

How to write good requirements (Module 10 of 10) **Summary and closeout**

Version 1.1.3

How to write good requirements

1001-1



Course Module topics

1. Introduction to requirements
2. Stakeholders and their importance
3. Identifying the stakeholders' wants
4. Converting stakeholder wants to needs
5. Documenting stakeholders' needs
6. Converting stakeholder needs to requirements
7. Converting requirements to well-written requirements
8. Converting well-written requirements to good requirements
9. The use of requirements in the rest of the system development process
- 10. Summary and closeout**

How to write good requirements

1001-2



Objectives of Module 10

1. To review the course Modules and the learning
2. To present an overall summary of the course construction
3. To request overall feedback via the Course Evaluate Form (1002)
4. To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy

How to write good requirements

1001-3



Knowledge components

- Lecture
 - Sets the context and provides overview
- Readings
 - None
- Exercises
 - 10-1 Course evaluation form
 - 10-2 Self evaluation

How to write good requirements

1001-4

A cause of project failures

- Tom DeMarco conducted an annual survey of real world development projects between 1977 and 1987*
- Over 500 project histories
- Reported that 15% of all projects studied came to naught
 - they were canceled, aborted, "postponed" or delivered products that were never used
- Fully 25% of projects that lasted 25 person-years or longer failed to complete
 - In the majority of projects *there was not a single technological issue to explain the failure*
- **The cause of failure most frequently cited were people related, including:**
 - staffing problems
 - disenchantment with management or the client
 - lack of motivation and high turnover
 - communications (inter-personal) problems

* DeMarco, Tom, and Lister, Timothy, Peopleware, Dorset House Publishing Company, 1987.

Creating Outstanding Systems Engineers

3-5

Module 10 topics

- **Summary and highlights of Modules 0-9**
- Overall summary of course
- Module 10 exercises



How to write good requirements

1001-6

Disclaimer: content of Modules may be changed at any time without prior notification

Writing good requirements

Module 0: Introduction and overview

Dr Joseph E Kasser, DSc, CEng, CM, FIET, FIES, FINCOSE (1995-2018),
CMALT (2008-2021), G3ZCZ, VK5WU

Email (for the course): beyondsystemsthinking@yahoo.com

<https://therightrequirement.com>

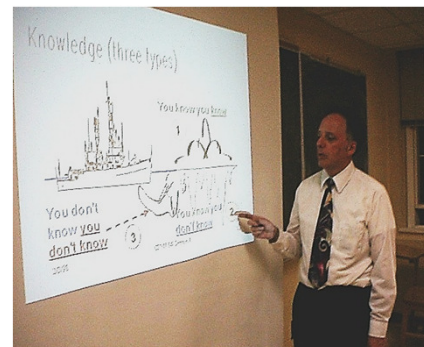


Writing good requirements

0001-7

Course assumptions

- Lectures summarize and point out important points
- Practical activities bring knowledge to life and provide experiential anchor points
- Participants
 - May interrupt lecture with questions
 - Expected to do homework
 - Exercises and readings
 - Maximize return on investment



Writing good requirements

0001-8

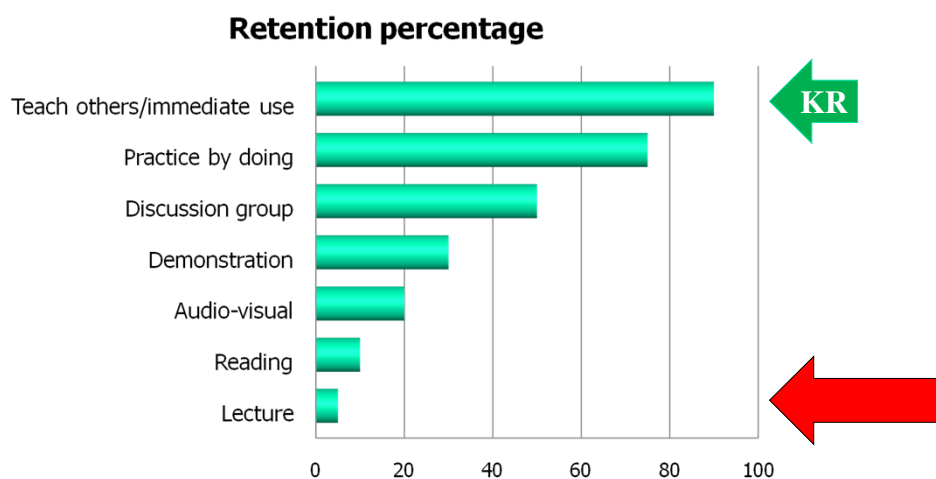
Course architecture

- Optimized for maximum learning using the balanced classroom
- Designed for various learning styles
- ~~Live Recorded lectures~~
- Live weekly discussion and question and answer Modules
- Readings for in-depth knowledge
- Practical exercises
- Knowledge reading exercises
- Lots of feedback to keep learning on track

Writing good requirements

0001-9

Quantitative perspective: Retention rate after 2 weeks



Redrawn Dales' cone and Learning Pyramid

How to write good requirements

1001-10

Desirable situation (FCFDS)

- Providing the five top aspects of the engineering design process that best equip secondary students to understand, manage, and solve technological problems (Wicklein, et al., 2009):
 1. Multiple solutions to a problem/requirement
 2. Oral communications
 3. Graphical/pictorial communication
 4. Ability to handle open-ended/ill-defined problems
 5. Systems thinking
- Grading based on cognitive skills and knowledge
 - Incorporating higher levels of Bloom's taxonomy
- Going beyond systems thinking to holistic thinking
 - Systems thinking provides understanding
 - Holistic thinking identifies problems and provides solutions

How to write good requirements

1001-11

Solution situation

No classes on these

```

graph TD
    CM[Classroom Module] --> L[Lecture]
    CM --> E[Exercises]
    CM --> KR[Knowledge readings]
    CM --> IA[In-class assignment]
    style KR stroke:#f00,stroke-width:2px
    style IA stroke:#f00,stroke-width:2px
  
```

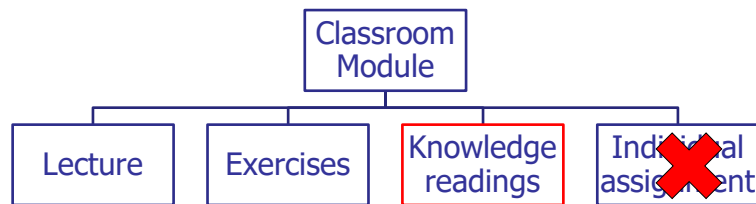
5 top aspects (Wicklein, et al.) (student's perspective)	Lecture	Exercises	Knowledge readings
Multiple solutions to a problem/requirement	Listened	Experienced	Experienced additional examples
Oral communications	-	Experienced	Experienced
Graphical/pictorial communications	Received	Experienced	Experienced
Ability to handle open-ended/ill-defined problems	-	Experienced	-
Systems thinking	Listened	Went beyond	Went well beyond

Knowledge readings provide additional and extra opportunities

How to write good requirements

1001-12

Balanced classroom



	Bloom's taxonomy	Lecture	Exercises	Knowledge readings
6	Creating			✓
5	Evaluating			✓
4	Analyzing			✓
3	Applying		✓	
2	Understanding	Unknown	✓	✓
1	Remembering	Listened	✓	✓

Writing good requirements

0001-13

Knowledge readings

- Provides a better learning experience
 - learning for the purposes of presentation is the best way of ensuring retention and understanding of the knowledge
- Demonstrates that different people perceive information differently
- Enables the instructor to correct any misinterpretations as they arise
- Provides you with the opportunity to
 - practice presentations skills
 - obtain feedback of content and style
- Very positive student feedback



Updated Bloom's taxonomy

Writing good requirements

0001-14



Exercises

1. Sized to take no more than 60-90 minutes
 1. Watch video/reading 0003
2. Need not be 100% complete to present
3. Two types
 1. Practical
 - Apply knowledge from Module and previous Modules
 2. Knowledge readings
 - Learning by teaching
 - Feedback on interpretation
 - Discussed in optional reading/video 0002

Writing good requirements

0001-15



How to deal with the exercises

- Use the COPS Problem Formulation Template
- Determine what needs to be done (requirements)
 - Work BACK from the answer!!!!!!!!!!
 - Systems Thinker's Toolbox, Section 11.8
- **Create a compliance matrix**
 - Systems thinker's toolbox Section 9.5.2 and Module 1
- Create a presentation template
 - Systems thinker's toolbox Section 14.6 provides one
- Plan the use of allotted time
 - Figure out how much time to allocate to each part of the exercise
- Think about observations and insights from readings and prior knowledge
- Produce the required presentation
- Incorporate material to show you have looked at the readings or equivalent material
 - Citations (author, date)

Writing good requirements

0001-16



COPS Problem Formulation Template

1. *The undesirable situation*

- As perceived from the HTPs (objects and relationships)

2. *Assumptions*

- About the situation, problem, solution, constraints etc.

3. *The Feasible Conceptual Future Desirable Situation (FCFDS)*

- As perceived from the HTPs

4. *The problem*

- **What** needs to be done to convert the FCFDS to reality in **reverse order**

5. *The solution*

- **How** the undesirable situation will be/was remedied in forward order

Writing good requirements

0001-17




Two major mistakes students make

- Mistakes and consequences
 1. Not answering questions or parts of questions that were asked
 - No points
 2. Answering questions or parts of questions that were not asked
 1. No points and waste of time
 2. Examiner can miss the answer while sieving through the extraneous information leads to reduced points
- Two tools to eliminate mistakes
 1. Compliance matrix
 2. Answer template

How to write good requirements

1001-18



How to write good requirements

Module 1 of 10

Introduction to requirements

Version 1.1.4

How to write good requirements

0101-19



Module 1 objectives

#	Objective	Met
1	To provide the background to requirements	9-21
2	To provide definitions of the terminology used in the course	23-29
3	To explain the purposes of requirements	31
4	To explain some of the problems caused by poorly-written requirements	33-34
5	To explain the difference between requirements, well-written requirements and good requirements	25
6	To explain what constitutes a well-written requirement	36-55
7	To explain generic and system specific requirements and the benefits of the distinction	57-60
8	To show some examples of bad requirements and how to fix them	62-70
9	To explain why they are bad	62-70
10	To explain what needs to be done to convert them to well-written requirements	62-70
11	To provide the opportunity to obtain 5 levels of knowledge updated Blooms taxonomy	72-74

How to write good requirements

0101-20

Requirements (What)*

- Three types of requirements
 1. **Functional requirements** - concern a result or behavior that shall be provided by a function of a system. This includes requirements for data or the interaction of a system with its environment.
 2. **Quality requirements** - pertain to quality concerns that are not covered by functional requirements, such as performance, availability, security, or reliability.
 3. **Constraints** - are requirements that limit the solution space beyond what is necessary to meet the given functional requirements and quality requirements.
- **Comments**
 - Why add "includes" to functional requirement?
 - Quality is conformance to specifications (Crosby, "Quality is free", 1979)
 - Examples given are known as non-functional requirements

* International Requirements Engineering Board (IREB) Certified Professional for Requirements Engineering Foundation Level Syllabus, Version 3.1.0, September 1st 2022, EU 1.1

How to write good requirements

0101-21

Requirements (Where)

- Requirements can occur as*
 - **System requirements** – what a system will do
 - **Stakeholder requirements** – what stakeholders want from their perspective
 - **User requirements** – what users want from their perspective
 - **Domain requirements** – required domain properties
 - **Business requirements** – business goals, objectives, and needs of an organization
- **Comments**
 - Focuses on wants, not needs
 - Users are not stakeholders in this section of the IREB document
 - "Requirements Engineering is about satisfying the stakeholders' desires and needs" (EU 2.1)
 - So user requirements are not satisfied by requirements engineering

Synonym?

