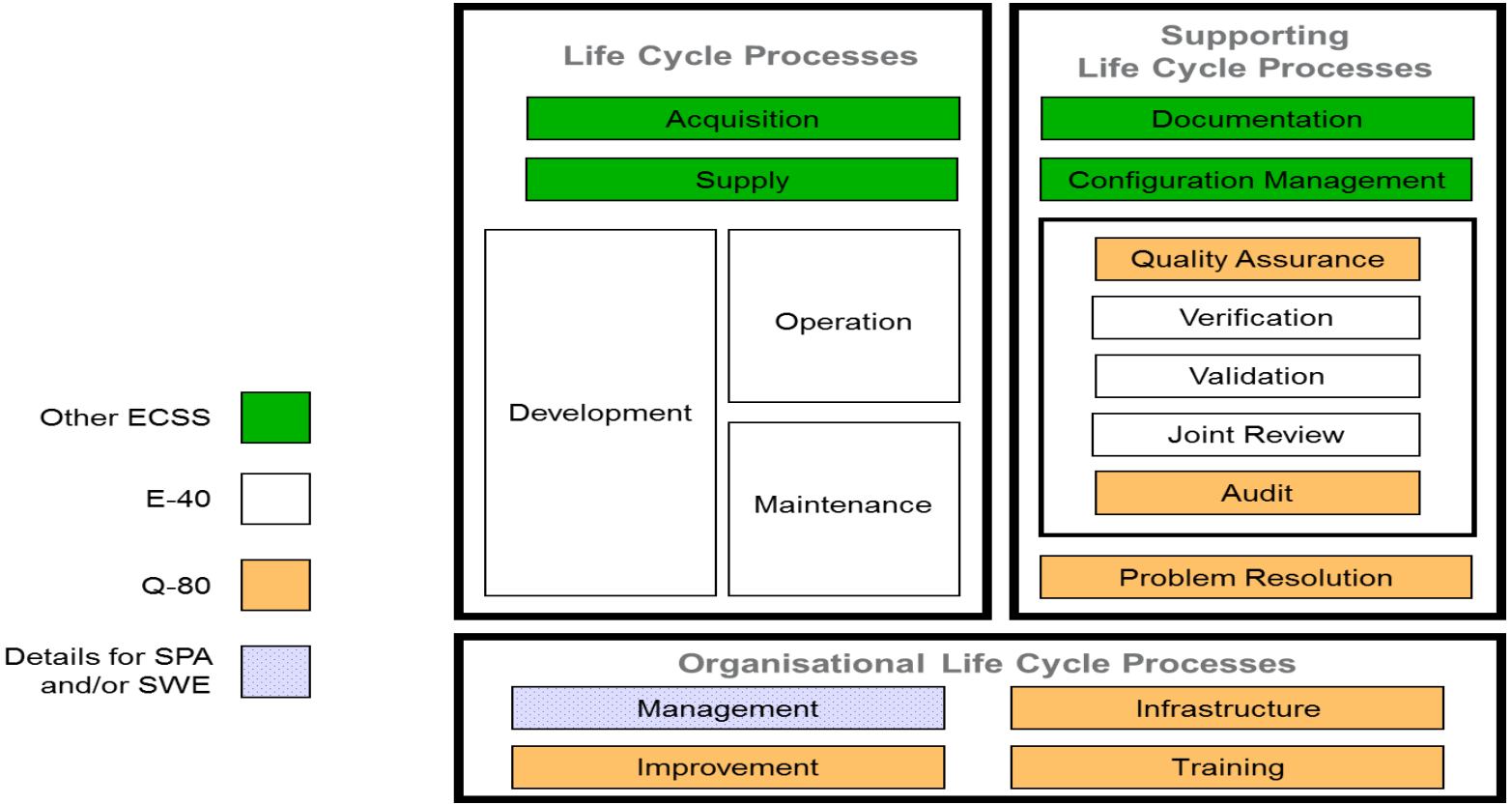
☞ **5.3.3.2 As designed data list** ☜

b. For each deliverable software CI, the supplier shall provide a software configuration file (SCF) in conformance with Annex E, software configuration file DRD.

M-40

Software product assurance, E-40 and Q-80

Software related processes in ECSS standards



The Objectives of Software Product Assurance

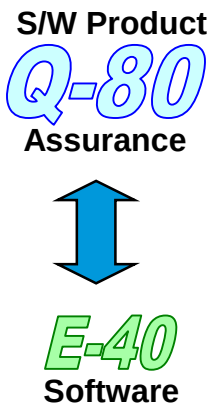
1. The objectives of software product assurance are to provide adequate confidence to the **customer** and to the **suppliers** that developed or procured/reused software satisfies its requirements throughout the system lifetime
    a. In particular, that the software is developed to perform **properly** and **safely** in its operational environment, meeting the quality objectives agreed for the project
2. SPA consists of both:
    a. The assurance of the **process** (software process assurance)
    b. The assurance of the quality of the **product** (software product quality assurance)

Q-80

4.1-2

The Relationship between E- 40 and Q-80

1. E-40 covers all aspects of space software engineering from requirements definition to retirement
2. Q-80 complements E-40 with product assurance aspects, integrated in the space system software engineering processes as defined in E-40
3. Q-80 is the entry point for E-40 into the Q-series of standards
4. Equally, the interface of Q-80 to the E-series of standards is via E-40
5. Together the two standards specify all processes for space software development



Q-80

4.1

References from E- 40 to Q-80

1. Q-80 requirements are directly referenced and made applicable through E-40 requirements

☞ 5.2.4.8 Software safety and dependability requirements☜

a. The customer shall specify the software safety and dependability requirements in accordance with ECSS-Q-ST-80 clauses 5.4.4, 6.2.2 and 6.2.3, based on the results of the safety and dependability analysis performed at system level.

🗁 Expected Output 🗁

Software safety and dependability requirements [RB, SSS; SRR]

S/W Product
Q-80
Assurance

E-40
Software

E-40
5.2

References from E- 40 to Q-80

1. Q-80 requirements are referenced and made applicable through the DRDs defined in E-40 (normative)

SSS traceability to ECSS-E-ST-40 and ECSS-Q-ST-80 clauses

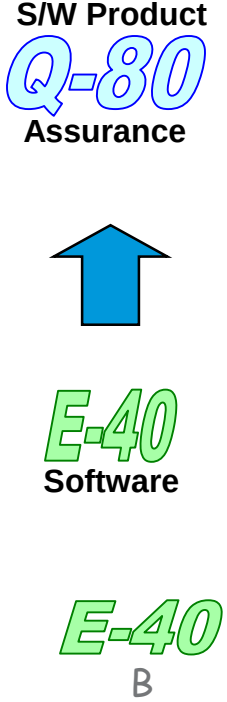
ECSS Standard	Clauses	DRD section
ECSS-E-ST-40	5.2.2.1 eo a., ……	<5.2> ……
	……	……
	5.3.8.1	<5.5>
ECSS-Q-ST-80	7.1.1 eo a	<5.9>
	7.1.2 eo a	<5.9>
	7.2.1.1. eo a	<5.9>
	7.2.1.3 eo a	<5.1>c.

…….

…….

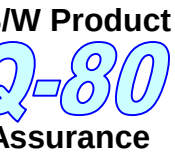
<5.9> Quality requirements

a. The *SSS* shall list the quality requirements applicable to the software (e.g. usability, reusability (5.2.4.7), and portability), and the applicable software development standards (5.2.4.5)



Main SPA Issues Referenced from E- 40

1. Software safety and dependability
    a. including criticality classification and E-40 tailoring
2. Product Quality requirements
    a. and their quantitative definition
3. Software reuse and procurement
    a. including identification and assessment/inspection
4. Software configuration management
5. (Independent) Validation and verification
    a. and testing
6. Software problems and nonconformances

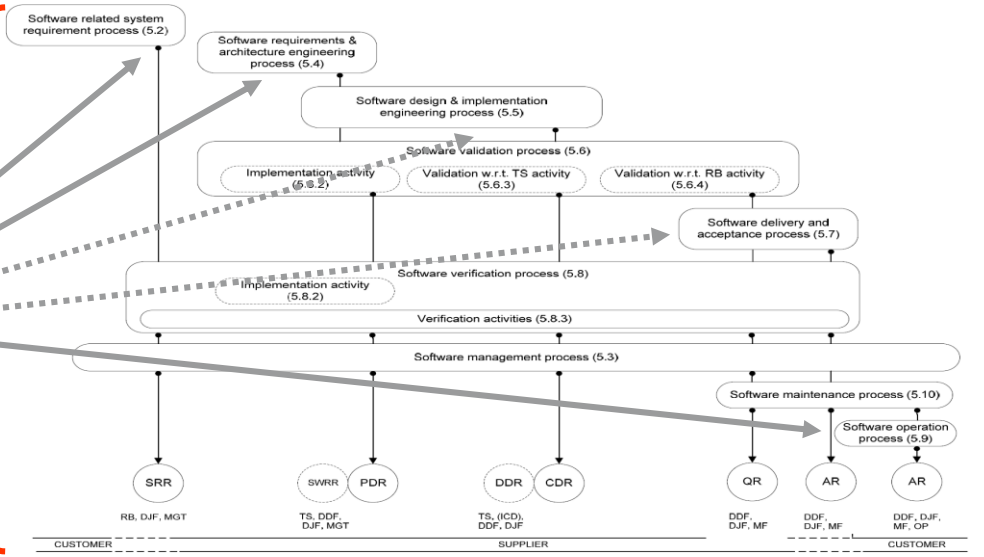


Main Q-80 Requirements on Engineering activities

applicable to all software engineering processes [Q-80 6.2]

applicable to individual software engineering processes or activities [Q-80 6.3]

on the software development life cycle [Q-80 6.1]



