Requirements Module: Writing Requirements

Space Systems Engineering, version 1.0

Module Purpose: Writing Requirements

- Define and understand the characteristics of and rules for writing good requirements.
- Understand the value of providing the rationale and the preliminary verification technique with each requirement.
- Define and establish the difference between requirements verification and requirements validation.
- Establish the importance of resolving undefined (To Be Defined — TBD) and estimated (To Be Resolved — TBR) requirements early.

Related Readings:

Writing Good Requirements, INCOSE 1993, Ivy Hooks.

NASA Systems Engineering Handbook 2007, Appendix C, How to Write a Good Requirement

*Good Requirements Are **SMART***

- **S**pecific -
  - It must address only one aspect of the system design or performance
  - It must be expressed in terms of the need (what and how well), not the solution (how).
- **M**easurable -
  - Performance is expressed objectively and quantitatively
  - E.g., an exact pointing requirement (in degrees) can be tested thus verified prior to launch.
- **A**chievable -
  - It must be technically achievable at costs considered affordable
  - E.g., JWST early designs specified an aperture requirement eventually descoped due to technical issues with deployment.
- **R**elevant -
  - It must be appropriate for the level being specified
  - E.g., requirement on the solar cells should not be designated at the spacecraft level.
- **T**raceable -
  - Lower level requirements (children) must clearly flow from and support higher level requirements (parents).
  - Requirements without a parent are referred to as orphans, and need to be assessed for necessity of inclusion.

Rules for Writing Good Requirements

- Requirements have mandatory characteristics:
  - *Needed*
  - *Verifiable*
  - *Attainable: technically, cost, schedule*
- Each requirement should
  - *Express one thought*
  - *Be concise and simple*
  - *Be stated positively*
  - *Be grammatically correct; free of typos and misspellings*
  - *Be understood only one way; they are unambiguous*
  - *Use consistent terminology to refer to the system/product and its lower level entities*
  - *Comply with the project's template and style rules*

More Rules for Writing Good Requirements

What a requirement must state:

- **WHO** is responsible
- **WHAT** shall be done
- Or **HOW WELL** something shall be done
- Or under what **CONSTRAINTS** something shall be done

Requirement format: **"WHO shall WHAT"**

- Uses active not passive voice

Example product requirements:

- The system shall operate at a power level of…
- The software shall acquire data from the…
- The structure shall withstand loads of…
- The hardware shall have a mass of…

Use the correct terms:

- *Requirements* are binding - **Shall**
- *Facts* or Declaration of purpose - **Will**
- *Goals* are non-mandatory provisions - **Should**

- Do NOT use "Must"

Goodness Checklist: Is this Requirement…

- Free of ambiguous terms?
  - Examples: as appropriate, etc., and/or, support, but not limited to, be able to, be capable of
- Free of indefinite pronouns?
  - Examples: this, these
- Free of unverifiable terms?
  - *Examples: flexible, user-friendly, robust, light-weight, maximize, adequate, small, portable, easily - other "ly" words and other "ize" words*
- *Free of implementation?*
  - *Requirement should state WHAT is needed, NOT HOW to provide it, i.e., state the problem not the solution.*
- *Necessary?*
  - *Ask "Why do you need the requirement?"; the answer may lead you to the real requirement.*
- *Free of descriptions of operations?*
  - *To distinguish between operations and requirements, ask "Does the developer have control over this?" "Is this a need the product must satisfy or an activity involving the product?"*
- *Free of TBDs (To Be Determined)?*
  - *Use a best estimate and a TBR (To Be Resolved) with rationale when possible.*

Example Requirements: Good or Bad?

- **The aircraft shall have three engines (initial DC-3 requirement).**
  - The aircraft shall meet the operation requirements with a single engine out.
- **The lunar lander shall include an airlock.**
  - The lunar lander shall provide the capability for crew to ingress/egress while maintaining pressurization.
- **The crew shall have the capability to perform extra-vehicular activities (EVAs).**
  - The vehicle shall allow extra-vehicular activities during operations.
- **The spacecraft shall maximize lifetime.**
  - The spacecraft shall have a lifetime of at least three years.
- **The software shall display data in a user-friendly fashion.**
  - The software shall display data as described in ICD 2345 Table 3.1.