???

* IREB, EU 1.3

How to write good requirements

0101-22

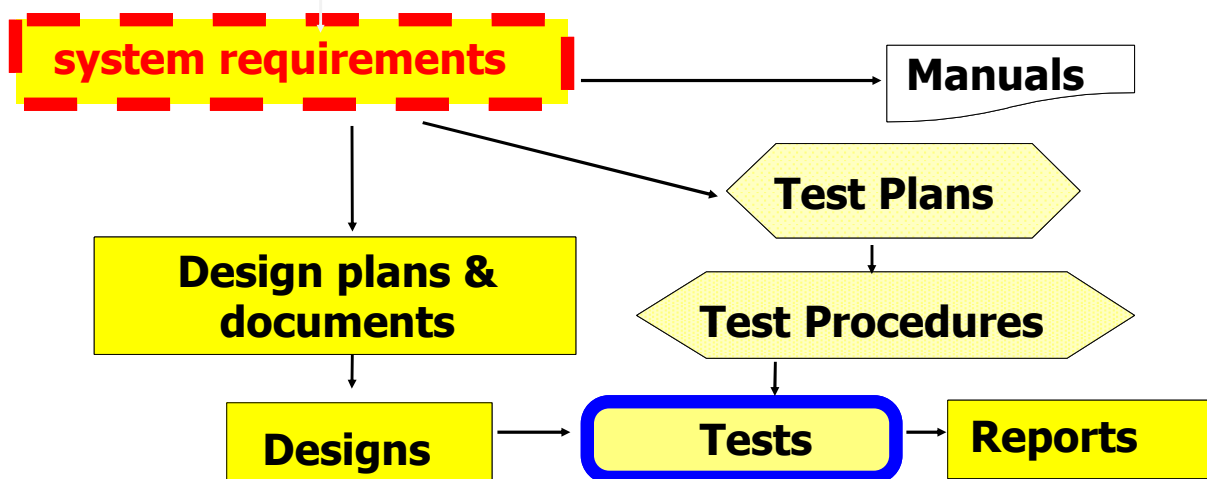
The benefits of using "system generic" and "system specific" requirements

- Requirements state the attributes and properties of something needed
- Example
 - System generic requirement for operating temperate of system is -10 to +50 degrees centigrade
 - System specific requirement for operating temperate of system in application is +5 to +50 degrees centigrade
- Prompts stakeholders not to forget operating temperature
- Instead of thinking about the needed temperature, stakeholder can think if generic value is needed or should be changed in specific application

How to write good requirements

0101-23

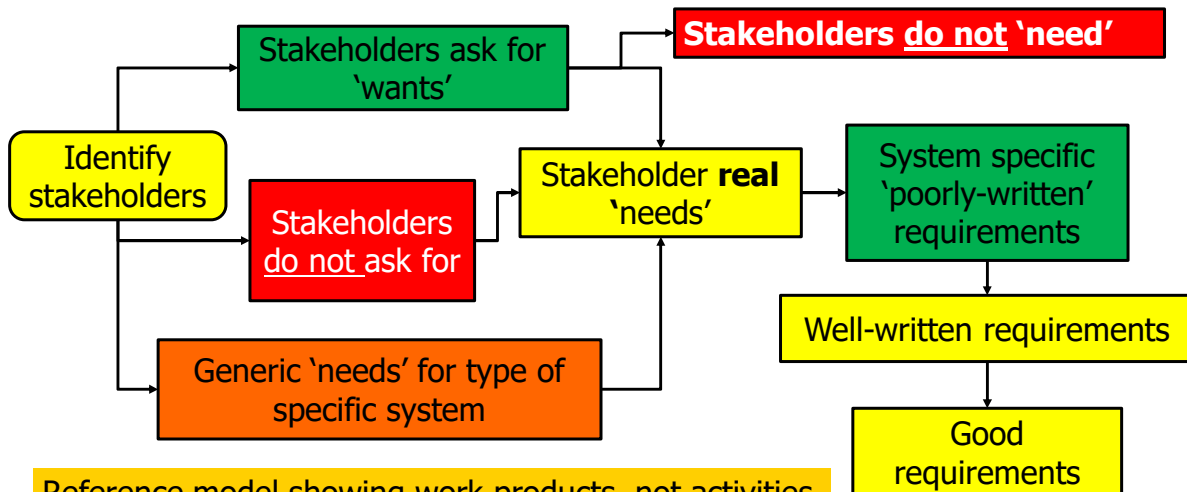
Importance of good system requirements



How to write good requirements

0101-24

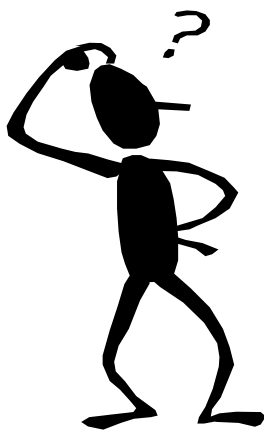
Gap analysis for writing good requirements



The top 5 reasons why you can't write a good requirement

0101-25

Six attributes of a well-written requirement



1. Atomic
2. Unambiguous
3. Verifiable
4. Understandable*
5. Adequate*
6. Complete
7. ~~Consistent*~~
8. ~~Linked or traced~~

1. Are attributes of a 'requirement' but do not show up in the wording of the statement
2. Are attributes of a good requirement

* IREB, 3.1.0, EU 3.8

How to write good requirements

0101-26

Non-functional requirements

- The **state of being** requirements
 - E.g., "ilities"
- Availability can be written as:
 - The system shall be operational for 24 hours each day
 - What shall be **done**
 - The Availability of the system shall be 24 hours daily
 - What shall **be**
- Acceptance criteria in this situation are generally a combination of reliability and mean time to repair
- Safety
 - Combination of "built into the system" and procedures
 - Risk procedure will be overridden
- Require safety to be "built into the system" rather than rely on procedures

Ensure system can meet the functional requirements as and when required

How to write good requirements

0101-27

The difference between "system generic" and "system specific" requirements

- System generic requirements
 - Come from the system generic attributes
 - Are specific system independent
 - Maximize the probability of creating a complete set of requirements
 - Provide an inherited template to quantify system generic attributes
- System specific" requirements are the instance of the system generic requirements (properties) for the system specific attributes

How to write good requirements

0101-28



Exercise 1-0

- Tell us about yourself in less than 20 seconds
 1. Who you are
 2. Where you are
 3. Why you are here
 4. What you want to learn
 5. Anything else you want to mention

How to write good requirements

0101-29



Exercise 1-1 Compliance to a poorly written requirement

1. Imagine that you have just written the test procedure for verifying the system compliance to the poorly written requirement 509.1
2. Visualize the test procedure performed
3. Prepare a <5 minute presentation containing
 1. Reformulated problem per COPS Problem Formulation Template
 2. What you did to test each part of the requirement
 3. How you ensured the whole poorly-written requirement was tested
 4. A compliance matrix for the exercise
 5. Lessons learned from exercise
4. Save as a PowerPoint file in format Exercise1.1-abcd.pptx
 1. Note (abcd are your initials, where d is to be used when two participants have the same initials)
5. Post/email presentation as, when and where instructed

How to write good requirements

0101-30



Exercise 1-2 Requirements evaluation

1. Read the requirements in reading 0102
2. Comment on at least 15 requirements in the reading
 - Are they “good” or “bad”, and why
3. Prepare <5 minute PowerPoint presentation containing
 1. Reformulated problem per COPS Problem Formulation Template
 2. The requirements you chose to comment on
 3. Your comments
 4. A compliance matrix for the exercise
 5. Lessons learned from exercise
4. Save as a PowerPoint file in format Exercise1.2-abcd.pptx
5. Post/email presentation as, when and where instructed

How to write ~~good~~ requirements

0101-31

How to write good requirements Module 2 of 10



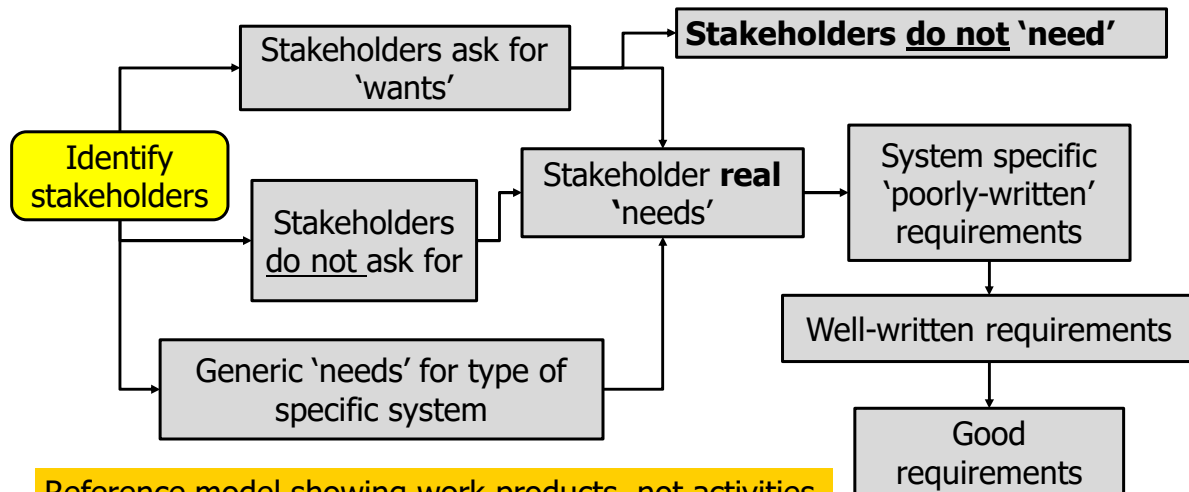
Stakeholders and their importance

Version 1.1.2

How to write good requirements

0201-32

Gap analysis for writing good requirements



The top 5 reasons why you can't write a good requirement

0101-33

Module 2 objectives

#	Objectives	Met
3	Explain where and how to locate potential stakeholders for the project	14-21
4	Explain the extended process to realize the solution system as a generic model for locating stakeholders	22-26, Readings
5	Explain the difference between	
5a	Customers and other stakeholders	28,29
5b	Information & contractual communications between stakeholders & how to manage them	28,29
5c	Stakeholder wants and needs	31
5d	Direct and indirect stakeholders	34,35
5e	Generic and specific stakeholders	37
6	Explain the degree of influence of each stakeholder on the requirements	39,40
7	Provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	42-45

How to write good requirements

0201-34

Most important lesson learned

1. You can't write good requirements unless you understand the stakeholder's real needs
2. You can't understand the stakeholder's real needs until you gain an understanding of the undesirability in the current undesirable situation
 - You need three domain knowledge
3. You can't gain an understanding of the undesirability in the current undesirable situation until you have a functional model of some kind (as-is)
 - Concept maps, scenarios, flow charts, descriptive paragraphs, in computer, etc.
 - Undesirability (why the customer is willing to fund the project)
 - Assumptions
4. You should develop a draft concept of operations (CONOPS) for the system the stakeholder needs (to-be) before and while communicating with the stakeholders
5. You should also develop a conceptual model of the transition process for
 1. Acquiring the needed system (build/develop/integrate or buy)
 2. Transitioning the needed system into the current situation
 3. Disposing of the existing system (if necessary)