S/W Product

Q-80

Assurance

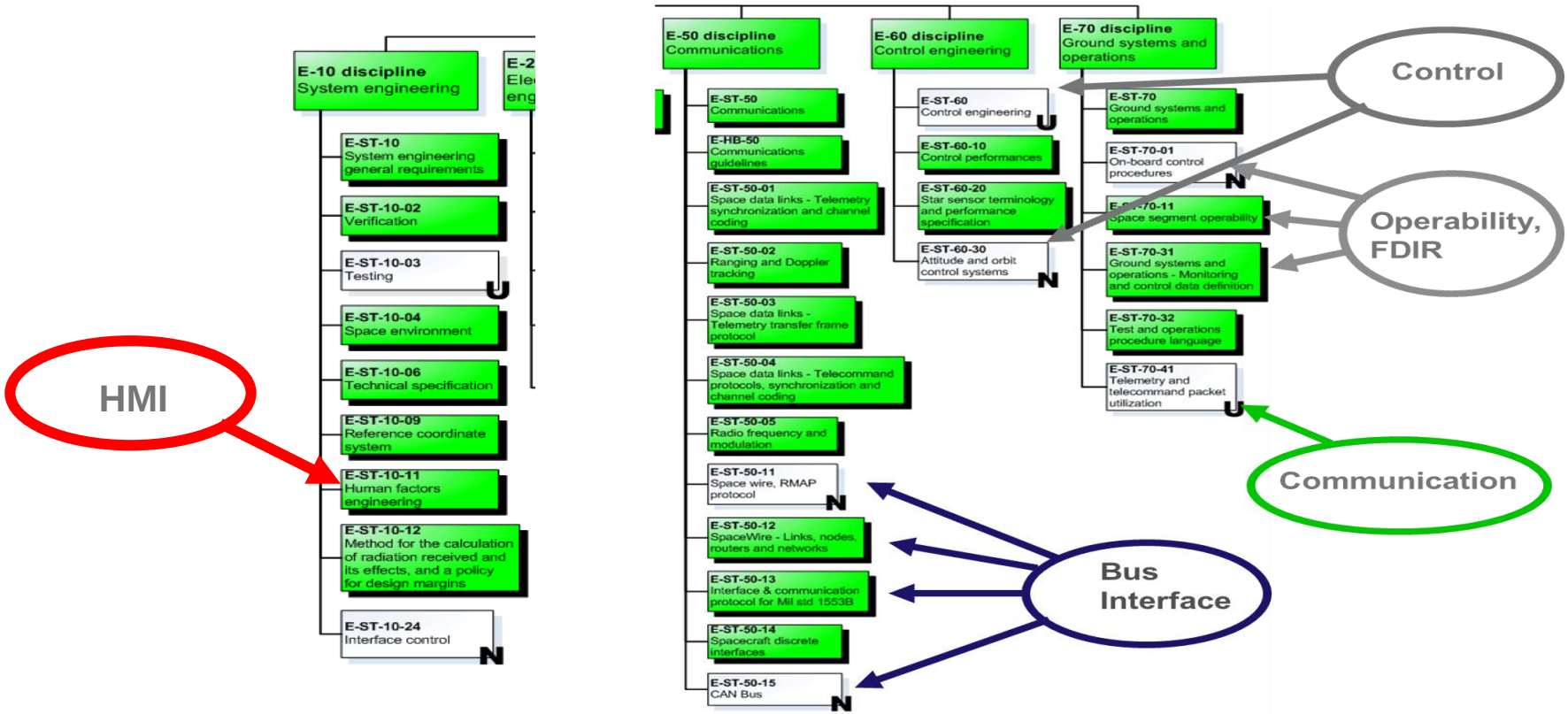
E-40

Software

Q-80

The Engineering standards generating software functional requirements

The Engineering standards generating software functional requirements



Summary of Part 1

Summary of Part 1

1. Space software engineering is part of the engineering branch of the ECSS standards

2. The **E-10** standard specifies implementation requirements for the responsible system engineering organization. E-40 complements E-10 for the specific software activities to be performed at system level
    a. The link is reflected in E-40
    Clause 5.2, Software related system requirement process
    b. These specific activities are performed in the project **phase B**
    c. They can be delegated by the customer to the supplier
3. **Software related system requirements are important**

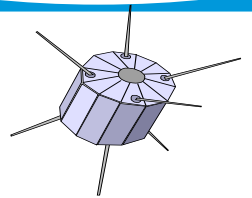
4. E70 is the ground segment and operability standard.
5. M10 and M40 relate to project and configuration management
6. Q80 complements E-40 with respect to Software quality assurance
7. Several technical standards (not process models) generate software functional requirements.

PART 2:
A Roadmap to the Standard

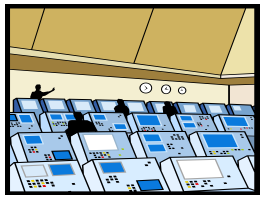
Abstract

ECSS-E-40 is the standard for space software engineering. This module provides a road map to the standard, introducing the participant to its key concepts, and processes.

A Standard for All Space Software



1. The E-40 standard is intended for application to **all software** developed as part of a space project
    a. flight software
    b. ground software



2. For ASIC and FPGA:
    a. ECSS-E-ST-20-40C ASIC and FPGA engineering (<=> E40 for SW)
    b. ECSS-Q-ST-60-02D ASIC and FPGA product assurance (<=> Q80 for SW)

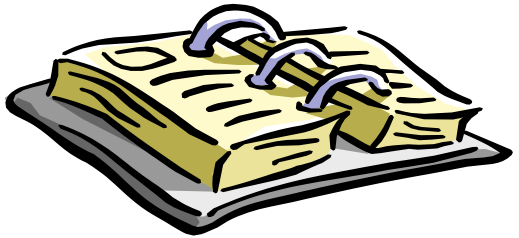
NEW

software
set of instructions and data executed on a processing unit
NOTE 1: A processing unit can be hardware, e.g. a processor or software, e.g. a virtual machine or an interpreter.
NOTE 2: Some processing units only require data, e.g. configuration of state machines or configuration data of a neural network.

ECSS-E-40 Status

ECSS-E-40A 13 April 1999
First approved issue

ECSS-E-40B Draft July 2000
Not a formal version but applied to several projects
Public Review Version



ECSS-E-40 Part 1B November 2003,
ECSS-E-40 Part 2B 31 March 2005
Previous published version

ECSS-E-ST-40C 6 March 2009
Published Version
It is the reference for this course

ECSS-E-ST-40C Rev1 Q4 2021
Public Review planned on Q3 2021
Some important aspects are also included in this course

Available from www.ecss.nl

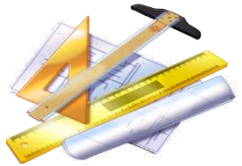
E-HB-40: The E-40 Handbook

1. The main objective of the handbook is to collect software engineering best practices for the implementation of E-40 requirements
    a. It covers both flight and ground software
    b. It comes from project lessons learned
2. "*Getting started*" and "*Getting compliant*" introduction
3. Guidelines for each software process in E-40
4. Focus on specific issues (automatic code generation, reuse, on-board control procedure, etc.)
5. Technology supplements (use cases & scenarios, model driven engineering, real-time, testing for dependability), and addresses some generic engineering techniques.

Publication 11/12/2013

Revision initiated Q3 2021 to align with ECSS-E-ST-40C Rev1

Life cycles



Methods, MDE

V&V

Real-Time

E-HB-40: Software Life Cycle

Life cycles

- Waterfall (not iterative) : RB frozen
- Incremental : RB frozen
- Evolutionary, RB evolve
- Spiral, Agile: RB evolve, risk related to final product → *ECSS-E-HB-40-01A (07/04/2020)*

Reviews and iterative life cycles

- SRR all versions
- PDR early version
- DDR middle version
- CDR, QR on last versions
- AR on the last

	V1		V2		V3		V4	
	Project	Technical	Project	Technical	Project	Technical	Project	**Technical**
SRR	X		X		X			
PDR	X		X			X		
DDR		X	X			X		
TRR				X		X		
TRB/DRB		X		X				
CDR					X			
QR					X			
AR							X	

E-HB-40: Logical Model

- Support the TS
- Representation of the requirements
- Independent from the implementation
- Used to check completeness and consistency
- Possibly executable

E-HB-40: Unit tests and code coverage

UT objectives: check correctness of unit source code against design

No other way of testing than UT for :

- low level sw, hw i/f, drivers
- complex units, error management code
- boundary testing

Code coverage:

- contribute to UT objectives (cover all software)
- contribute to all sw reliability
- can be achieved by other mean that UT (e.g. functional tests)

Tailoring drivers:

- criticality of unit
- combine with functional tests , check coverage and complete
- level or size of the units

E-HB-40: Autocode

Basic need:
a modelling style standard
to make sure the model is autocodable

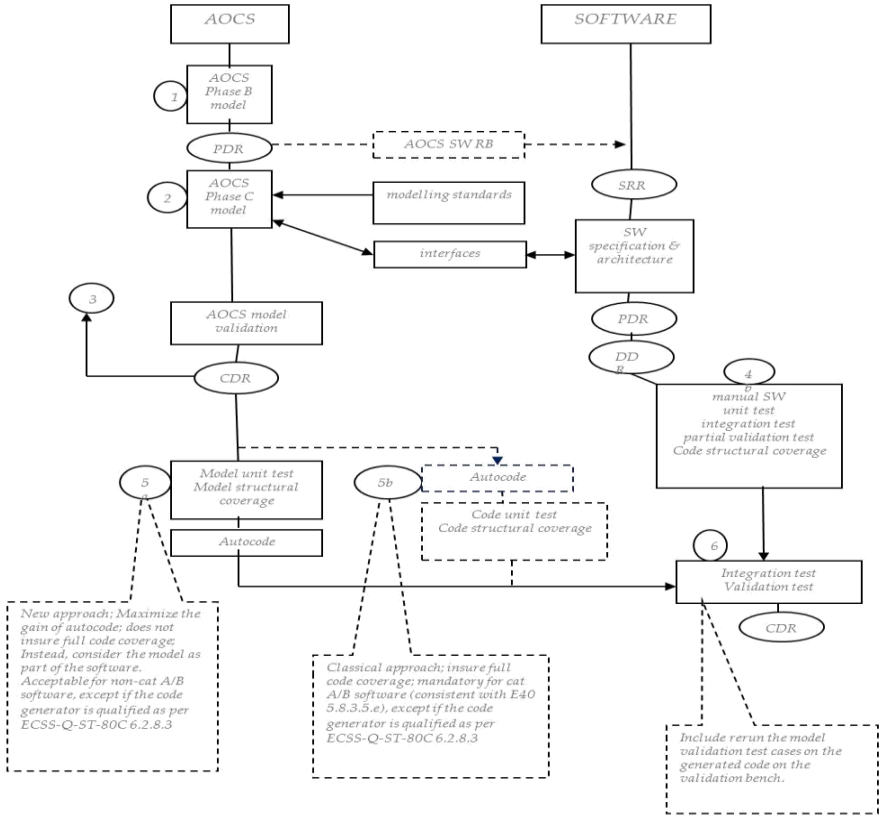
Traditional way:

Model -> autocode
-> UT, IT, structural coverage

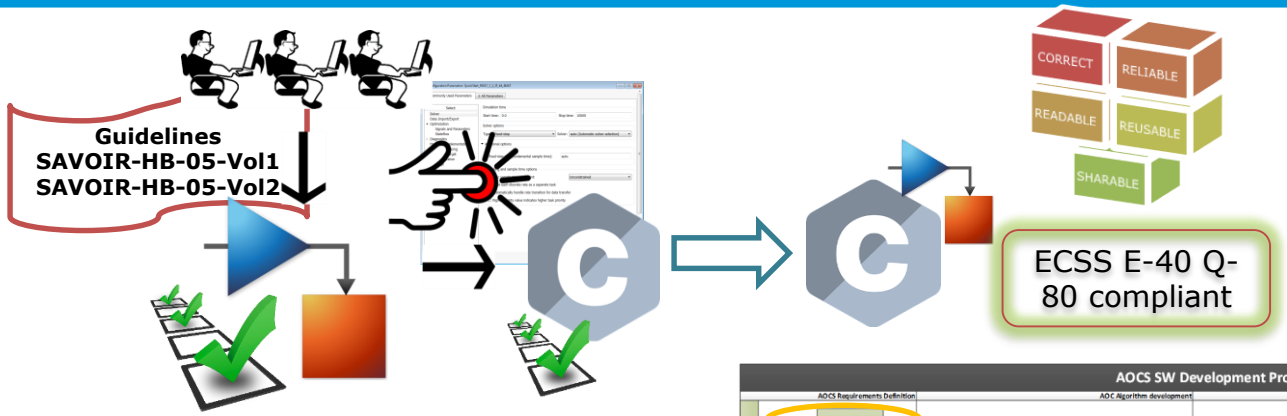
Envisaged way:

Model -> model UT & structural coverage
-> autocode

►SAVOIR-HB-005

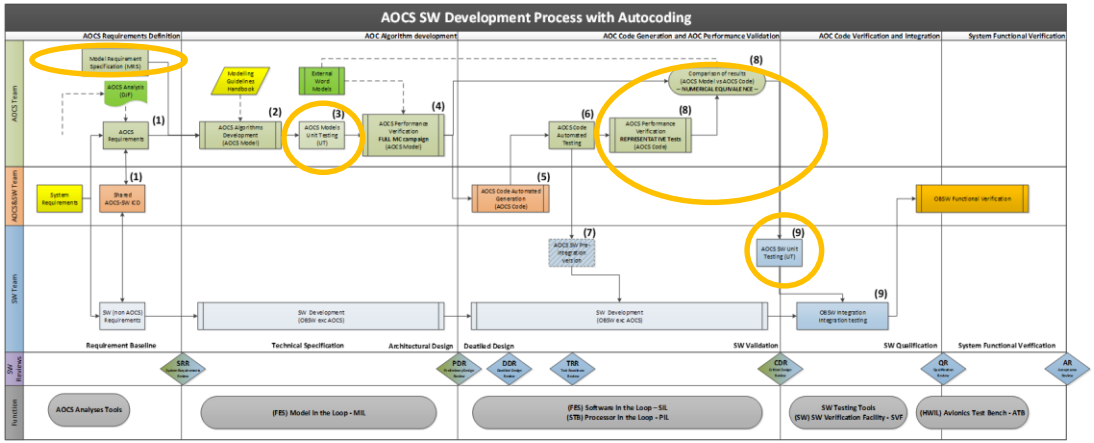


SAVOIR-HB-005



I. *Vol I: General concepts*
   a) *Development and verification process*
   b) *Compliance with existing standards*

II. *Vol II Modelling guidelines (AOCS modelling)*
   a) *Define modelling guidelines*
   b) *Configuration of code generation toolboxes*
   c) *Classification of guidelines*



The E-ST-40-07: Simulation Model Portability

1. To enable simulation **model** reuse between project phases as well as between projects thus reducing cost of overall **simulator** developments as well as contributing to knowledge capturing.
2. The project activities requiring simulation support are described in E-TM-10-21.
3. The scope of the standard addresses the definition of simulation model interfaces and the associated development process in order to enable:
    a. Simulation Model Portability
    b. Simulation Model Reuse
    c. Model Development Productivity
    d. Simulation Model Integration
    e. Support for a model driven engineering process
    f. Support for simulation model meta data
    g. Support for dynamic simulations
    h. Handbook containing guidelines how to apply the ECSS-E-40-07 standard.
    i. Data integration

The E-40-07: Simulation Model Portability

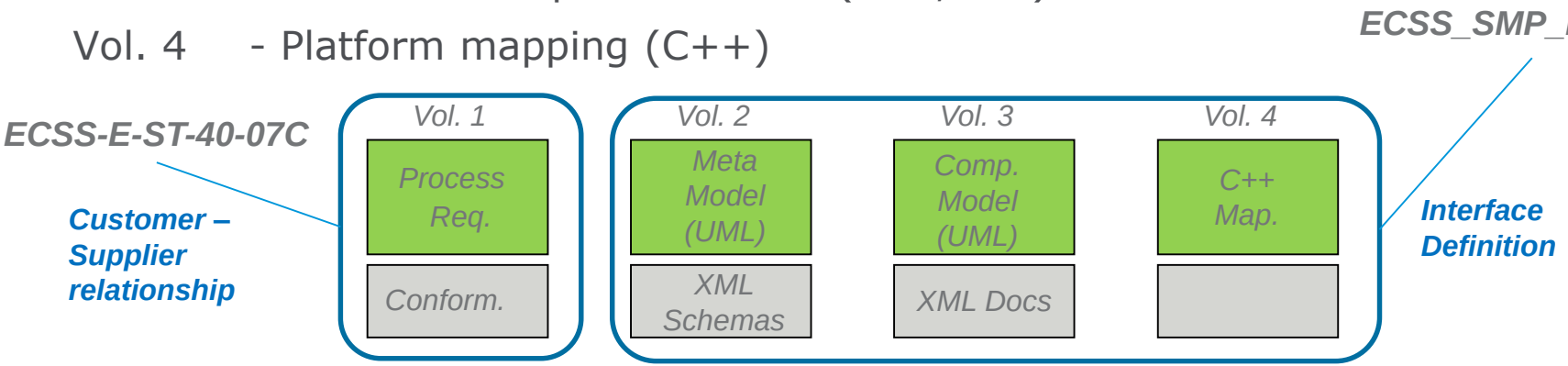
Vol. 1 - Process extending the E-40 Software Development Standard

- Requirements between Customer and Supplier – mainly defines the deliverables

Vol. 2 - Platform independent language defining the simulation models (UML, XML Schemas)

Vol. 3 - Software component model (XML, IDL)

Vol. 4 - Platform mapping (C++)



Overview of the E-40 Standard

Principles
(Section 4)

- Key Concepts
- Introduction to the processes

Requirements
(Section 5)

- Requirements on each process

Tailoring
(Annexes R and S)

- Pre-tailoring per criticality A, B, C, D
- Tailoring guidelines

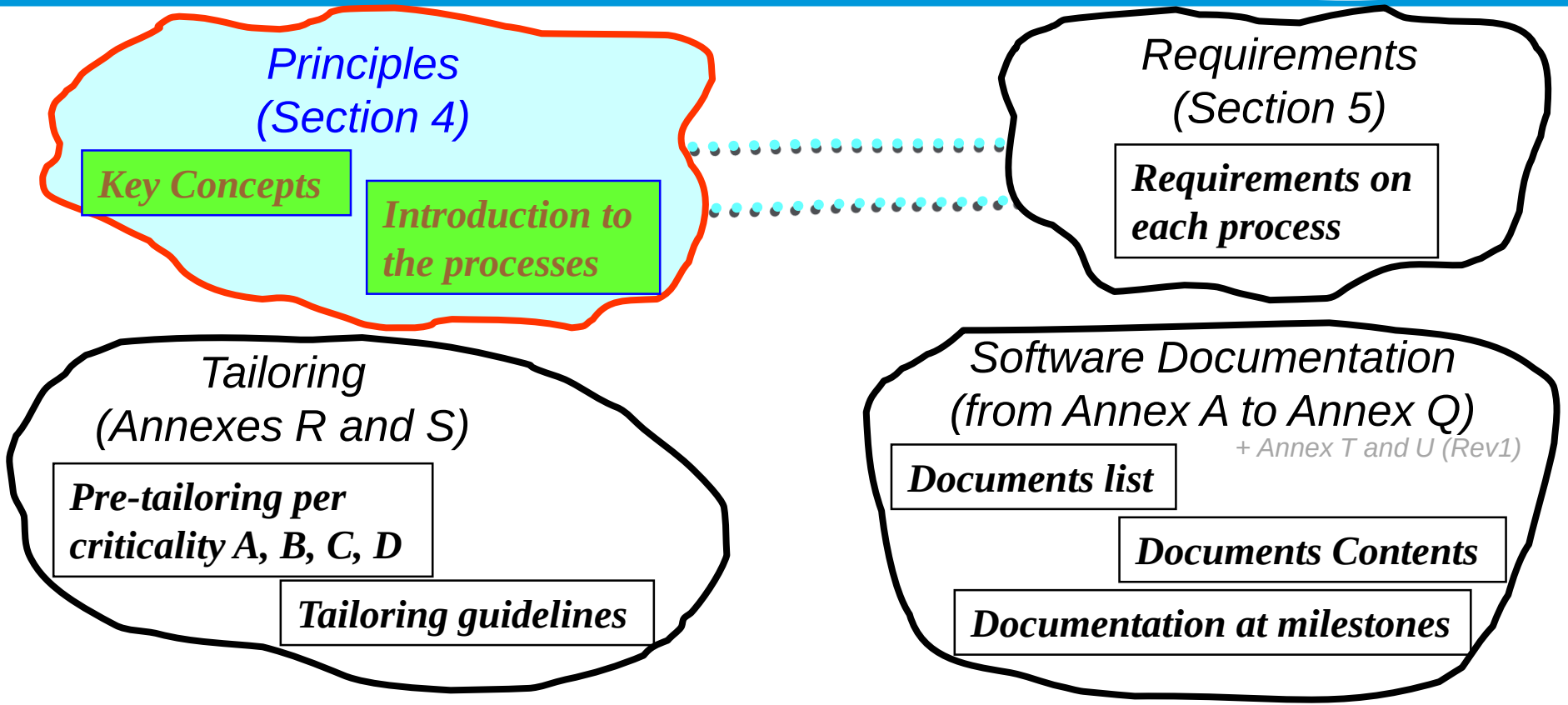
Software Documentation (from Annex A to Annex Q)

+ Annex T and U (Rev1)

- Documents list
- Documents Contents
- Documentation at milestones

Principles (Section 4)