Rationale Captures the Motivation and Assumptions of a Requirement

The rationale of each requirement defines

- Why a requirement is needed
- What assumptions were made
- What design effort drove the requirement
- Other data that will be needed to maintain the requirement over time

Example

- Requirement: "The truck shall have a height of no more than 14 feet."
- Rationale: 99% of all US interstate highway overpasses have a 14 foot or greater clearance. (Assumptions: The truck will be used primarily on US interstate highways for long-haul freight in the US.)

Space Example Requirement & Rationale

- Requirement: “The Constellation Architecture shall provide communication and tracking services prior to launch, during all mission phases, through recovery.”
- Rationale: Communication and tracking services must be provided to elements during all mission phases. Ground based assets can provide this capability to elements when in direct line-of-sight with Earth. The Space Network can support launch, operations in LEO, re-entry and landing. Earth-based comm. and tracking services cannot, however, support the lunar landing, lunar ascent, or lunar surface operations if they are at polar locations or on the far side of the Moon. Another infrastructure element, such as a lunar relay, must be provided to support communications and tracking to elements during operations on the lunar far side or at the poles.
  - This requirement does not imply continuous coverage.
  - Lower level requirements will specify continuity and type of coverage.

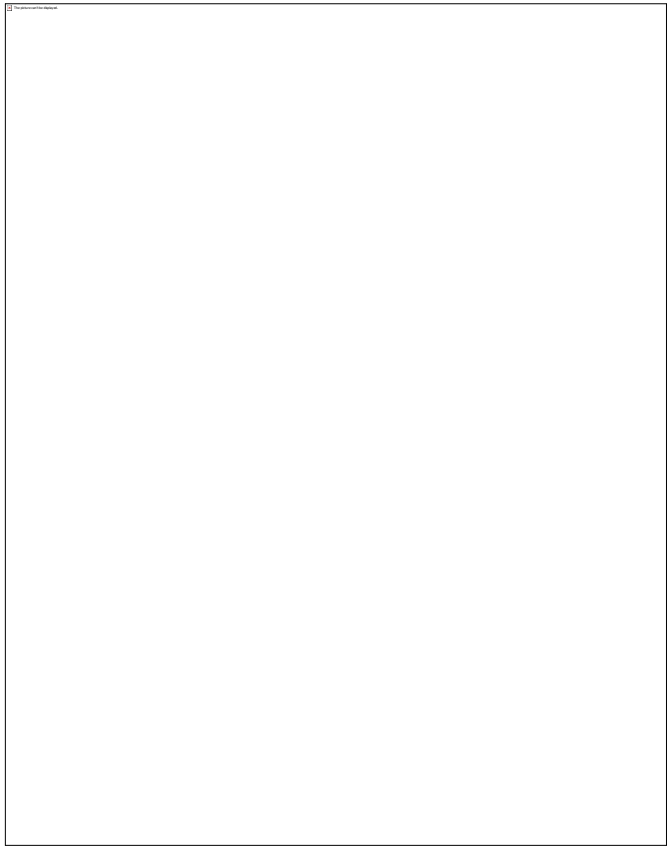
Early Consideration of Requirement Verification Helps Avoid Costly Problems

- The systems engineer should create a preliminary verification plan as each requirement is written. Typically this plan is no more than establishing the technique (test, demonstration, analysis or inspection) that will be used for verifying a requirement.

- This early consideration of requirement verification helps:
  - Confirm the requirement is indeed verifiable
  - Define the system verification plan
  - Identify needed facilities for subsystem and system test

Preliminary Verification Plans Help Eliminate Impossible or Impractical Situations

- The Magellan synthetic aperture radar had dozens of command variables each with hundreds of possible values and 8 redundant, cross-strapped subsystems. The number of hardware and command options made it impossible to test every valid radar command with every configuration.
- The solution was to test 4 different hardware configurations with a small subset of stressing command variables at the extreme operating temperatures.
- This helped establish confidence that the radar system would work with any valid command and hardware configuration while in orbit around Venus.
- Magellan met or exceeded all of its mission requirements and mapped Venus from 1989 until its termination in 1994.