How to write good requirements

0201-35

Stakeholders

- People or organizations that are internal or external to the project who have a vested interest in its success or failure
 - Defined in Module 1
- Are sources of (stakeholder) requirements (requirement-requests)
- Stakeholders should be documented in an up-to-date stakeholder list with (at least) the following information*
 - Name
 - Function (role)
 - Additional personal and contact data
 - Temporal and spatial availability during the project progress
 - Relevance (influence)
 - Area and extent of expertise
 - Goals and interests in terms of the project

* IREB, 3.10, EU 4.1

How to write good requirements

0201-36

Stakeholder location in the literature

- Subset of an undefined list
 - “including”
- Which are relevant?
- How to manage conflicting concerns?
 - Quality Function Deployment (QFD)
- No systemic and systematic method for how to
 1. identify stakeholders
 2. manage stakeholder expectations



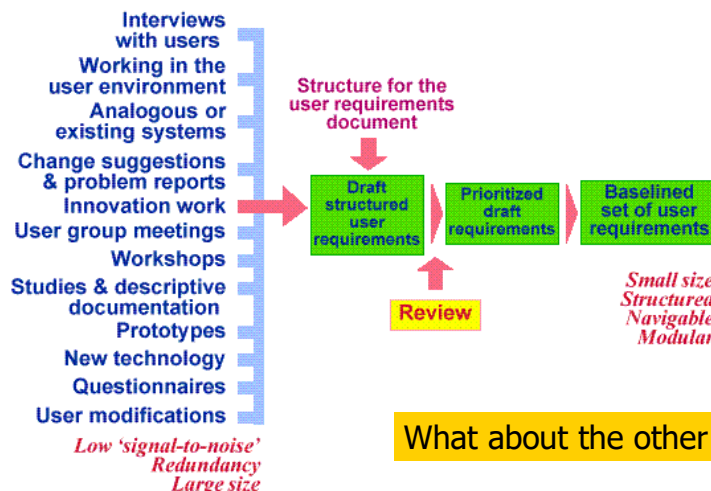
How to write good requirements

0201-37

Incomplete user stakeholders (1990s)

Sources of user requirements

© QSS Inc. All rights reserved. DERA



0201--38

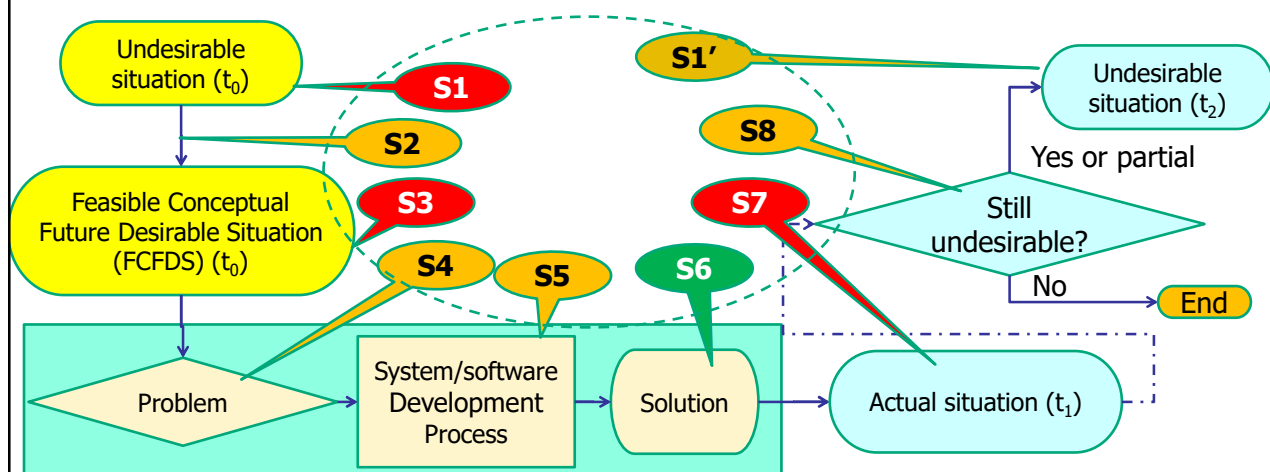
Viewpoint analysis (1999)

- Look at the system from the perspectives of:
 - End Users
 - Operators
 - Support Staff
 - Developers
 - Owners
- Draw out requirements and prioritise

How to write good requirements

0101-39

The nine system model (Functional view)



How to write good requirements

0201-40

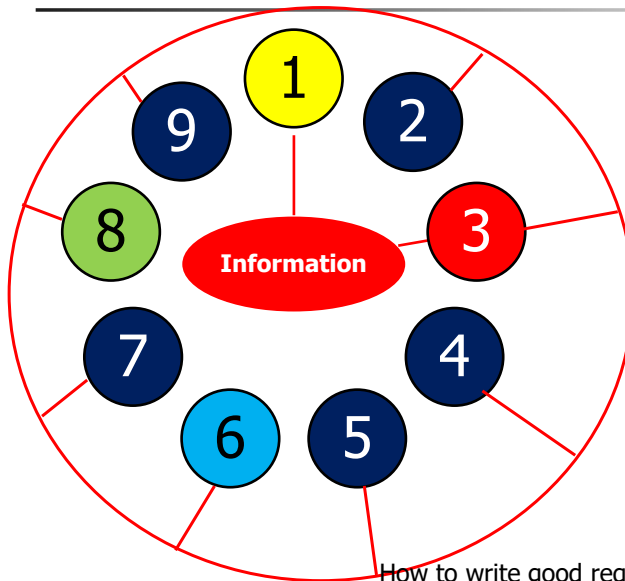
The 9 systems: situations, systems & processes

1. Undesirable or problematic **situation**
 - Baselined at t_0 , but will evolve during realization of solution system
2. **Process** to develop the Feasible Conceptual Future Desirable Situation (FCFDS)
3. The FCFDS that remedies the undesirable **situation**
4. **Process** to plan the transition from the undesirable situation to the FCFDS
5. **Process** to realize the transition by providing the solution system
6. Solution **system** that will operate within FCFDS'
7. Actual or created **situation** at t_1
8. **Process** to determine that the realized solution remedies the evolved undesirable situation
9. **Organization(s)** containing the **processes**

How to write good requirements

0201-41

Systems view of stakeholders - contractual view

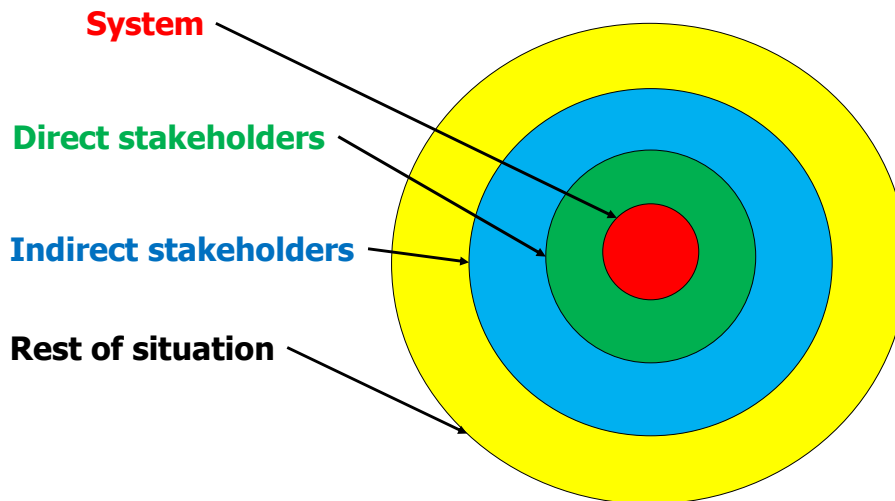


- Contractor is stakeholder 1
- Customer is stakeholder 3
- Contractual information only flows between customer and contractor (red line)
- Customer is contractual buffer between all stakeholders and contractor project personnel
- Clear control of which needs become requirements and who makes the decision
- Important perspective when all stakeholder needs cannot be met within the budget and schedule
- Divide and conquer

How to write good requirements

0201-42

Direct and indirect stakeholders



How to write good requirements

0201-43

Exercise 2-1

1. Using the nine-system model, identify some of the direct and indirect generic stakeholders for the SECTS upgrade project
2. Create at least three scenarios in each of the three states of the situation
3. Prepare a <5 minute presentation containing
 1. The generic stakeholders and the scenarios in which they were found
 2. The three most influential stakeholders and why they are influential
 3. A compliance matrix for the exercise
 4. Formulated problem per COPS problem formulation template
 5. Lessons learned from exercise
4. Save as a PowerPoint file in format Exercise2.1-abcd.pptx
5. Post/email presentation as, when and where instructed

How to write good requirements

0201-44

How to write good requirements

Module 3 of 10

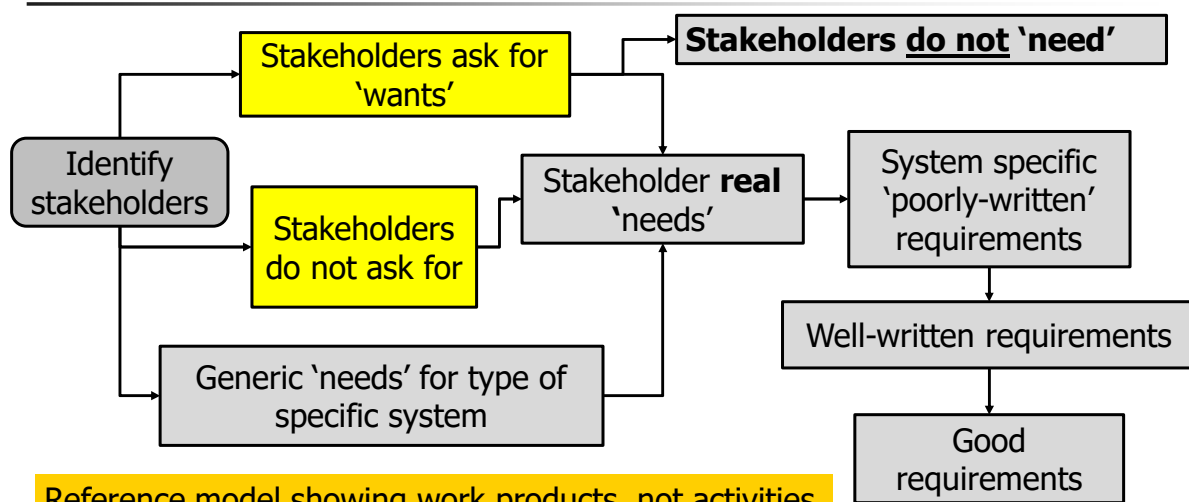
Communicating with the stakeholders

Version 1.1.3

How to write good requirements

0301-45

Gap analysis for writing good requirements



How to write good requirements

0301-46

Module 3 objectives

#	Objective	Met
1	Identify some barriers to communications	21, R 0303
2	Explain more than eight tools to overcome the barriers	23-46, R 0302
3	Explain how to extract "wants" from stakeholders	48, 50-54
4	Explain how to discourage stakeholders from asking for not needed	53,54
5	Explain what to do before meeting the stakeholders	48
6	Explain what to do when meeting the stakeholders	50-55
7	Explain what to do after meeting with the stakeholders	57
8	Provide the opportunity to exercise the 5 levels of knowledge	59-62

How to write good requirements

0301-47

Eight tools to overcome the barriers

1. Asking questions
2. Holistic Thinking Perspectives
3. Active Brainstorming
4. Active listening
5. Pattern matching
6. STALL
7. Keep it Simple, Student (KISS)
8. The principle of hierarchies



How to write good requirements

0301-48

Properties of a good question statement

- Answer implementation independent
 - Does not tell what the answer is
 - Not a leading question
- Quantitative
- Concise
- Pertinent
 - Boundaries (external and internal)
- Complete
- Stated as a function
- Unambiguous
- Insensitive to errors in interpretation

Notice similarity between questions and requirements?