Space System Processes – Section 4



Key Concepts in the E-40 Standard

Software in space systems

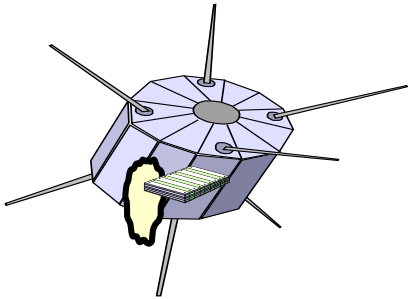
1. **Software is different from other engineering disciplines**
    a. Software has no mass, nor produces heat
    b. Software has no other physical property
2. **Software is highly flexible**
    a. Ideal for highly complex functions
    b. Increasingly used in space systems, from system level functions to the basic functions of a specific device
    c. Related effort (requirements, design, test) often underestimated
3. **Software engineering is a *pure* intellectual activity**
    a. Principal output is *documentation* (comprising code)
    b. Focus of the E-40 standard is on requirements for *contents* and *structure* of documentation

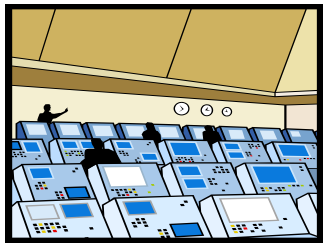
Summary of Key Concepts in E-40

1. **Software is part of the overall System**
   a. Software is not to be treated in isolation
2. **Customer-supplier relationship**
   a. The relationship is made explicit
3. **Reviews as synchronization points**
   a. Reviews are a point of synchronization for the lifecycle processes
4. **Process orientation**
   a. Logical orientation (processes) rather than time-based (phases)

1- Software is Part of the Overall System

1. E-40 makes explicit the fact that Space Projects generally involve many engineering disciplines, of which software is only one
2. This is reflected in the inclusion of requirements on system engineering processes related to software in the Standard

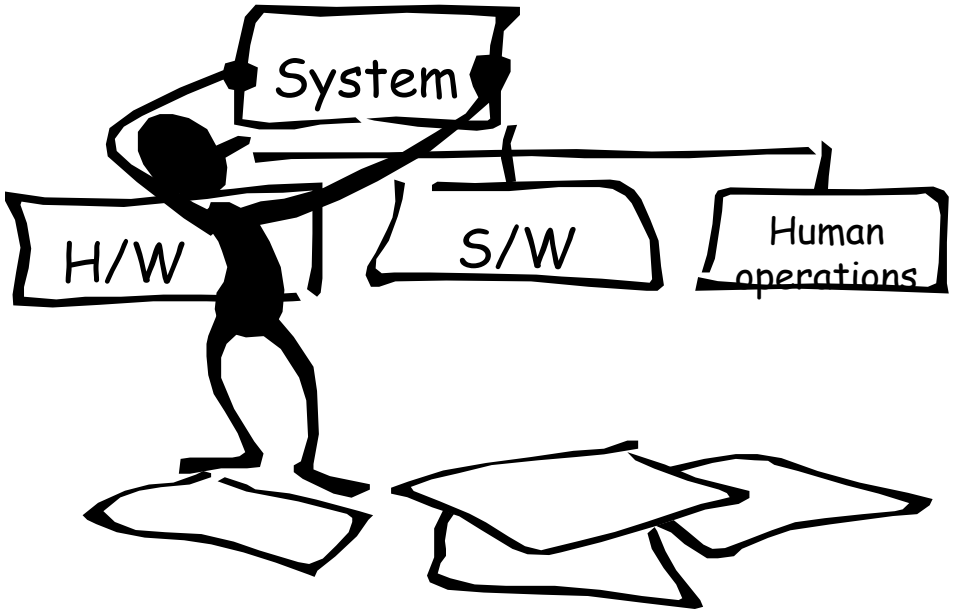




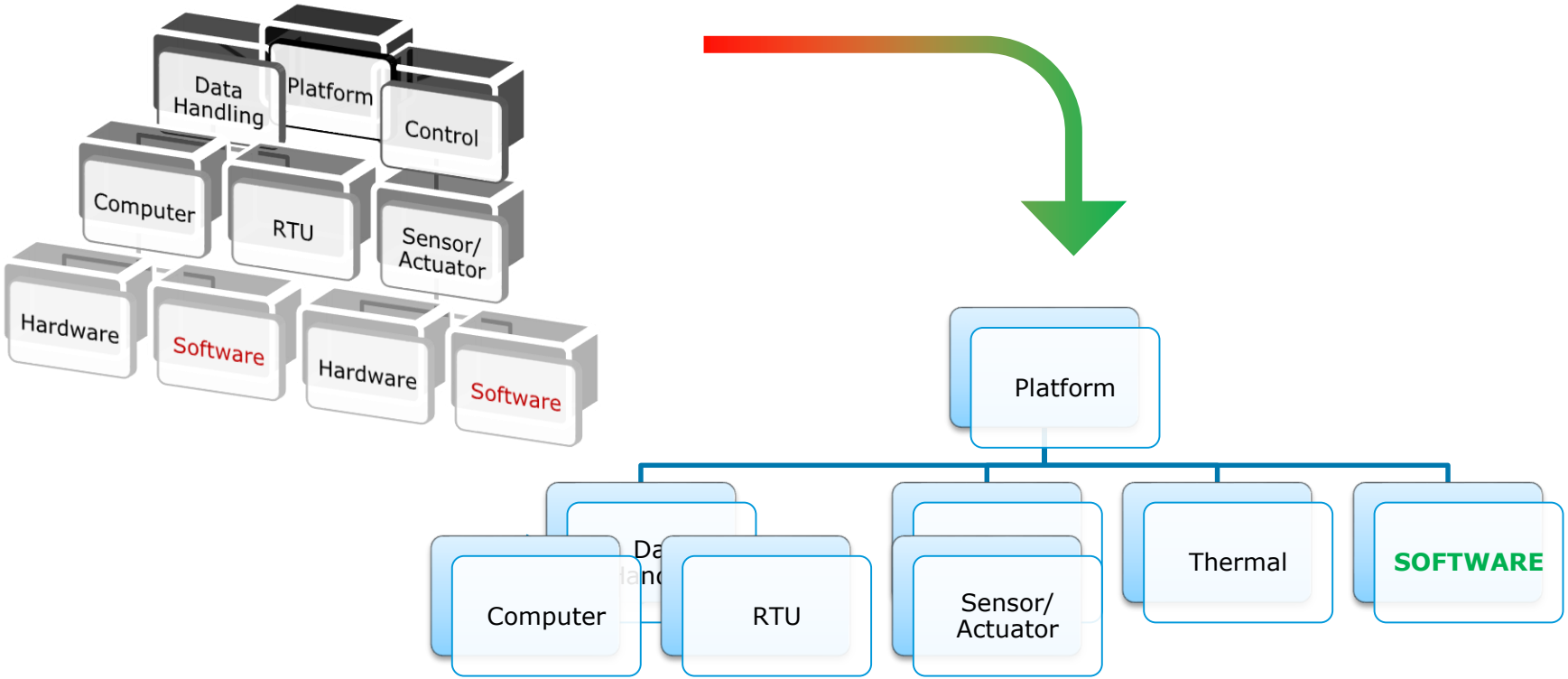
Software components are part of the overall mission system, together with other engineering components

1- Software is Part of the Overall System

1. As part of the **system design** process, a physical architecture and design at system level is created
2. This physical architecture includes *everything*: hardware, software, and human operations
3. The driving force is the system level requirements
    a. The requirements are allocated to the different subsystems



1- Software in the WBS



2- The Customer-Supplier Relationship

1. A fundamental principle in the E-40 Standard is the **customer-supplier relationship**
    a. it is assumed for all software development
    b. the organisational aspects are defined in M-10

Customer

Supplier

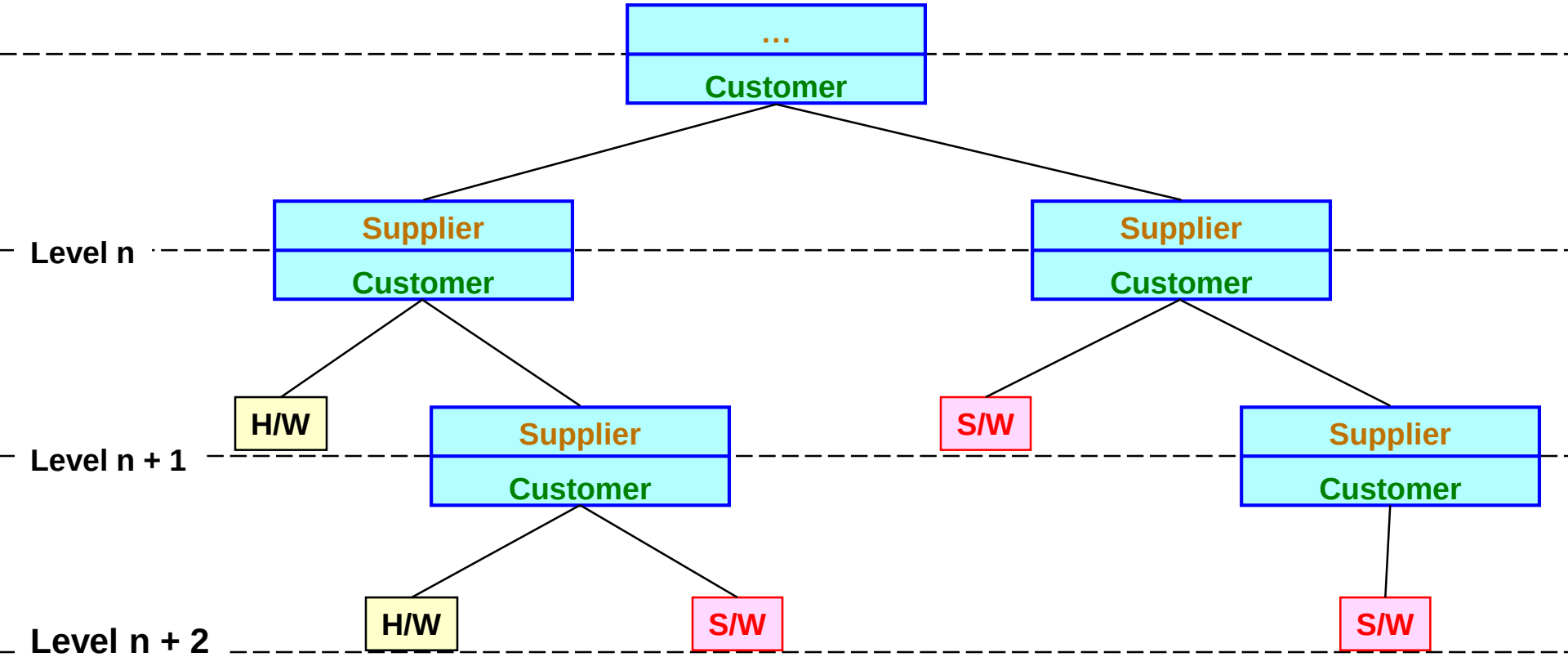
organisational aspects

Project
M-10
Organisation

2. Customer – Supplier
   or
   System - Software

E-40
4.2.1

2- The Recursive Customer-Supplier Model



3- Reviews are the Main Synchronisation Points

1. The reviews are the main interaction points between the customer and the supplier
    a. All reviews are applicable to software
    b. They are sequenced according to the overall system-level planning
2. The reviews are the main synchronisation points between processes