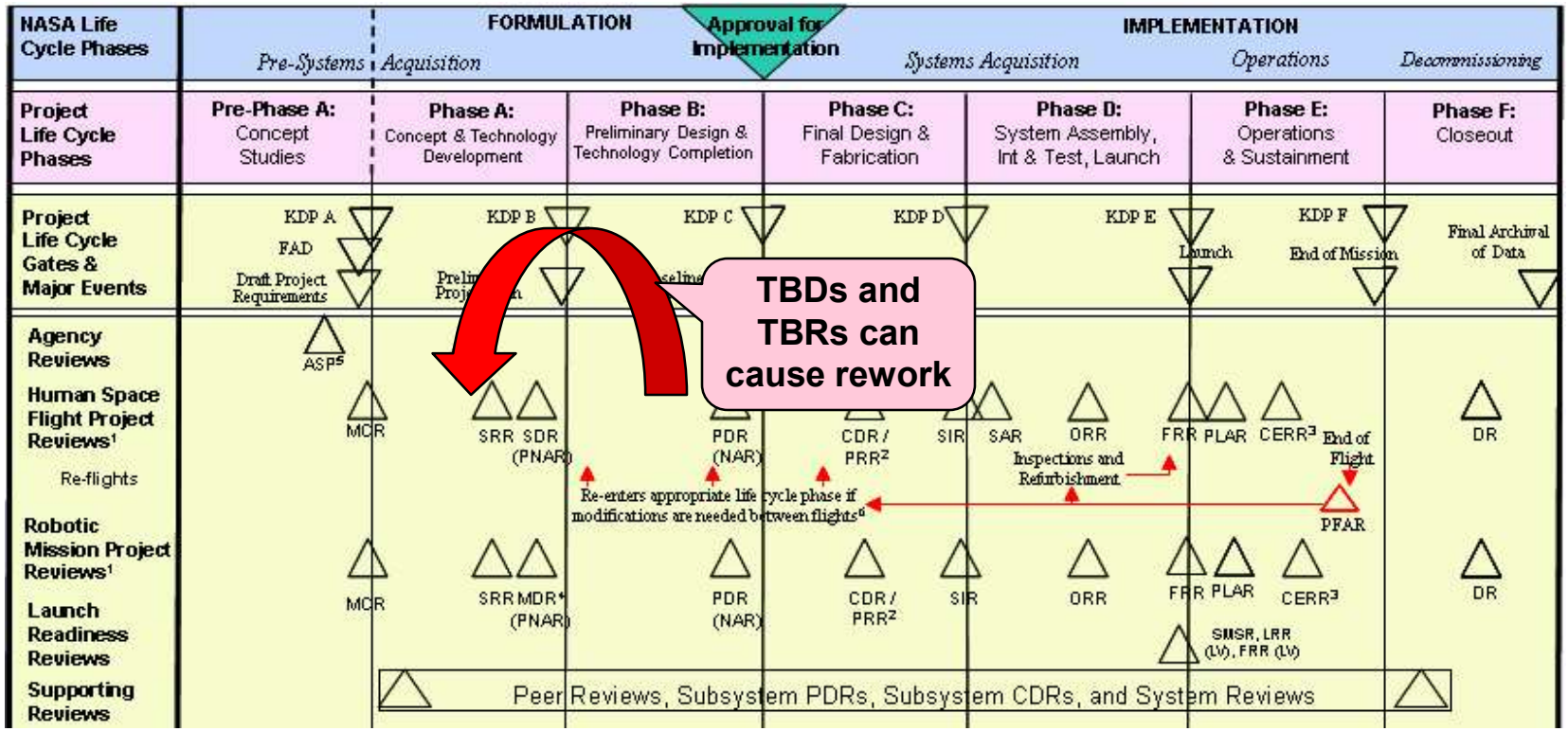
Templates Help Capture Verification Technique, Rationale, Significance and Traceability

Requirement Definition Card					
Requirement ID#:	*Unique alphanumeric*		Requirement Origin (Spec, Conop, etc.):	*Pointer to origin*	
Requirement:	*Requirement statement*				
Source:	*Author*		Date:	*Origination Date*	
Verification:	*How will the requirement be tested? One or two sentences.*				
Customer Satisfaction:	*(1-5) 5=essential to have, 1=indifferent to capability*		Customer Dissatisfaction:	*(1-5) 5=very dissatisfied if not implemented, 1=not concerned if not implemented*	
Estimated Implementation Cost:	(1-5) 5=expensive to build, 1=inexpensive to implement		Estimated Implementation Risk:	*(1-5) 5= very high risk to implement, 1=no risk, 2= low risk*	
Related Requirement(s):	*Identifier*		Conflicting Requirement(s):	*Identifier*	
Rational:	*Pointer to info*		History:	*Origin, changes, or deletion (as applicable)*	
Customer Approval:	*Name*	*Date*	Provider Approval:	*Name*	*Date*