How to write good requirements

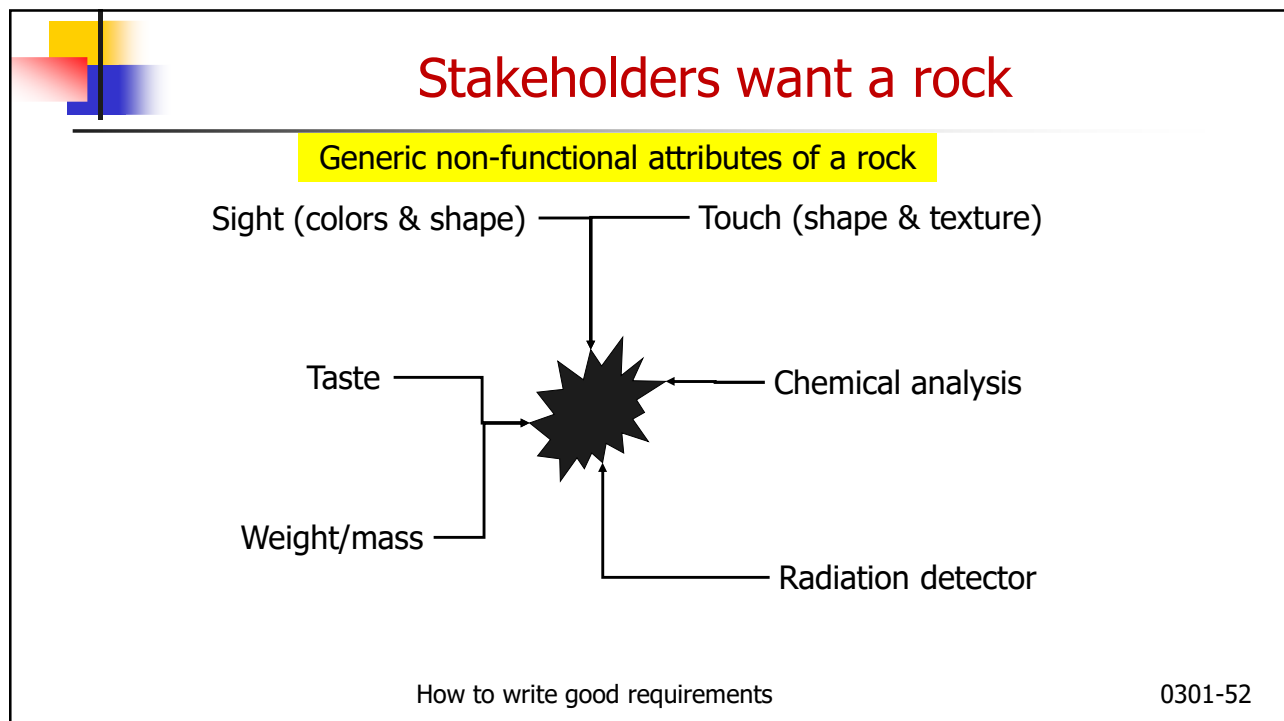
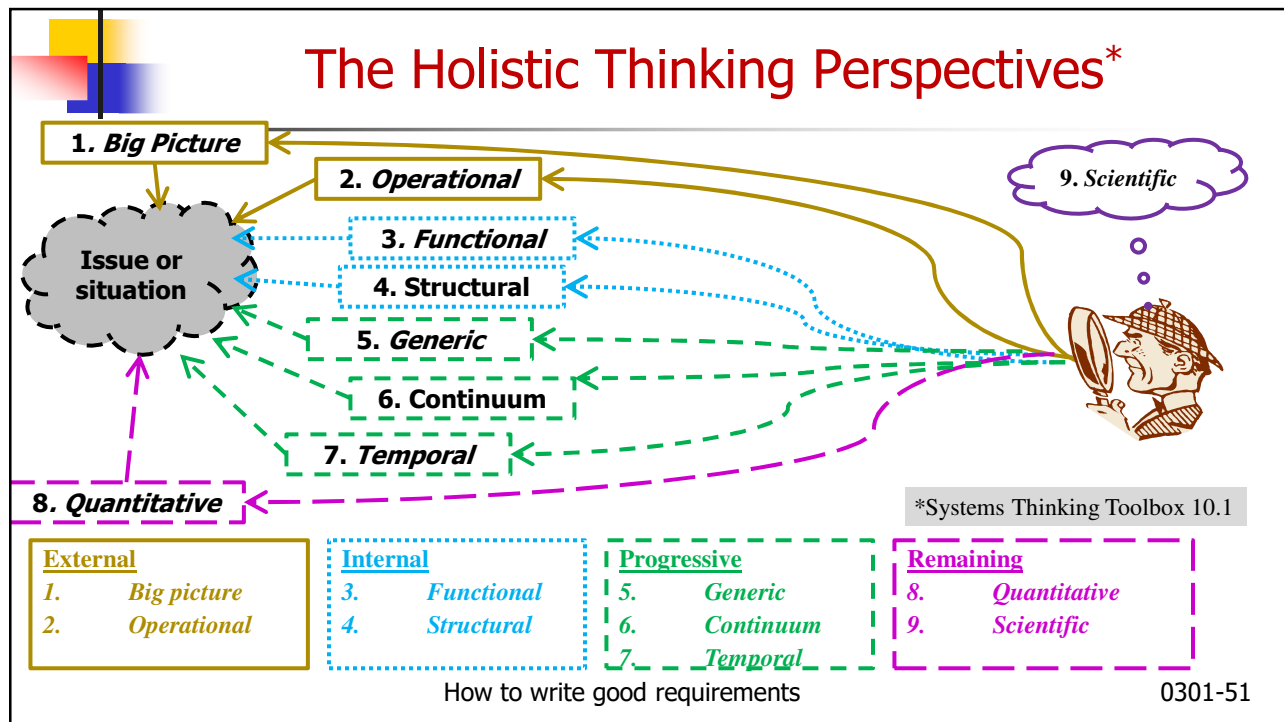
0301-49

Standard answer to nearly every question

- "it depends"
 - Request of further information
 - Removes annoyance of not answering question immediately
 - Opens dialogue to explore situation
 - What it depends on

How to write good requirements

0301-50



Needs, wants and problems

- Difference between
 1. What stakeholders want (ask for)
 2. What stakeholders need
 3. What stakeholders tell you

Stakeholders		Know what they <u>need</u>	
		Yes	No
Know what they <u>want</u>	Yes	Well-structured problem	Ill-structured problem
	No	Ill-structured problem	Well-structured problem'

How to write good requirements

0301-53

One way to tackle the problem

- Know what they need
 - Document the need
- Know what they want
 - Use applicable generic functional model to make sure it is a need
 - Use applicable generic functional model to show why a want is not a need (if applicable)
- Don't know what they want
 - Try the rock approach based on the applicable generic functional model
- Don't know what they need
 - Help them work out what they need using
 - The applicable generic functional model
 - Benchmarking results of similar systems

How to write good requirements

0301-54



Exercise 3-1

1. Select one generic stakeholder from each of the three situations from Exercise 2-1.
2. Craft at least 5 questions for each stakeholder to be met
3. Assume you met the specific instance of the generic stakeholders
4. Document the information you recorded after meeting with each of the stakeholders and asking the questions (assume reasonable feasible answers)
5. Prepare a <5 minute presentation containing
 1. The generic stakeholders and why you selected them
 2. The questions you prepared for each of the three stakeholders
 3. The documented responses from one of the stakeholders
 4. A compliance matrix for the exercise
 5. Formulated problem per COPS problem formulation template
 6. Lessons learned from exercise
 7. This slide
6. Save as a PowerPoint file in format Exercise3.1-abcd.pptx
7. Post/email presentation as, when and where instructed

How to write good requirements

0301-55



Exercise 3-2 (10-20 minutes)

- The unintended ambiguity in the question "make sure he's dead?" resulted in an unintended outcome
1. Prepare a <2 minute presentation containing
 1. What was the requirement she was complying with?
 2. What question should the operator have asked?
 3. Lessons learned from exercise
 4. This slide and the version number
 2. Save as a PowerPoint file in format Exercise3.2-abcd.pptx
 3. Post/email presentation as, when and where instructed

How to write good requirements

0301-56

How to write good requirements

Module 4 of 10

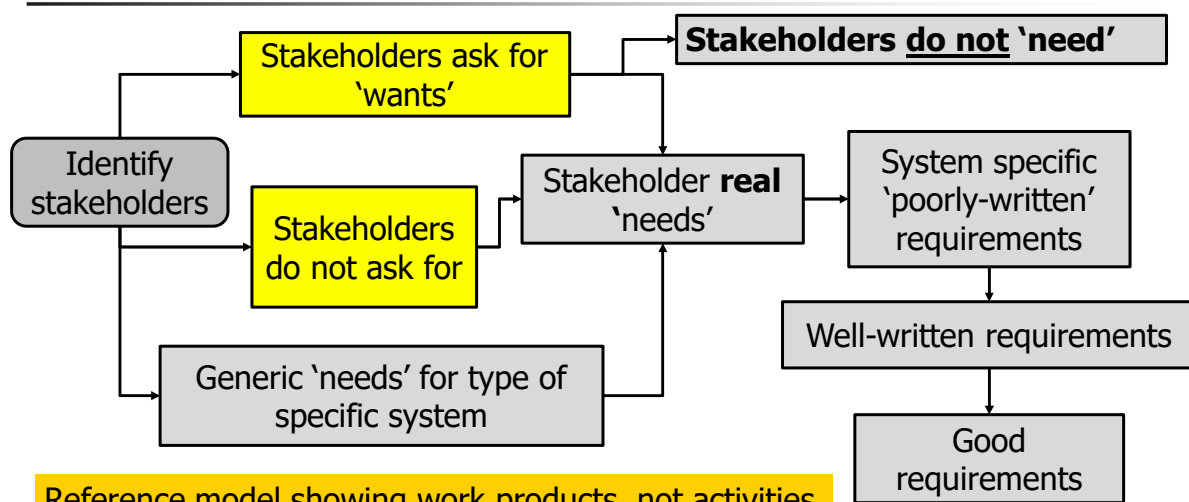
Converting stakeholder wants to needs

Version 1.3.4

How to write good requirements

0401-57

Module 4 objectives



The top 5 reasons why you can't write a good requirement

0301-58



Meeting the objectives

#	Objectives	Met
1	Explain how to convert stakeholder functional and performance "wants" to "needs"	16-21
2	Explain the difference between functions and misuse functions	23-28
3	Introduce risk management	29-35
4	Explain three ways to maximize the completeness of the needs	21
5	Explain need for prioritization of needs and how to prioritize them	37-38
6	Explain how to influence the stakeholders to want the system they need	46-50
7	Explain how to determine if the need is for COTS equipment	52
8	Provide the opportunity to exercise 5 levels of knowledge in the updated Blooms taxonomy	54-56