E-40
4.2.1

4- Origins of E-40 in ISO/IEC 12207

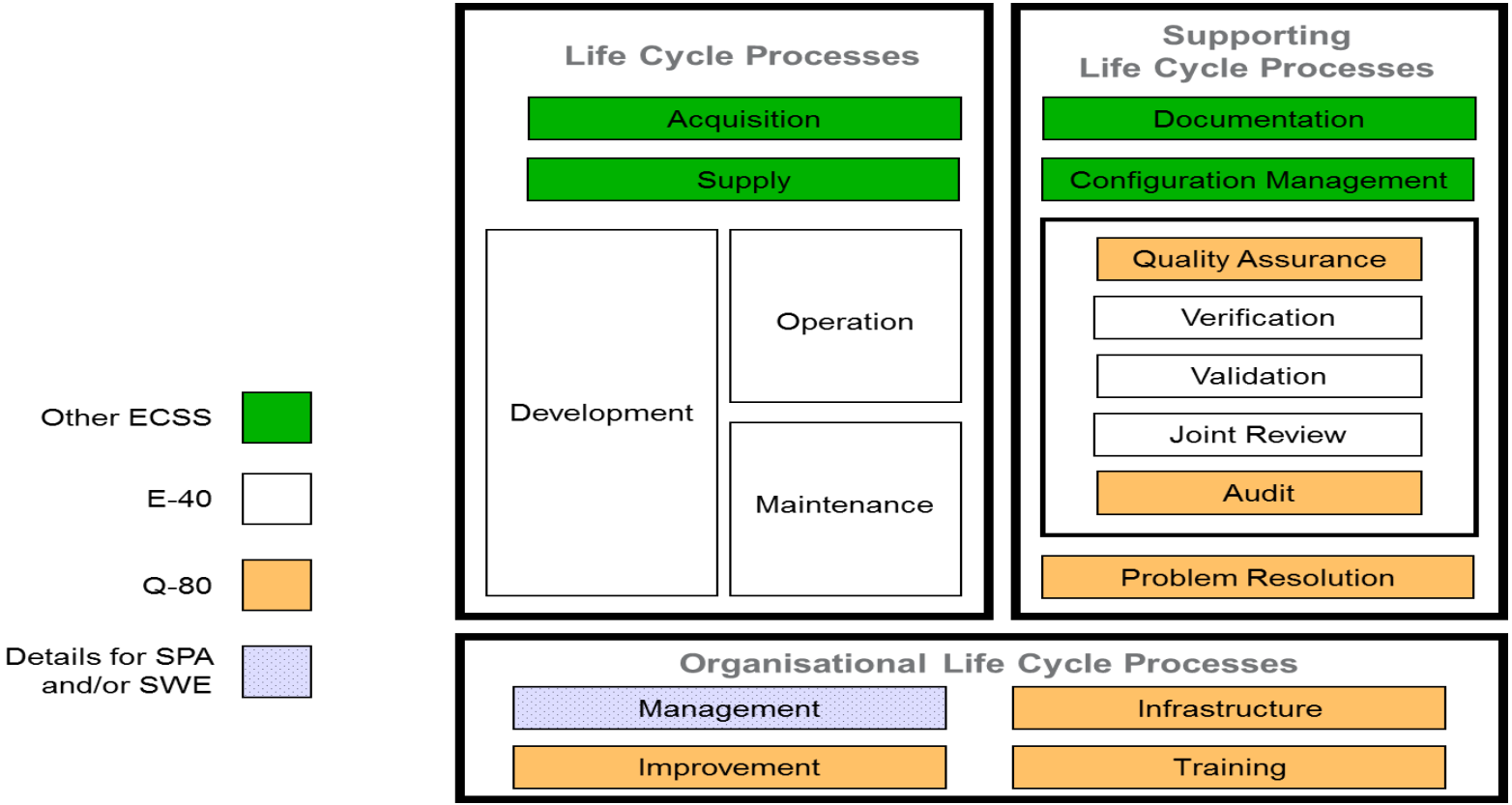
1. The structure and approach of the E-40 standard has its origins in the ISO/IEC standard 12207 (however not updated of the last release)
    a. Title: **Information Technology – Software Life Cycle Processes**
2. The ISO standard has a clear set of goals:
    a. "This International Standard establishes a **common framework for software life cycle processes**, with **well-defined terminology**, that can be referenced by the software industry."
    b. "It contains **processes**, **activities**, and **tasks** that are to be applied during the acquisition of a system that contains software, a stand-alone software product, and software service and during the supply, development, operation, and maintenance of software products."
    c. "This International Standard also provides a process that can be employed for **defining**, **controlling**, and **improving** software life cycle processes."

4- A Process-Oriented Approach

1. The major heritage from the ISO standard is the **process-oriented approach** to the life cycle
    a. More **freedom** this way!
2. Many previous approaches to engineering standards prescribed exactly *when* activities were to be carried out
    a. In contrast, the E-40 approach prescribes only *what* needs to be done, allowing the organisation considerable freedom in deciding when to do it, but respecting the constraints identified by the Reviews
3. For example, different life cycle models can be chosen by the organisation
    a. Waterfall, incremental, evolutionary, etc.
4. However… spacecraft waterfall model influences all subsystems…

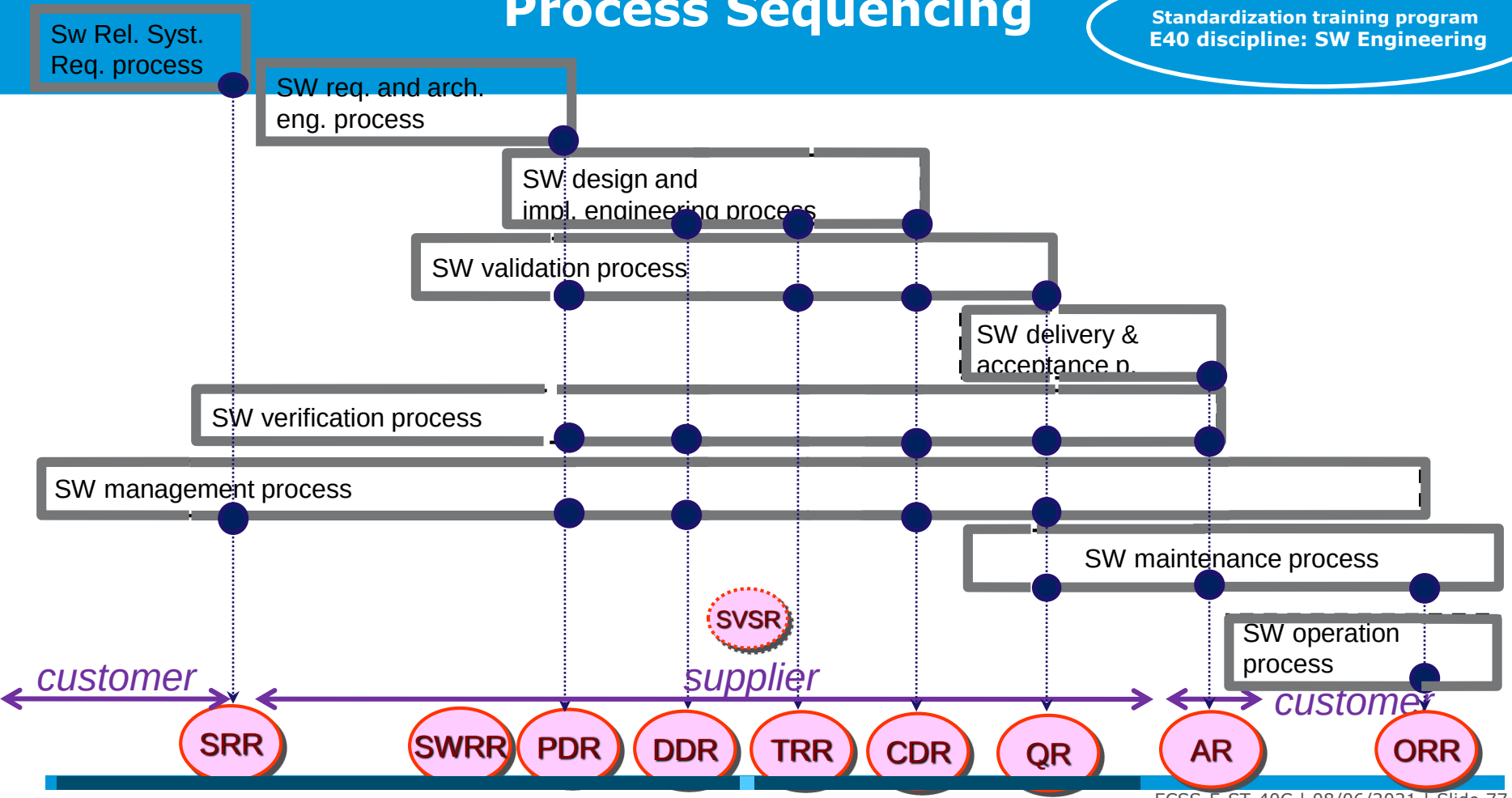
Freedom to choose an appropriate lifecycle model is one advantage of the process-oriented approach

4-ECSS covers the ISO/IEC 12207 Standard



Introduction to the Processes

Process Sequencing



Three levels of software validation

	First	Second	
Review	CDR	QR	AR
Against	TS	RB	RB
On (depend on model philosophy)	SVF	SVF/ATB/EM	Final environment (ATB/PFM)

E40 Roles

- Customer
- Supplier
- User: use functions of the software (i.e. reset)
- Software Operation Support Entity (SOS Entity): Help desk, hotline of the software
- Maintainer: correct the issues reported to SOS Entity, improve the software
- Operator:

Overview of E-40 Processes

5.2 Software related system requirements

- 5.2.2 Sw. rel. Syst. req. analysis
- 5.2.3 Sw. rel. system verification
- 5.2.4 Sw. rel. system integration & ctrl
- 5.2.5 System Requirement Review

5.4 SW req. & arch. engineering process

- 5.4.2 Software requirements analysis
- 5.4.3 Software architectural design
- 5.4.4 Preliminary Design Review

5.5 SW des. & impl. engineering process

- 5.5.2 Design of software items
- 5.5.3 Coding and testing
- 5.5.4 Integration

5.6 Software validation process

- 5.6.2 Validation process implementation
- 5.6.3 Validation w.r.t. the technical spec.
- 5.6.4 Validation w.r.t. the req. baseline