Requirements Validation

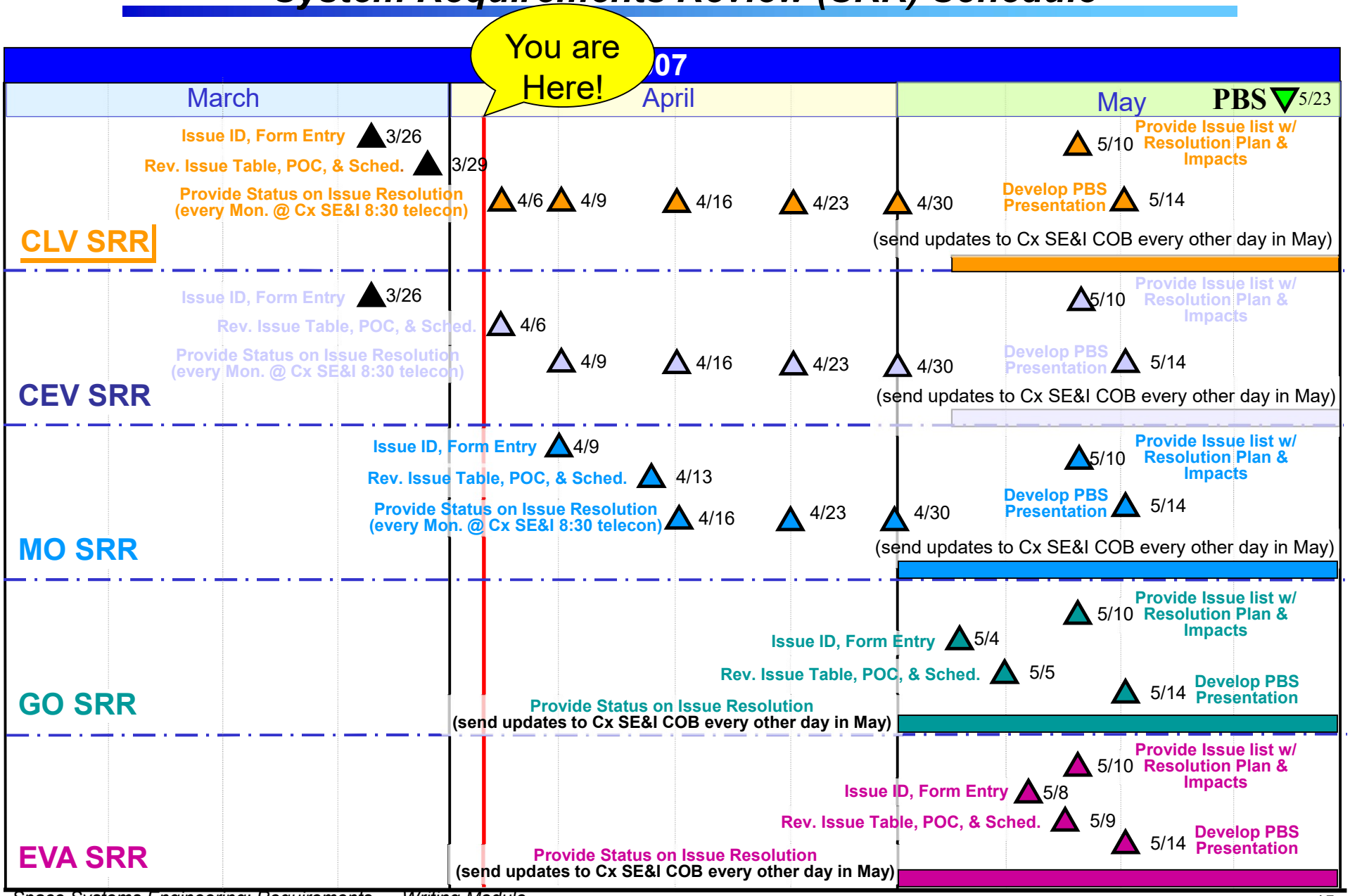
- Requirements validation asks three questions:
  - *Do we have the correct problem?*
  - *Do our requirements capture this problem?*
  - *Are our requirements SMART?*
- Performed to *assure* the system requirements *set* is complete, consistent, and each requirement is achievable and verifiable.
- Performed by subject matter experts, the system performing organization and the system authorized customer.
- Does the requirement set completely address and accomplish the mission?
- Are the requirements consistent with the established system boundaries and constraints?
- When does requirement validation take place?
  - *Usually before the System Requirements Review (SRR)*