How to write good requirements

0401-59



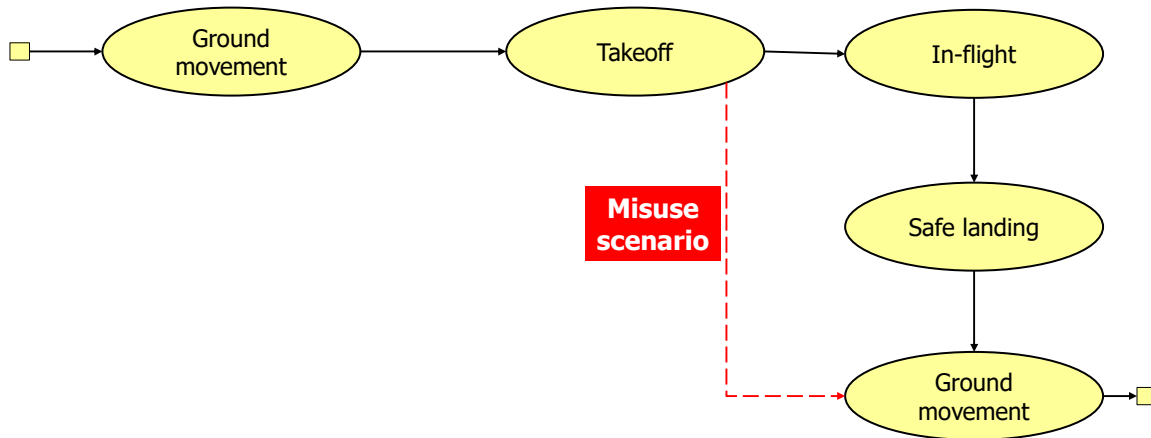
The three functional representations

- Each **representation** contains quantified functional scenarios including 'don't cares'
 - Current situation (as-is)
 - CONOPS (to-be)
 - Transition process
- Compare wants/needs to generic CONOPS and transition representations
 - Checklands' SSM, see Reading 0402, can be tailored to situation
- Adjust CONOPS and transition representations to include specific needs, creating the specific CONOPS

How to write good requirements

0401-60

Misuse scenarios (functions)

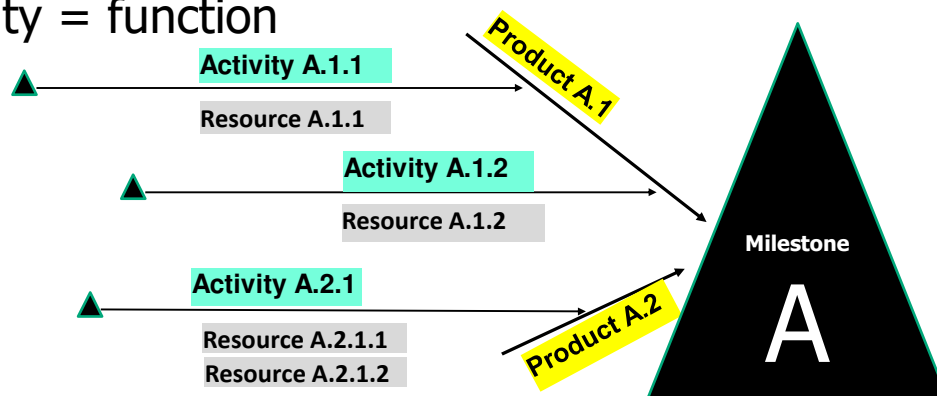


How to write good requirements

1001-61

Product-Activity-Milestone (PAM) chart*

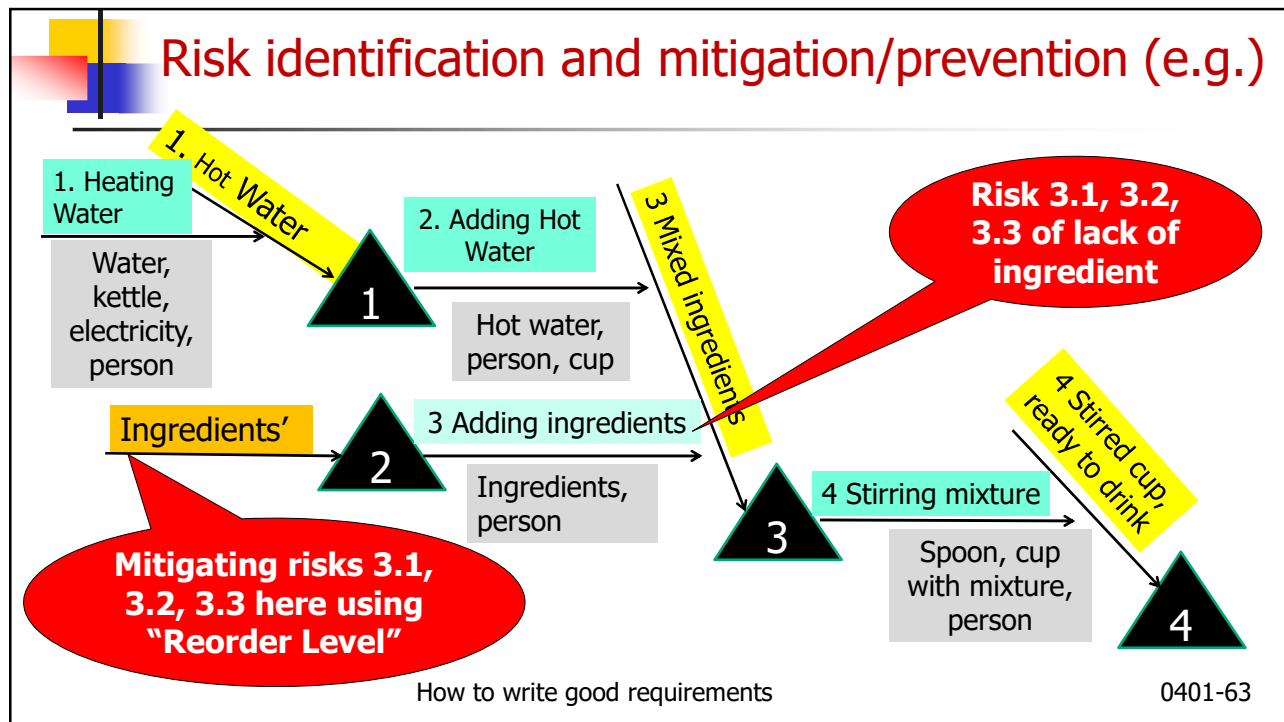
- Activity = function



Systems Thinkers Toolbox 2.14

How to write good requirements

0401-62



Traditional Risk Assessment Matrix

Probability of occurrence (L) (Likelihood)	5	5	10	15	20	25
	4	4	8	12	16	20
	3	3	6	9	12	15
	2	2	4	6	8	10
	1	1	2	3	4	5
		1	2	3	4	5

Severity of consequences (S)
(Impact)

Based on one number (L*S)

The level of risk for each root cause is reported as:

1-4	Low (green)
5-12	Moderate (yellow)
15-25	High (red)

How to write good requirements

0401-64

Where do numbers come from?

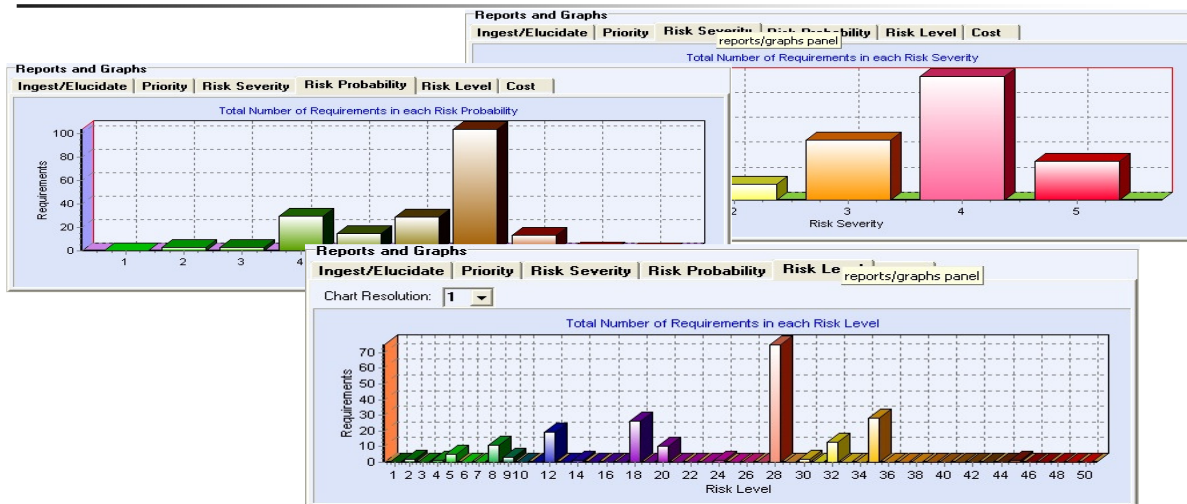
- Experience
- Historical data
 - E.g., actuarial data
- Expert estimates
- Standards
- Assumptions
- Guesses
- Other places



How to write good requirements

0401-65

Project Risk profiles*



How to write good requirements

0401-66

What' next?

■ Traditional approach

1. Convert the quantified needs scenarios to **system requirements**
2. **Write system requirements**
3. Estimate cost and schedule to implement **requirements** (ideal)
4. **Adjust for feasible schedule and affordable cost** (ideal)
5. Do some **risk management**
6. Create
 1. Project Plan (PP) or Systems Engineering Master Plan (SEMP)
 2. Test and Evaluation Master Plan (TEMP)
 3. Other appropriate documents

■ Systems approach

1. Add quantified **misuse** scenarios (**risk management**)
2. Estimate cost and schedule to implement **need scenarios**
3. **Prioritize needs**
4. **Adjust for feasible schedule and affordable cost**
5. **Write good system requirements**
6. Create
 1. Project Plan (PP) or Systems Engineering Master Plan (SEMP)
 2. Test and Evaluation Master Plan (TEMP)
 3. Other appropriate documents

How to write good requirements

0401-67

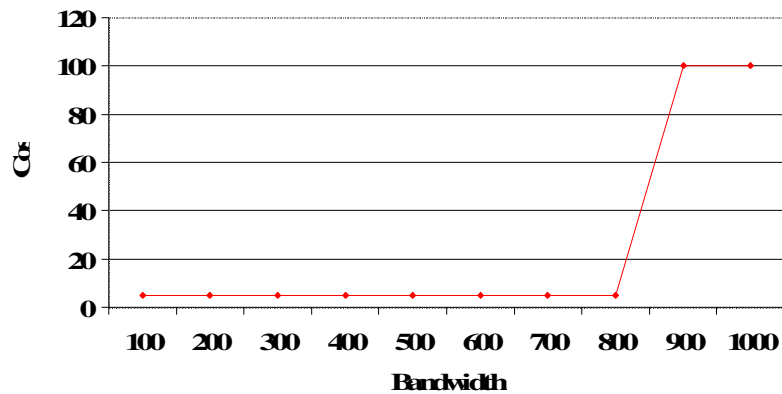
At this time

- All needs are stored in a set of functional scenarios
 - Mission and support product/system
 - Acquisition process (build/buy) and transition
- Remove contradictions and duplications from set of scenarios
- Perform appropriate feasibility studies
 - Technical
 - Cost
 - Schedule
- Adjust until affordable and achievable within time constraints
 - Remove needs or add costs and schedule time