5.7 Software delivery and acceptance process

- 5.7.2 Software delivery and installation
- 5.7.3 Software acceptance

5.8 Software verification process

- 5.8.2 Verification process implementation
- 5.8.3 Verification activities

5.9 Software operations process

- 5.9.2 Process implementation
- 5.9.3 Operational testing
- 5.9.4 Software operation support
- 5.9.5 User support

5.10 Software maintenance process

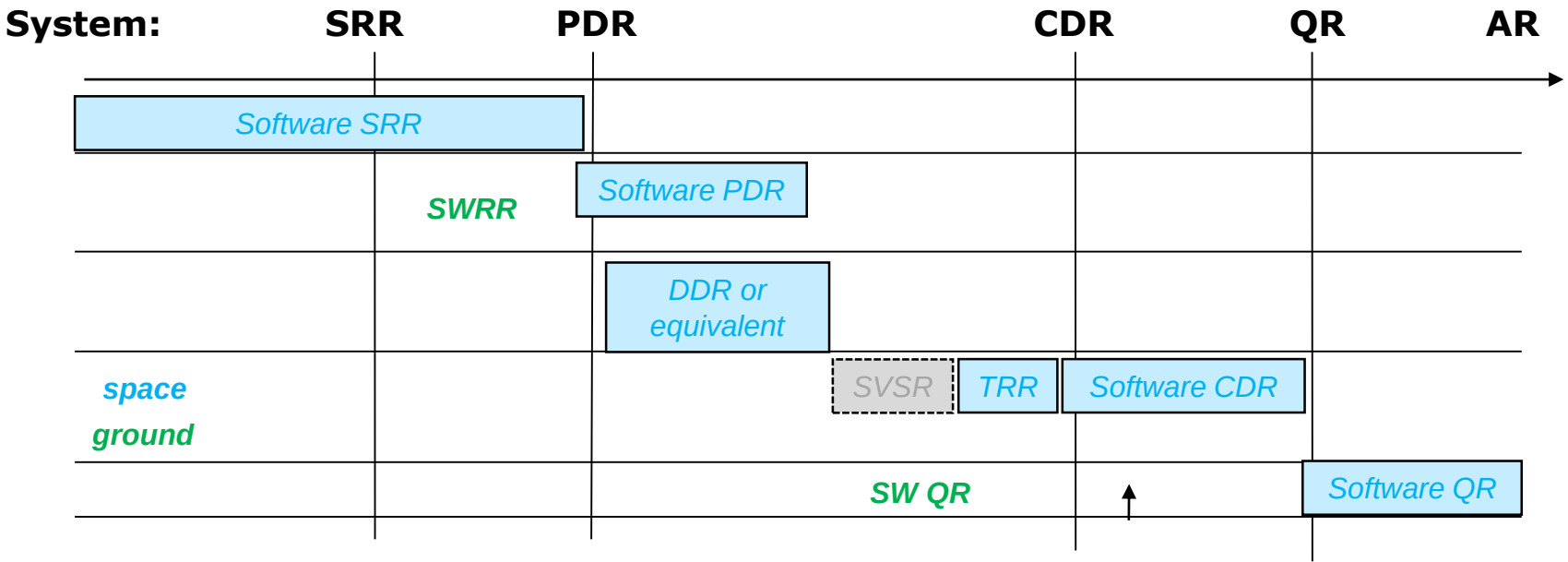
- 5.10.2 Process implementation
- 5.10.3 Problem & modific. analysis
- 5.10.4 Modification implementation
- 5.10.5 Conducting mainten. reviews
- 5.10.6 Software migration
- 5.10.7 Software retirement

5.3 Software management process

- 5.3.2 Sw life cycle managmt.
- 5.3.3 Joint review process
- 5.3.4 Sw. Proj. Rev. Descr.
- 5.3.5 Sw Tech. Rev. Descr.
- 5.3.6 Review Phasing
- 5.3.7 Interface management
- 5.3.8 Tech. bdg & margin mngt
- 5.3.9 Compliance to Standard

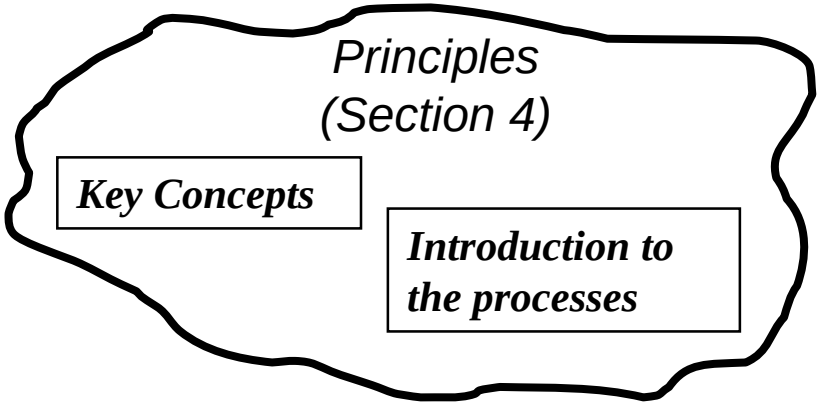
Synchronisation Software – System reviews

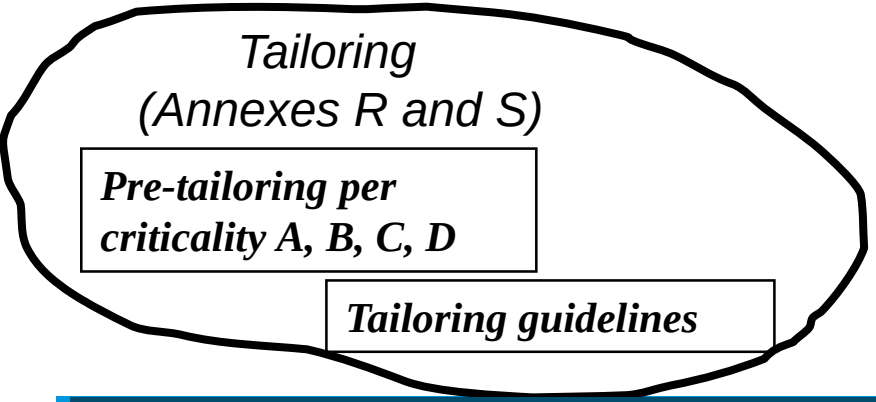
NOTE: this diagram is just there to give a flavour of the review synchronisation; green boxes show when reviews may take place. It is not logically equivalent to the E-40 requirements.

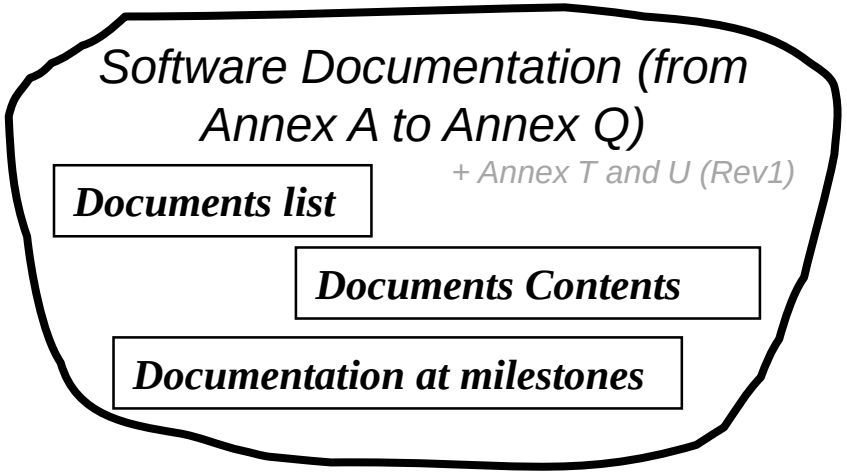


Requirements (Section 5)

Requirements – Section 5







How the Requirements are organised in E-40

1. Each requirement in the Requirements Clause 5 can be identified by a **unique hierarchical number**
2. Sometimes additional text is also provided to further explain the aims of the requirement
3. Requirements are associated with one or more **expected outputs**
4. In this course, there is a single, easily identifiable representation for both requirements and expected outputs

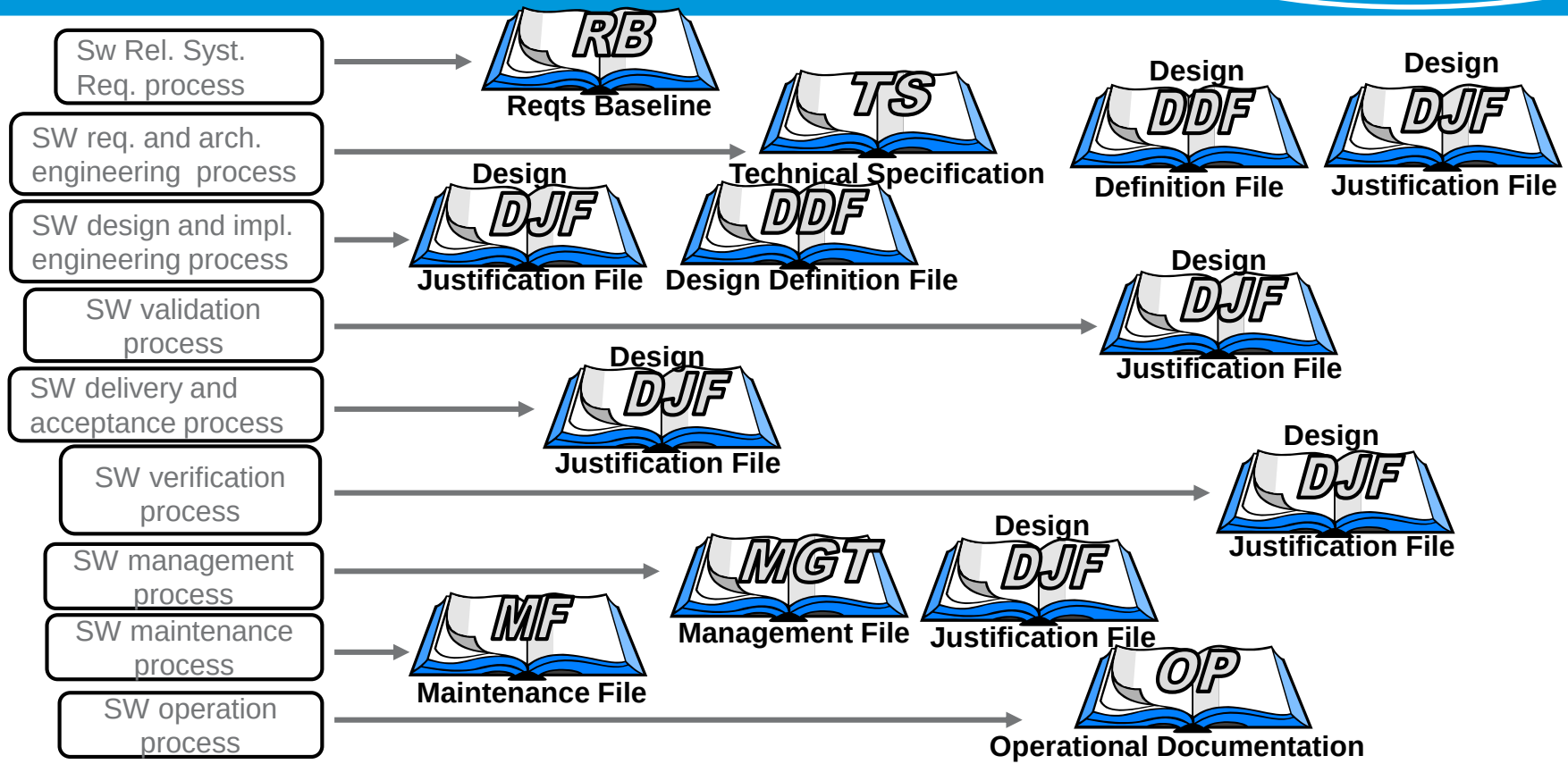
☞ Requirement ☜

Representation of a requirement in this course

🗁 Expected Output 🗁

Representation of an expected output in this course

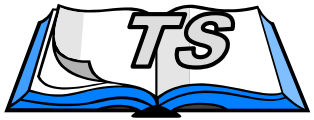
Processes to Files



From Process to File/DRD to Review

SW Req. & Arch. Engineering (5.4)