Resolve TBDs and TBRs Early in the Project Life Cycle



- **TBDs and TBRs identify missing or uncertain information.**
- **When resolved they may cause rework of items previously completed.**
- **Assign responsibility for all TBDs and TBRs.**
- **Resolve as soon as possible.**

Overall Constellation Program
System Requirements Review (SRR) Schedule



Constellation Program Status (spring '07) (Closure of SRR)

Board AI

- Total AI Count: 48 Total
- 9 Closed
- 39 Open
  - 30 are "Past Due"
  - 9 are in work, "Not Due Yet"

RIDS

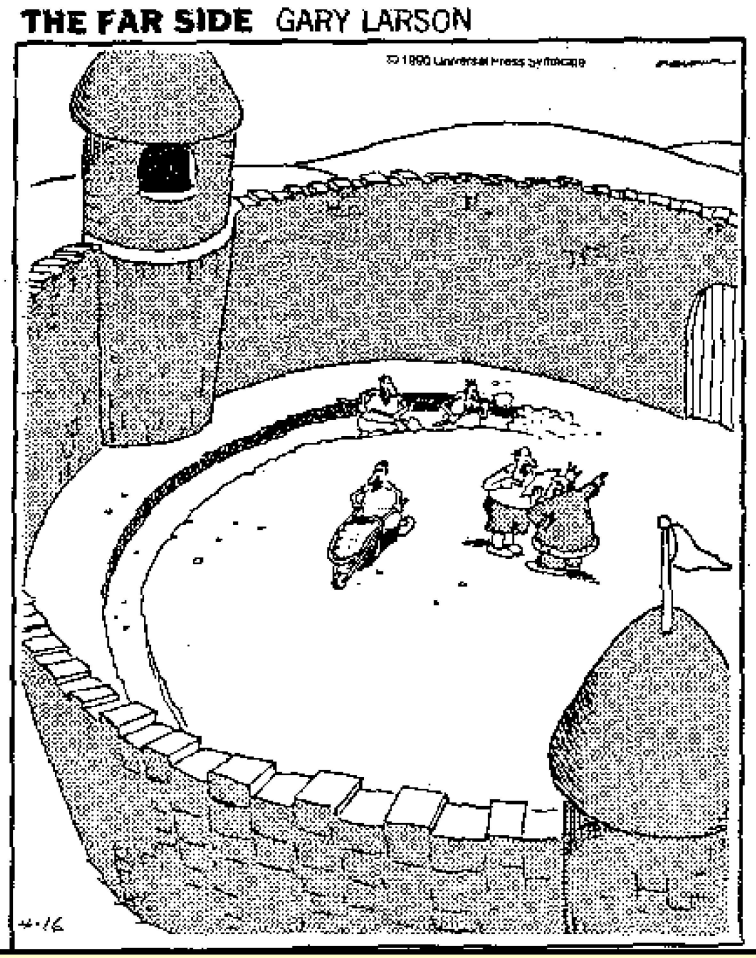
- Total RID Count: 6,283
- 6,225 Closed
- 58 Open
  - Majority closed prior to PBS
  - All RIDs should be closed prior to CxP SDR

TBD/TBR

- Total Count: 2,532
  - Received plans for 2,064 TBRs/TBDs in 72 of 95 documents (76% of docs.)
  - Need burn-down plans for 196 TBRs/TBDs in 16 documents (17% of docs.)
  - 7 documents (274 TBRs/TBDs) will not be updated/baselined until post CxP PBS. These TBRs/TBDs will not be closed prior to CxP PBS

Document	Document Title	Being Worked
CxP 70008	Master Integration and Verification Plan (MIVP)	4
CxP 70023	CxP Prg Design Spec Natural Environments (DSNE)	9
CxP 70024	CxP HSIR	1
CxP 70036	Cx Environ Qual & Accept Testing Reqs (CEQATR)	5
CxP 70050 Vol 2	Electrical Power System Spec Vol 2	1
CxP 70061	C3I Strategic Plan	1
CxP 70067 Vol 1	CxP Program Human-rating Plan Vol 1	19
CxP 70080	Electromagnetic Environmental Effects (E3) Req'ts	6
CxP 70086	Software Verification and Validiation Plan	5
CxP 70135	Structural Design & Verification Requirements	7
Total		**58**

User Participation and Communication are Key



Suddenly, a heated exchange took place between the King and the moat contractor.