How to write good requirements

0401-68

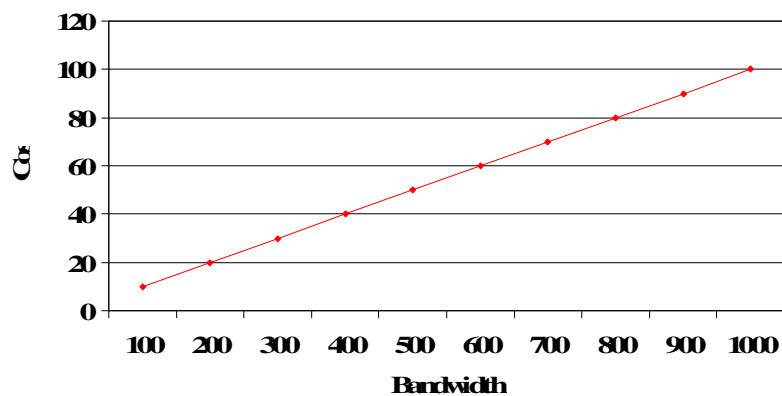
Technical sensitivity analysis -1



How to write good requirements

0401-69

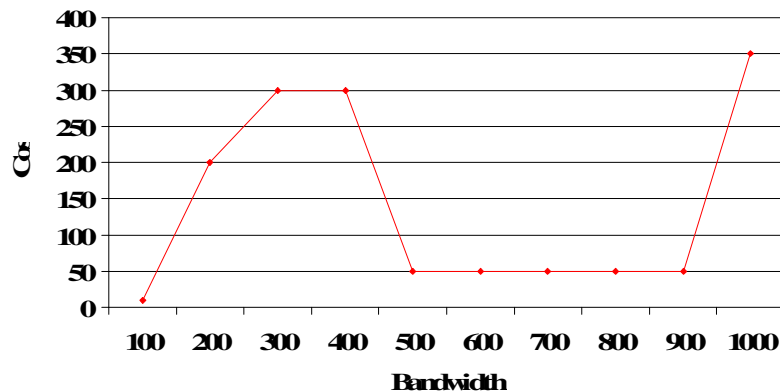
Sensitivity analysis -2



How to write good requirements

0401-70

Sensitivity analysis -3

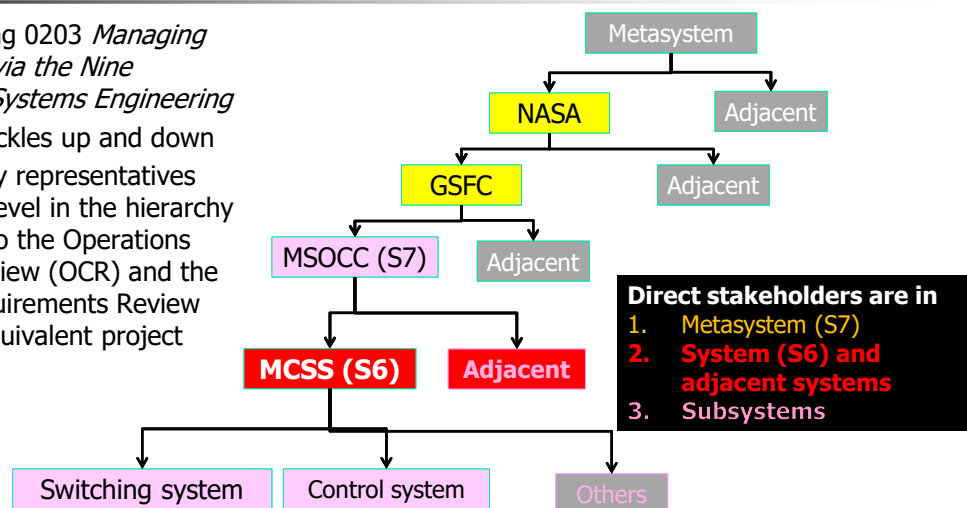


How to write good requirements

0401-71

Influencing direct/indirect stakeholders

- From Reading 0203 *Managing Complexity via the Nine Systems in Systems Engineering*
- Influence trickles up and down
- Which is why representatives from each level in the hierarchy are invited to the Operations Concept Review (OCR) and the System Requirements Review (SRR) (or equivalent project milestones)



How to write good requirements

72



Exercise 4-1 scenarios to requirements

1. Update the extracted scenarios in your Exercise 2-1 presentation
2. Assign an identification to each scenario
3. Add at least one misuse function to at least one scenario
4. Write at least 10 well-written requirements with traceability to any of the scenarios including the scenario(s) updated with the misuse function(s)
5. Prepare a <5 minute presentation containing
 1. The misuse function in its scenario highlighting the modification(s) if any
 2. At least 10 well-written requirements including some traceable to the misuse function
 3. Traceability of each requirement to the source scenario and function
 4. A compliance matrix for the exercise
 5. The exercise problem formulated per COPS problem formulation template
 6. Lessons learned from exercise
6. Save as a PowerPoint file in format Exercise4.1-abcd.pptx
7. Post/email presentation as, when and where instructed

How to write good requirements

0401-73



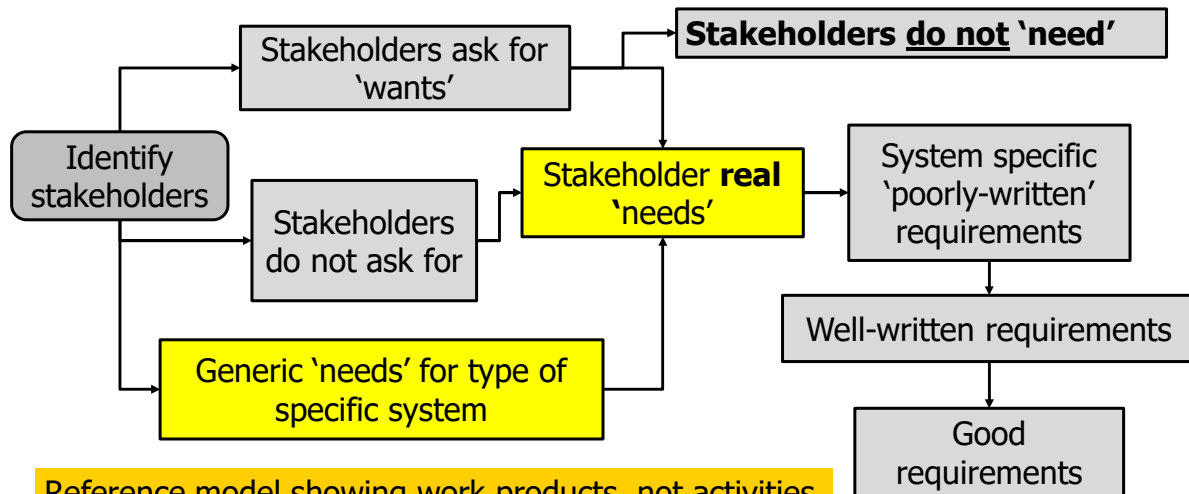
How to write good requirements Module 5 of 10 **Documenting and storing stakeholders' needs**

Version 1.0.2

How to write good requirements

0501-74

Gap analysis for writing good requirements



How to write good requirements

0301-75

Module 5 objectives

#	Objectives	Met
1	To explain how to document and store the stakeholder needs	16-17
2	To provide reasons for documenting needs	15
3	To explain some things you should be aware of about models and modelling	19-22
4	To discuss some things you should be aware of about modeling tools in general	24-26
5	To explain some of the different types of models	28-46
6	To explain uses, advantages and limitations of models	48-49
7	To explain how to create functional models	51-54
8	To explain how to communicate functional and operational models to stakeholders	56
9	To practice creating parts of a model	58
10	To practice writing requirements traceable to scenarios	59
11	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	58-62

How to write good requirements

0501-76

Models

- “Essentially, all models are wrong, but some models are useful.”
 - George E. P. Box, Science and Statistics, the Journal of the American Statistical Association Vol. 71, No. 356 (Dec., 1976), pp. 791-799.
 - Referred to statistical and analytical models
- Statistical models contain three basic types of assumptions
 1. Assumptions about the distribution of values in a variable
 2. Assumptions about the functional relationship between variables
 3. Assumptions about the probabilities
- Assumption
 - What applies to statistical models applies to all other models
 - Other types of models contain other types of assumptions

How to write good requirements

0501-77

Models and representations

- A model can be an abstract, logical, physical or some other representation of an existing part of reality or a part of an imagined reality
 - To be created
 - To use as background to a story
- May need more than one model to develop an understanding (HTP)
 - Allow us to engineer things
 - E.g. wave and particle theory of electromagnetic propagation
- Various types of models/representations, including
 - Breadboards for form and fit
 - Relationships for function
 - Performance
 - Physical mock ups
- **Be aware of the assumptions underlying a model before reuse**

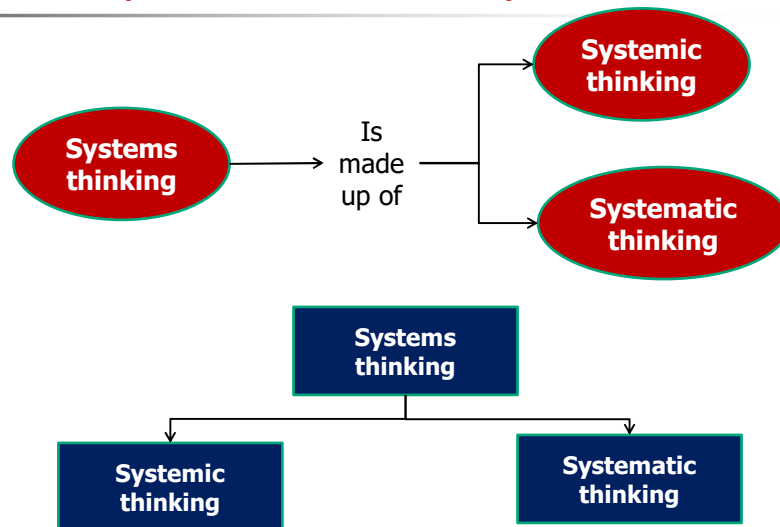


How to write good requirements

0501-78

Two *Structural* representations of systems thinking

- Do not combine *Structural* and *Functional* relationships in the same graphic
- Use two graphics
- Using a single graphic creates artificial complexity



How to write good requirements

0501-79

Models (advantages and limitations)*

- | | |
|--|--|
| <ul style="list-style-type: none"> • Advantages <ul style="list-style-type: none"> ■ Relationships and interconnections between requirements are easier to understand with graphic models ■ Focusing on a single aspect reduces the cognitive load for understanding the modeled requirements ■ Requirements modeling languages have a restricted syntax which reduces possible ambiguities and omissions | <ul style="list-style-type: none"> • Limitations <ul style="list-style-type: none"> ■ Keeping [different] models that focus on different aspects consistent with each other is challenging ■ Information from different models needs to be integrated for causal understanding ■ Models focus primarily on functional requirements; most quality [non-functional] requirements and constraints cannot be expressed in models with reasonable effort ■ The restricted syntax of a graphic modeling language implies that not every relevant item of information can be expressed in a model |
|--|--|

* IREB, 3.10, EU 3.4.1

How to write good requirements

0501-80

Communicating models to stakeholders

- **Don't !!!!!!!!!!!!!**
- Don't expect the stakeholders to learn a modelling language or understand the graphics produced by your modelling tools
- Communicate the relevant data with the rest abstracted out (next slide)
 - Using the Principle of Hierarchies in Reading 0302
- Use customer's language to represent the data in the model
 - Simple connected blocks and circles
 - Sketches, Rich Pictures (Checkland)
 - Videos and animations
 - Pictures
 - Others
- Use the eight tools to overcome the barriers from Module 3 as appropriate