The process requirements cause information to be collected into files/DRD …



Technical Specification

Software

Requirement Specification

Prelim. Design Review

… which become inputs to reviews.

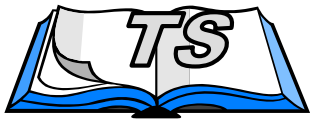
Expected Outputs specify: Files, DRDs and Reviews

☞ Requirement ☜

↓

🗁 Expected Output 🗁

"some information" [*TS, SRS, PDR*]



Technical Specification

Software

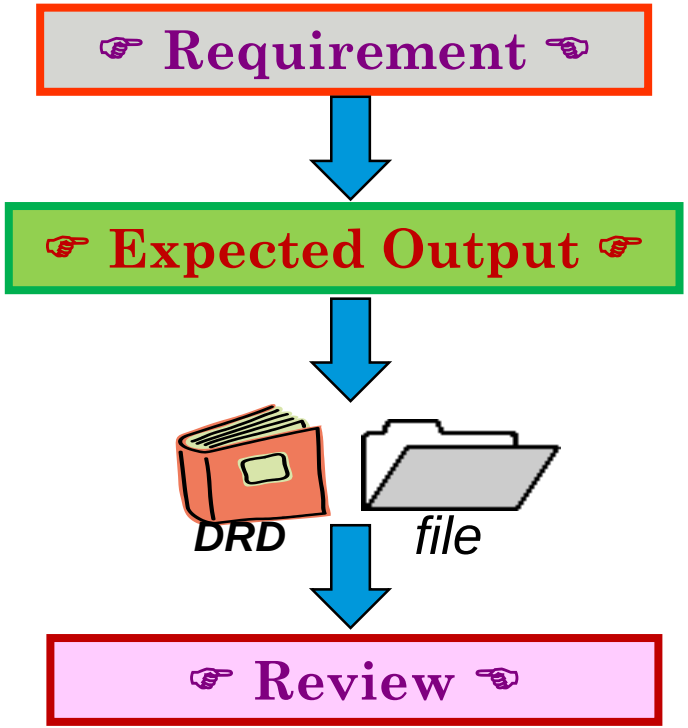
Requirement Specification

Prelim. Design Review

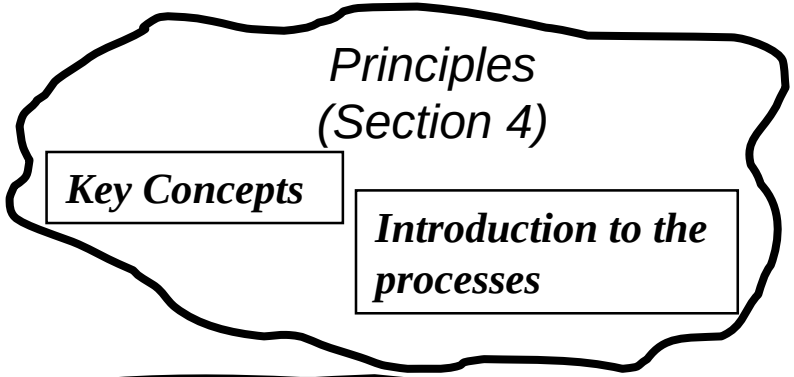
The associated file(s), DRD(s) and review(s) are indicated in square brackets

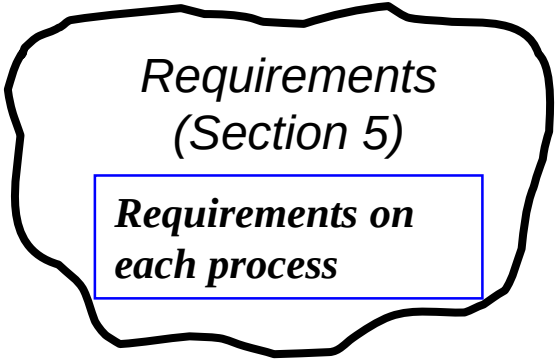
Expected Outputs and DRDs

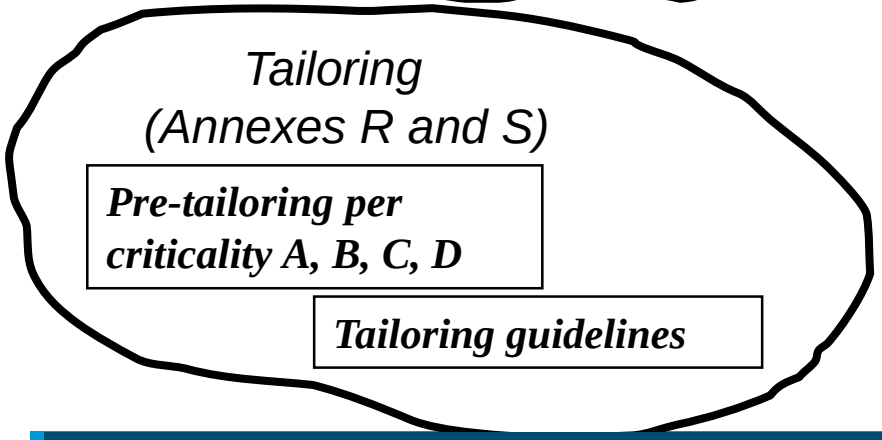
1. The E-40 requirements cause **information** to be placed into **documents** .
2. This information is specified as the **expected output** of the requirement
3. Most documents are specified as **DRD**s [Document Requirement Description]
4. DRDs are made of **sections** and document requirements.
5. The documents are collected into **files** (in the sense of "collection of information") [old ECSS concept]
6. The expected output must be available at a **review**
7. **Annex A** gives the Document Requirement List (DRL), traced to the DRD and to the delivery reviews
8. **Annex Q** gives all the traces for the expected outputs

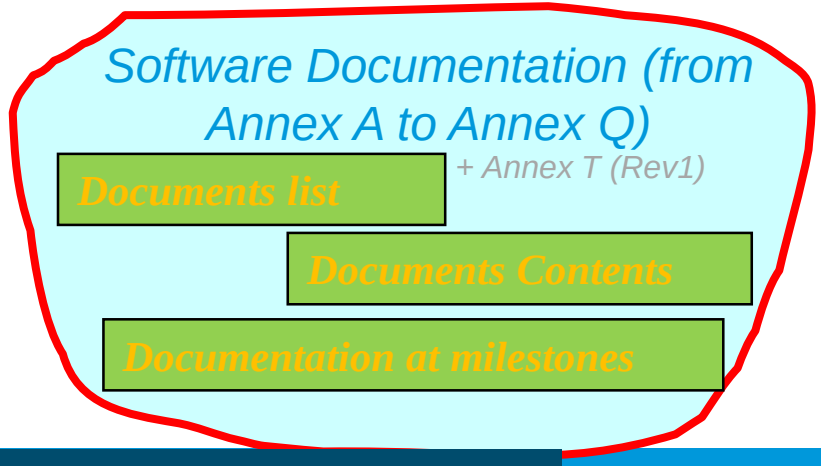


Software Documentation – from Annex A to Annex Q









The DRL

Annex A lists all the documents and their milestone

Related file	DRL item (e.g. Plan, document, file, report, form, matrix)	DRL item having a DRD	SRR	PDR	CDR	QR	AR	ORR
RB	Software system specification (SSS)	ECSS-E-ST-40 Annex B	✓					
	Interface requirements document (IRD)	ECSS-E-ST-40 Annex C	✓					
	Safety and dependability analysis results for lower level suppliers	-	✓					
TS	Software requirements specification (SRS)	ECSS-E-ST-40 Annex D		✓				
	Software interface control document (ICD)	ECSS-E-ST-40 Annex E		✓	✓			
DDF	Software design document (SDD)	ECSS-E-ST-40 Annex F		✓	✓			
	Software configuration file (SCF)	ECSS-M-ST-40 Annex E		✓	✓	✓	✓	✓
	Software release document (SRelD)	ECSS-E-ST-40 Annex G				✓	✓	
	Software user manual (SUM)	ECSS-E-ST-40 Annex H			✓	✓	✓	
	Software source code and media labels	-			✓			
	Software product and media labels	-				✓	✓	✓
	Training material	-				✓		

The DRDs

Annex B to P lists all the Document Requirements Definition

The expected outputs

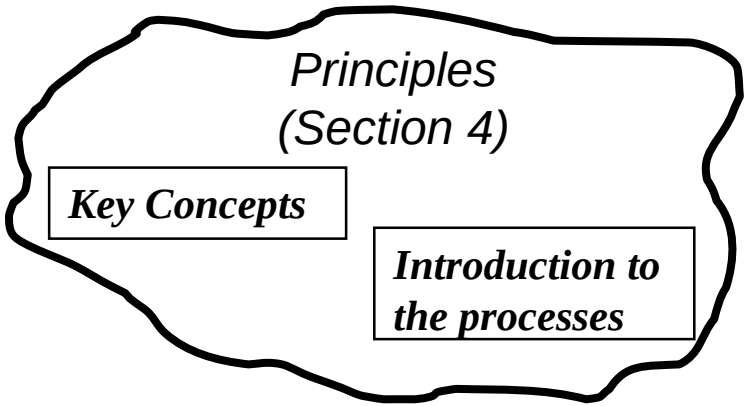
Annex Q gives the traces
Review – DRD – Requirement – DRD section - File