Pause and Learn Opportunity

Discuss James Webb Space Telescope (JWST) Requirements Examples

JWST Science Requirements Document (.pdf)

Chapter 4 Table 4.1 Verification Matrix

Chapter 7.0 Rationale Chapter

Chapter 8.0 Actual Science Requirements Chapter

Appendix B Traceability Matrix

Module Summary: Writing Requirements

- Good requirements Are SMART: **S**pecific, **M**easurable, **A**chievable, **R**elevant and **T**raceable
- It is good practice to capture the rationale and preliminary technique for verification when writing requirements.
- Requirement validation is a process of ensuring that: the *set* of requirements is correct, complete, and consistent.
- Since the cost of reconciling undefined requirements grows as the project matures, undefined (TBD) and estimated (TBR) requirements should be resolved as early as possible.

Backup Slides
for Requirements — Writing Module

Work With the Customer to Distinguish Between 'Desirements' and Requirements

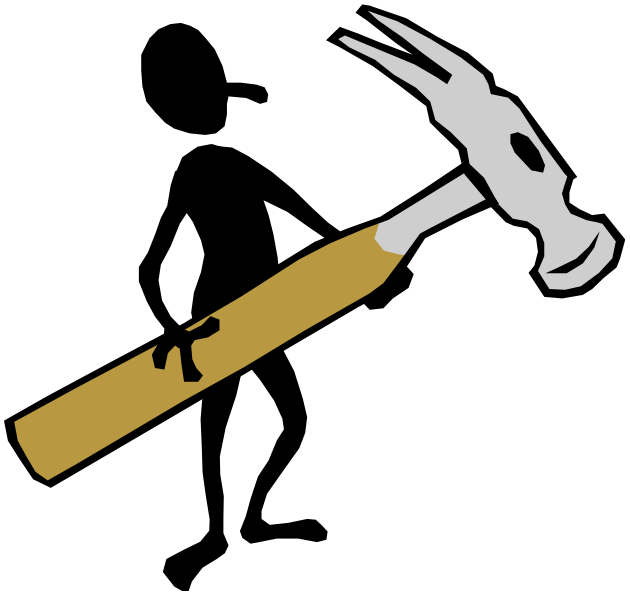
- *Desirement* – something that would be nice to have but is not mandatory to accomplish the mission
- Requirement - something that must be done for the mission to be successful
- Customers and stakeholders often blur the distinction between what is necessary and what would be nice to have

- Develop the problem statement independently of the solution
- Customers, stakeholders, design engineers and even systems engineers often have solution biases – Desirement space
- *Be Careful* - Customers or stakeholders may have a good idea, but early implementation solutions may prove impossible, impractical or sub-optimal when looked at in detail - capture what is necessary for mission success and *then* develop solutions to meet those needs.

Requirements Validation

- Requirement Validation is the process of confirming the completeness, compatibility and correctness of the requirements.
- Requirement Validation answers the questions: " Are the system design requirements correctly defined and mean what we intended? "Is the set of requirements, or the specification, self-consistent?"
- When does requirement validation take place?
  - Before design and during detailed design, i.e., mostly in Phase B and tapering down in Phase C
  - Ideally, completed prior to System Requirements Review (SRR)
- What is importance of getting requirements validation right early in the project life cycle?
  - So when it comes time to verify the system, you are verifying to the right requirements.

Automated Tools



- There are tools that provide requirements management capabilities and also offer executable models
- Tools such as CORE, DOORS, CRADLE, SLATE, Popkin, etc., provide feature rich interactive analysis
- The INCOSE web site www.INCOSE.org gives access to an INCOSE Tool Survey
  - Fifteen requirements management and ten systems architecture tools are listed

Requirements Verification and Requirements Validation

- Requirement verification is establishing confidence that the requirement has been met.
  - Requirements are verified by test, demonstration, analysis or inspection.
  - Test is the most common method of verification and the technique that provides the most confidence that the system will indeed work as intended in its anticipated environment.
- Requirement validation is a process of ensuring that :
  1. the *set* of requirements is correct, complete, and consistent,
  2. a model can be created that satisfies the requirements, and
  3. a real-world solution can be built and tested to prove that it satisfies the requirements.
- If a systems engineer discovers that the customer has required a perpetual-motion machine, the project should be stopped.