How to write good requirements

0501-81

Exercise 5-1 Creating scenarios

1. Prepare a <5 minute presentation containing
 1. Convert any 10 of the responses from the stakeholder in Exercise 3-1 to a set of linked scenarios
 2. Identify the mission, support and risk prevention/mitigation functions (as appropriate)
 3. A higher level drawing (or N² chart) showing the scenarios and the interfaces between the scenarios
 4. The exercise problem formulated per COPS problem formulation template
 5. A compliance matrix for the exercise
 6. Lessons learned from exercise
2. Save as a PowerPoint file in format Exercise5.1-abcd.pptx
3. Post/email presentation as, when and where instructed

How to write good requirements

0501-82

Exercise 5-2 Deriving requirements from scenarios

1. Write a complete set of well-written requirements traceable to the scenario in the next slide (as complete as you can, within a 60 minutes time limit)
2. Prepare a <5 minute presentation containing
 1. The well-written requirements
 2. Identify the mission, support and risk prevention/mitigation requirements (if any)
 3. If there is anything in the scenario that does not generate a requirement, discuss the reason(s) (maximum of two)
 4. The exercise problem formulated per COPS problem formulation template
 5. A compliance matrix for the exercise
 6. Lessons learned from exercise (<3 min)
 7. Why you think this exercise was included in the lesson
3. Save as a PowerPoint file in format Exercise5.2-abcd.pptx
4. Post/email presentation as, when and where instructed

How to write good requirements

0501-83

Exercise 5-3 Background

- Imagine you have just watched 5 student presentations
- The attributes of the presentations are shown in compliance matrix format
- All 5 presentations comply with the 4 requirements for the exercise
- 4 presentations contain some additional attributes

Attributes of exercises	Presentations				
	A	B	C	D	E
Exercise requirement 1	✓	✓	✓	✓	✓
Exercise requirement 2	✓	✓	✓	✓	✓
Exercise requirement 3	✓	✓	✓	✓	✓
Exercise requirement 4	✓	✓	✓	✓	✓
Slides are in a logical order	✓			✓	
Title and presenter's name slide	✓			✓	
Colorful theme	✓	✓	✓		
Closing slide (any questions)	✓		✓	✓	

How to write good requirements

0601-84

Exercise 5-3 the real requirement

1. Rank the presentations (A-E) in order of your preference from liking most to liking least
2. Prepare a <5 minute presentation containing
 1. A sorted list of the presentations in order of preferences (high to low)
 2. The reasons for the ranking (sorted order)
 3. A compliance matrix for the exercise
 4. Lessons learned from exercise
 5. The exercise problem formulated per COPS problem formulation template
3. Save as a PowerPoint file in format Exercise5.3-abcd.pptx
4. Post/email presentation as, when and where instructed

How to write good requirements

0501-85

Comments on Kano model

- The Kano model was developed for mass-market products
 - (As was Agile software development)
- In bespoke (single customer) system development,
 - The system requirements **should** be considered as a minimum, because
 - Exceeding expectations (the requirements) **within cost and schedule constraints** does lead to greater satisfaction
- Did you notice how the presentations seemed better (more professional) in the exercises presented subsequent to Exercise 5-1 after participants added
 - Titles
 - Colour (some)
 - Endings

How to write good requirements

1001-86

How to write good requirements

Module 6 of 10

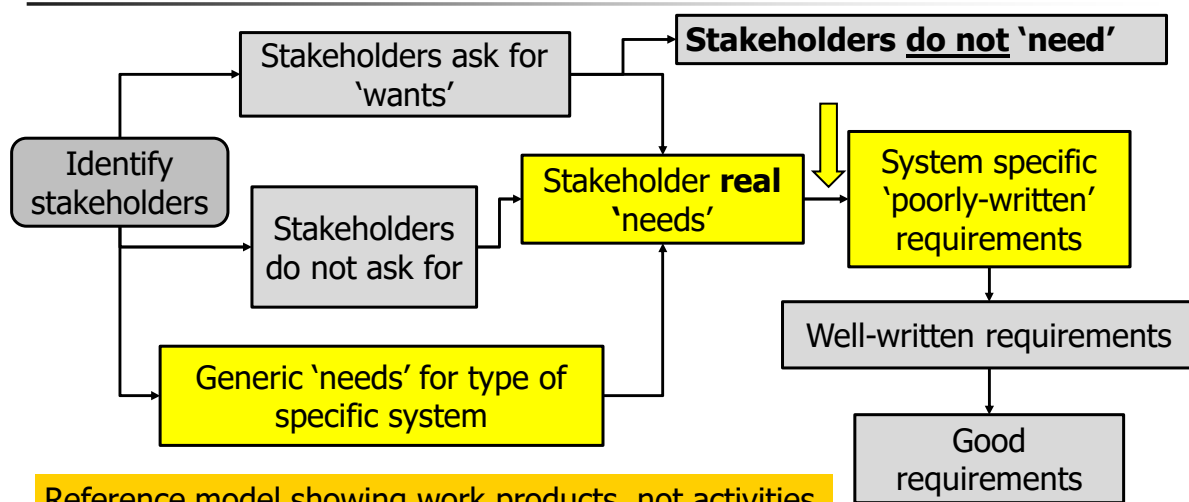
Converting stakeholder needs to requirements

Version 1.0.2

How to write good requirements

0601-87

Gap analysis for writing good requirements



How to write good requirements

0601-88

Module 6 objectives

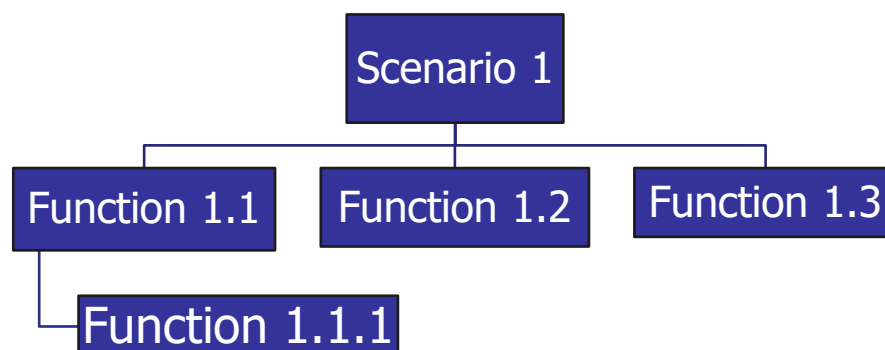
#	Objectives	Met
1	To explain what to do before writing the requirements	18-20
2	To explain what to do while writing the requirements	22-32
3	To explain what to do immediately after writing the requirements	41-42
4	To explain how to deal with some types of poorly-written requirements	37
5	To discuss attributes of requirements for flexible systems	39
6	To provide the opportunity to use a tool to convert some poorly-written requirements into well-written requirements	44-51
7	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	45-52

How to write good requirements

0601-89

Converting scenarios to atomic functions

- Use Principle of Hierarchies



How to write good requirements

0601-90

Functional requirement and performance requirements

- What is the difference between a **functional** requirement and a **performance** requirement, is it
 - A performance requirement is a functional requirement with numbers?
- Yes!
- So,
- A functional requirement is the instruction to perform a function, e.g.,
 - The system shall communicate
 - The system shall respond
 - The system shall track the sun
- Which is a part of the source scenario (poorly-written stakeholder request)
- **In this class**
 - **A function** is part of a scenario and may be a (sub)heading in a requirement specification
 - **A functional requirement** is a well-written requirement which specifies needed performance (performance requirement)

How to write good requirements

1001-91

Functional requirements (ChatGPT)*

- Definition
 - Functional requirements describe **what a system or software application must do.**
 - They outline the specific features, functions, and capabilities that the system should have to meet the needs of its users.
- Focus
 - These requirements focus on the behavior of the system, specifying the input, processing, and output for various functions.
 - **They answer the question, "What does the system do?"**
- Examples
 - In the context of a web application, functional requirements could include features like user authentication, data validation, report generation, and search functionality.

* Accessed 14 November 2023

How to write good requirements

1001-92

Functional requirements (ChatGPT)

- User Stories*
 - Functional requirements are often expressed in the form of user stories. These stories capture the intended functionality from the perspective of an end user.
- Use Cases*
 - Use cases are another way to document functional requirements. They describe specific scenarios or interactions between users and the system.
- In other words, stakeholder requirement requests and scenarios
- 'B' paradigm

* Accessed 14 November 2023

How to write good requirements

1001-93

Performance requirements (ChatGPT)*

- Definition
 - Performance requirements specify how well the system must perform certain functions under specific conditions. They address aspects such as response time, throughput, scalability, and reliability.
- Focus
 - Performance requirements are concerned with the system's efficiency, speed, and resource utilization.
 - They answer the question, "How well does the system perform its functions under different conditions?"
- Examples
 - Performance requirements might include specifications for response times (e.g., the system must respond to a user query within 2 seconds), throughput (e.g., support 1000 concurrent users), and reliability (e.g., achieve 99.9% uptime).

* Accessed 14 November 2023

How to write good requirements

1001-94

Performance requirements (ChatGPT)*

- Measurable Metrics
 - Performance requirements should be measurable and quantifiable. For example, instead of stating that the system should be fast, it's more effective to specify that the response time for a particular operation should be under a certain threshold.
- Scalability
 - Scalability is a crucial aspect of performance requirements. It addresses how well the system can handle increased loads or growing user numbers. It may involve considerations such as the ability to scale horizontally (adding more servers) or vertically (increasing resources on existing servers).
- Reliability and Availability
 - Performance requirements often include criteria related to system reliability and availability. This could involve specifying the maximum allowable downtime or setting targets for system uptime.

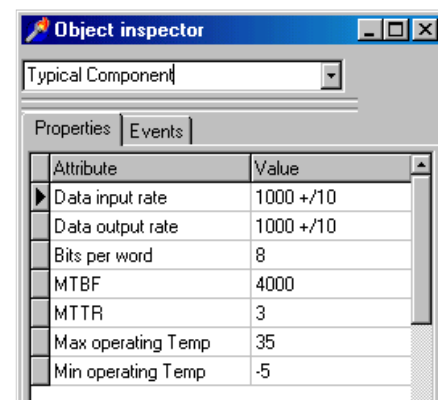
* Accessed 14 November 2023

How to write good requirements

1001-95

Converting properties and attributes to requirements

- 101 The system shall receive data at a rate of 1000 ± 10 bps
- 102 The data received by the system shall contain 8 bits per word
- 111 The system shall transmit data at a rate of 1000 ± 10 bps
- 112 The data transmitted from the system shall contain 8 bits per word
- 501 The **estimated** system MTBF shall be greater than 4000 hours
- 502 The **estimated** system MTTR shall be less than 3 hours
- 601 The maximum operating temperature of the system shall be greater than 35 degrees Centigrade
- 602 The minimum operating temperature of the system shall be less than -5 degrees Centigrade



Object inspector	
Typical Component	
Properties Events	
Attribute	Value
Data input rate	1000 +/-10
Data output rate	1000 +/-10
Bits per word	8
MTBF	4000
MTTR	3
Max operating Temp	35
Min operating Temp	-5

How to write good requirements

0601-96

Functional and non-functional (text) scenario

103. Except on Saturdays, the system shall transport up to 1000 men with up to 100 Kilograms of baggage each, up to 1000 miles, within 10 hours.