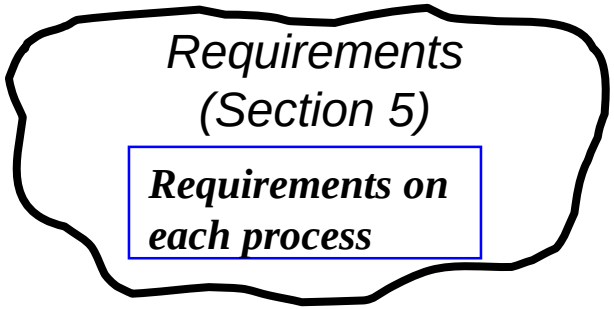
Q.2 SRR

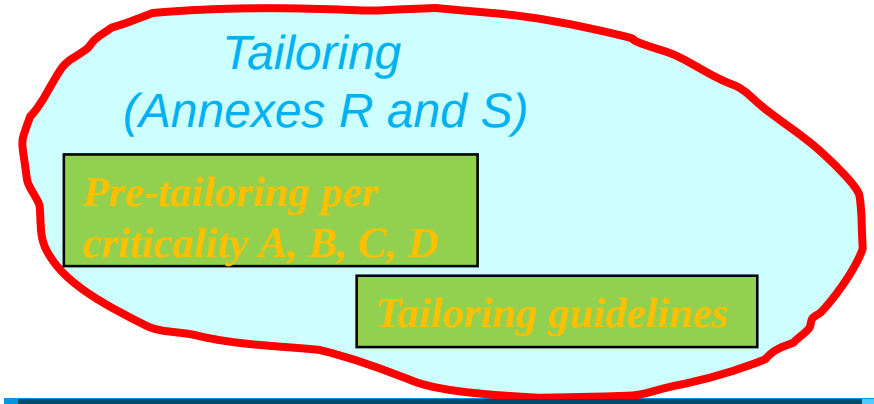
Table Q-1: Documents content at milestone SRR

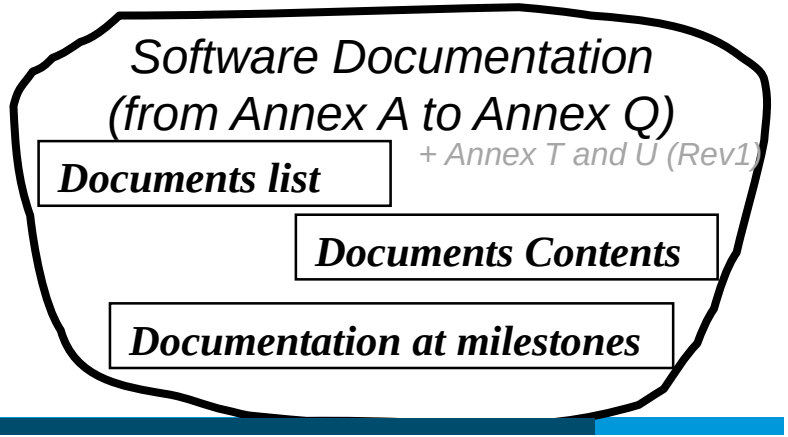
DRD	Requirement	Expected output	Name of expected output	Trace to DRD	File
SSS	5.2.2.1.a	a	Functions and performance system requirements allocated to software	<5.2>	RB
SSS	5.2.2.1.a	b	Verification and validation product requirements	<6.3>, <6.4>	RB
SSS	5.2.2.1.a	c	Software operations requirements	<5.11>	RB
SSS	5.2.2.1.a	d	Software maintenance requirements	<5.12>	RB
SSS	5.2.2.1.a	e	Requirements for in flight modification capabilities	<5.12>	RB
SSS	5.2.2.1.a	f	Requirements for real- time	<5.2>3	RB
SSS	5.2.2.1.a	g	Requirements for security	<5.6>	RB
SSS	5.2.2.1.a	h	Quality requirements	<5.9>	RB
SSS	5.2.2.2.a		System and software observability requirements	<5.13>	RB
SSS	5.2.2.3.a		HMI requirements	<5.2>	RB

Tailoring - Annexes R and S









Tailoring

1. The ECSS family of standards has been designed to **minimize** the need for **tailoring** for each project.
    a. This is a fundamental underlying concept throughout the system
2. The ECSS-E-40 standard lists **exhaustively** the requirements for the **best practices** in space software engineering
    a. that is, it covers *all possible types* of space software engineering projects
3. But it is pre-tailored **according to the criticality levels** as defined in the ECSS-Q-ST-80 (see Annex R: normative).
    a. A way to apply the standard in the most efficient manner possible
4. Further tailoring guidelines are provided based on programmatic and technical factors (**Annex S: informative**).
5. Customer makes the tailoring, or delegate to Supplier (risk!)

E-40
Software

E-40
Annex R, S

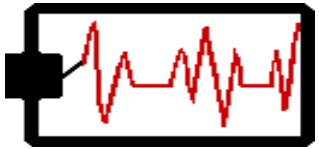
Some Requirements Concern Specific Types of SW

1. Clause 5 of E-40 is entitled simply **"Requirements"** because, as a rule, the requirements may be considered to be applicable to *any* space software
2. In practice, however, some requirements in Clause 5 depend on the **nature** of the software
3. Other requirements depends on a **particular** project **context** (e.g. if software needs to be migrated)
4. This is always **explicitly mentioned in the requirements**
5. Their list is summarized in **Annex S**

Flight software

HMI software

Annex S.2 indicates all the conditional requirements



Real-time SW

SW reuse

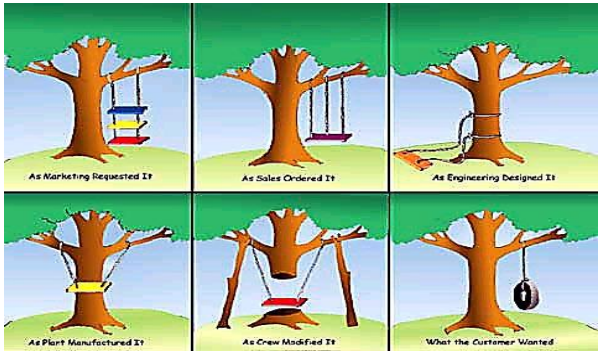
What are the E-40 Platinum Requirements?

They are all Platinum requirements

a. An army of reviewers has contributed to produce this third version of the E-40
b. There is a good reason why each of them is there!

Tailoring is further discussed in E-40 Handbook (4.2.5)

1. Non tailorable topics:
   a. specifying **requirements**
   b. validating software
   c. agreeing on a **development approach**
   d. managing the **configuration**



2. Tailoring build-in the Standard
   a. Per criticality A, B, C, D
   b. According to contract scope & context (Annex S.2 – S.3)

Further Tailoring is a RISK!

a. merging the requirement baseline and the technical specification increases the risk to loose the customer or supplier standpoint, i.e. to miss a use case or to miss an implementation requirement.

b. skipping joint reviews increases the risk to discover late disagreements between customer and supplier on the product capability or quality, causing substantial reengineering

c. non managing technical budgets and margins increases the risk to discover unfeasibility late in the project

d. not using design methods increases the risk to develop weak architectures and inconsistent designs

e. not defining interface increases the risk of integration issues

f. not documenting the detailed design increases the risk to loose control on the software development such as capability to anticipate implementation errors, to debug, to integrate, to maintain, to master safety and dependability, etc.

g. skipping unit tests increases the risk to discover bugs late in the process and to jeopardize the schedule.

h. not rerunning the full validation tests on the last version of the software increases the risk to leave bugs in the product

i. not performing full verification activities increases the risk to affect the quality of the product

j. non complying with DRDs content increases the risk of having non complete documentation, and of missing information for maintenance

k. non compliance with the DRDs structure increases the effort of the reviewers

➔ Risk Management

Summary of Part 2

Summary of Part 2

1. Software is a subsystem in the overall system
2. Explicit & recursive customer-supplier (or system-software) relationship
3. Process model with reviews
4. Requirements -> Expected outputs → files → documents -> reviews
    a. Annex A of E-40 lists the required information of each file at specific reviews
    b. For some documents, DRD is provided specifying their contents (most of them in E-40, but also in M-40 and Q-80)

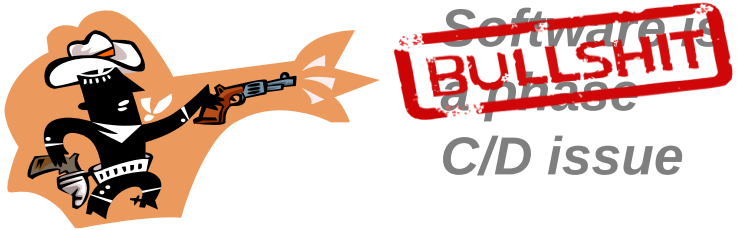
5. Tailoring is a fundamental aspect of using the ECSS standards in an efficient way
    a. Help the Customer to prepare the project Invitation to Tender
    b. Help the Supplier to bid and execute the project
6. Annexes R and S are the primary source of information on the proper approach to tailoring the E-40 requirements:
    a. Characterize the project
    b. Evaluate each requirement

Conclusion of the E-40 training

E-40 Training Conclusion

1. Software is a subsystem in its own rights

1. Its activities start in Phase B
    a. E-40 Clause 5.2, Software related system requirement process



2. **Requirements (in particular system/customer requirements) are important**

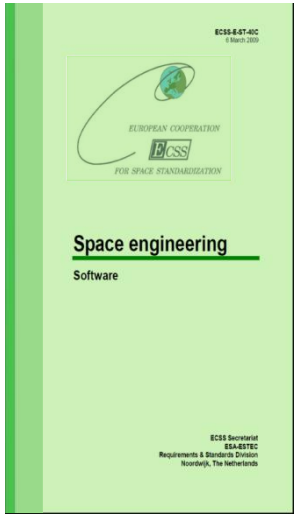
3. Understand the standard to better tailor it.

4. E-40 is applied together with Q-80

ECSS-E-ST-40C Rev1

- 68 Change Requests have been processed.
- Main updates:
  - Revisit of definitions, including new definition of “software”
  - Introduction of security aspects
  - Make the standard less waterfall
  - Reinforce the verification of code
  - Ensure consistency with ECSS-E-ST-20-40C standard to be published
  - Consider the outcomes of ISVV guidelines update
  - Introduce the Software Delivery Review Board and Software validation specification review
  - Introduce the Software Validation Control

Questions



Credits:
- Jean-Loup.Terraillon@esa.int
- Christophe Honvault@esa.int
- http://www.intecs.it/