- Functional
 - What is being transported
- Non-functional
 - When it is being transported (Sunday to Friday)

How to write good requirements

0601-97

Atomic requirements from scenario

103. The system shall operate six days a week, Sunday to Friday^[1].
104. The system shall transport up to 1,000 men each weighing no more than w Kilograms^[2].
105. The system shall transport up to 100,000 Kilograms of baggage.
106. The system shall transport the combination of men and baggage up to 1600 Kilometers^[3].
107. The system shall complete the transport within 10 hours^[4].
108. The volume of an individual item of baggage shall not exceed h by w by d Meters.^[5]

Questions

- [1] How many hours per day?
- [2] Should we use minimum, average or maximum weights for the people?
- [3] What state should the men and baggage be in after transportation?
- [4] Is the 10 hours included in 103?
- [5] What is form factor of baggage?
- [6] ...

This is why you need scenarios in a CONOPS

How to write good requirements

0601-98

RTM: Example of smart numbering and linking

- Allows human correlation
- Shows where prevention is being performed
- Double entry (bookkeeping)
- Need columns not shown, e.g.
 - S.23 traces back to **N**(eed).23
 - traces back to sta**K**eholder JK
- Expanded front end of the Requirements Traceability Matrix (RTM)
- Builds risk management into the front end of the process instead of being an add-on elsewhere
- Partial table shown
- Colours group scenarios (human factor)
- Missing link identified by colour
 - (defaults X.O.Z.O)

Scenario	Function	Risks	Risk is prevented in (->)	Prevents risk identified in (<-)
S.23	F.23.1	R.23.1.1	F.55.6	None
S.23	F.23.1	R.23.1.2	F.73.8	None
S.23	F.23.2	R.23.2.1	F.76.5	None
S.23	F.23.3	None	N/A	None
S.55	F.55.6	None	N/A	R.23.1.1
S.73	F.73.8	R.73.8.1	S.245.5	R.0.0.0
S.76	F.76.5	R.76.1.1	S.367.5	R.43.2.1

How to write good requirements

0601-99

Example of smart numbering and linking in conceptual tool

- Scenario shown in centre
 - Separates input and output
 - Eyes follow flow
- Missing link identified by colour
 - (defaults X.O.Z.O)
- Not part of RTM
- Allows computer tool to check for completeness of needs, and later requirements
- Colours group scenarios (human factor)
- Partial table shown
- View parts of table (database) at a time
 - By lines, scenarios, inputs, outputs, missing links, etc.

Input	Comes from	Scenario	Output	Goes to
S.23.I.1	S.12.O.4	S.23	S.23.O.1	S.25.I.2
S.23.I.2	S.10.O.1	S.23	S.23.O.2	S.28.I.1
S.23.I.3	S.12.O.4	S.23	None	N/A
S.23.I.4	S.76.O.1	S.23	None	N/A
S.55.I.1	S.43.O.1	S.55	S.55.O.1	S.73.I.1
S.73.I.1	S.55.O.1	S.73	S.73.O.1	S.0.I.0
S.76.I.1	S.65.O.1	S.76	S.76.O.1	S.80.I.1

How to write good requirements

0601-100

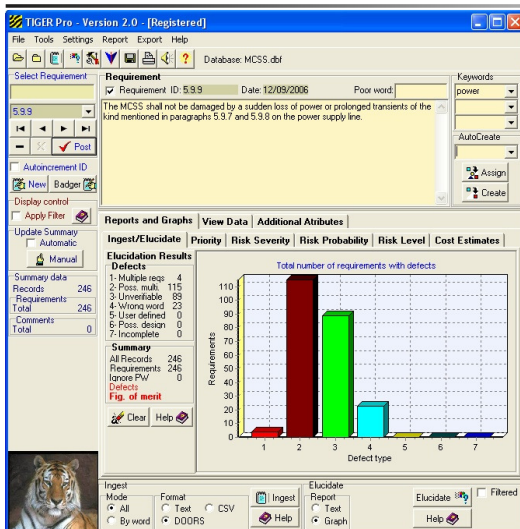
Dealing with poorly-written requirements

- Consider the poorly-written requirement as a poorly-written scenario
- Follow examples already shown in this Module and previous Modules for converting scenarios to functions
 - Break them out into atomic functions (functional requirements)
 - Write requirements traceable to the functions
- Try to discuss them with the stakeholder to get acceptance criteria which will further clarify the meaning
- General principle when dealing with a set of poorly-written requirements – expect missing requirements
- Use the three ways to maximize completeness (Module 4)
 1. Corresponding inputs and outputs
 2. Corresponding functions
 3. Templates

How to write good requirements

0601-101

Pre Exercise 6-1 Tiger Pro demonstration



- User manual is downloadable
- See Additional resources in Knowledge components for links
- Notes
 - No duplicate links
 - Prevents errors if links change in future



How to write good requirements

0601-102

How to write good requirements

Module 7 of 10

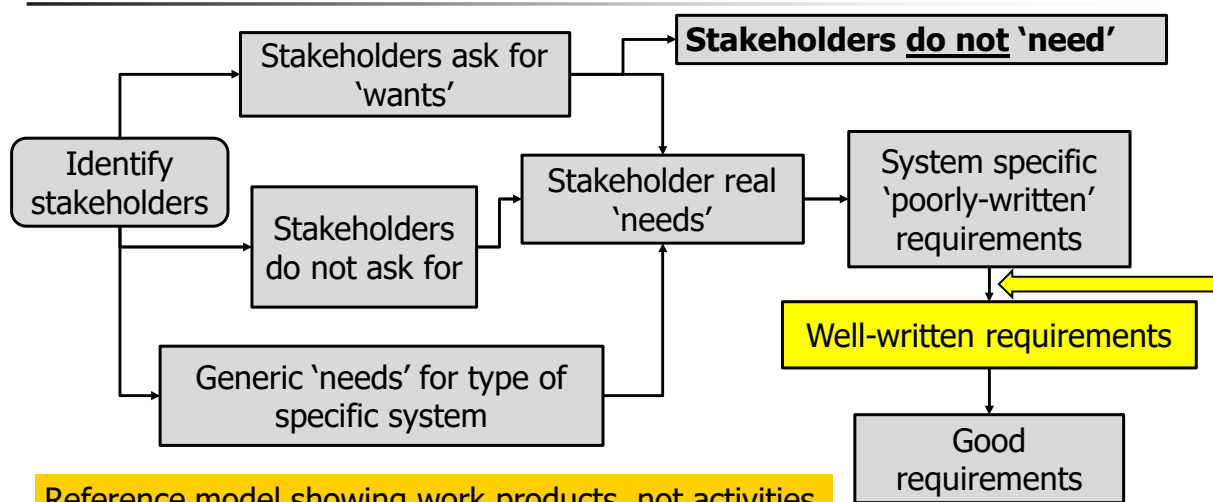
Converting requirements to well-written requirements

Version 1.1.3

How to write good requirements

0701-103

Gap analysis for writing good requirements



The top 5 reasons why you can't write a good requirement

0301-104

Module 7 objectives

#	Objectives	Met
1	To explain the requirements for well-written requirement	18-31
2	To explain the structure of a well-written requirement	33-37
3	To explain the use of spelling, grammar and vocabulary	39-44
4	To explain how to test the text statement for conformance to the specification for a well-written requirement	46
5	To practice what has been taught	15-16, 48-49
6	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	48-49

How to write good requirements

0701-105

9 Requirements for well-written requirements-1

1. A requirement statement shall be atomic (singular)
2. A requirement statement shall be unambiguous
3. A requirement statement shall be verifiable
4. A requirement statement shall use wording consistent with the specification of which it is a part
5. A requirement statement shall be concise (~~adequate~~)
6. A requirement statement shall use the verb "shall" to specify the instruction
7. A requirement statement shall contain the following six elements in the listed order (complete)
 - 7.1 ID number
 - 7.2 Conditions (if any)
 - 7.3 Subject
 - 7.4 Verb (shall)
 - 7.5 Target object
 - 7.6 Qualifiers (if any)

How to write good requirements

0701-106

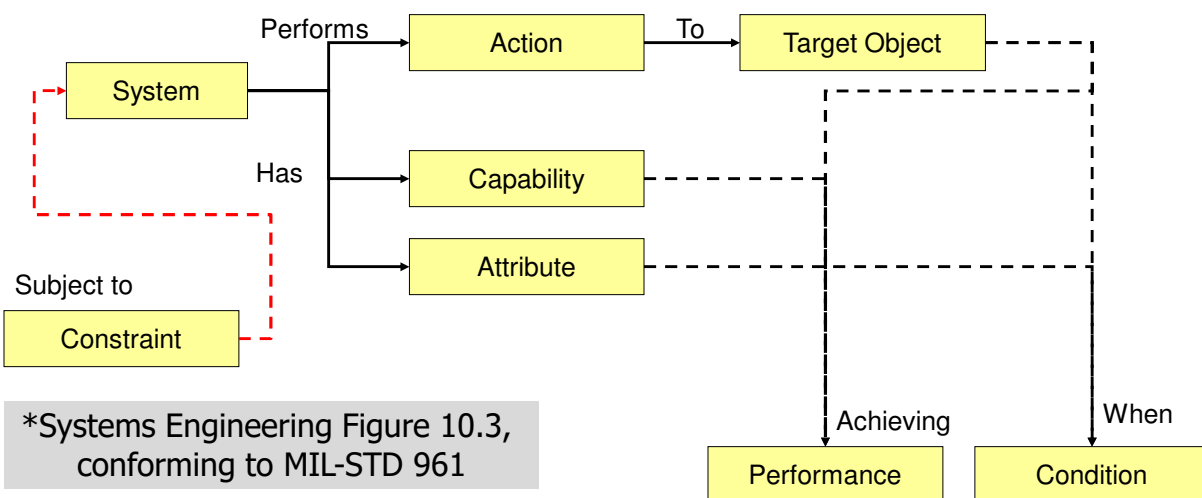
Requirements for well-written requirements-2

8. A requirement statement shall contain the segment "Unless otherwise specified" only if there is at least one exception
9. The wording in a bounding requirement shall reinforce the direction of bounding as follows (no synonyms):
 1. If the boundary is \leq , the wording shall be "less than or equal"
 2. If the boundary is \geq , the wording shall be "greater than or equal"
 3. If the boundary is $<$, the wording shall be "less than"
 4. Depending on the context, if the boundary is $>$, the wording shall be "greater than" or "at least"

How to write good requirements

0701-107

The structure of a well-written requirement*



How to write good requirements

0701-108

Well-written requirements

1.2 Bus operations

1.2.1 Unless otherwise specified in 2, the buses shall operate on their specific routes from 0530 to 2300 daily

1.2...

1.3...

2 Friday and Saturday operations

2.1 On Fridays the buses shall operate on their specific routes from 0530 to 1 hour before sunset

2.2 On Saturdays the buses shall operate on their specific routes from 1 hour after sunset to 2330

How to write good requirements

0701-109

Alternative well-written requirements

1.2 Bus operations

1.2.1 Except on Friday and Saturday the buses shall operate on their specific routes from 0530 to 2300 daily

1.2.2 On Fridays the buses shall operate on their specific routes from 0530 to 1 hour before sunset

1.2.3 On Saturdays the buses shall operate on their specific routes from 1 hour after sunset to 2330

Harder to misinterpret?

How to write good requirements

0701-110

Alternative (better?) well-written requirement

■ 1.2 Bus operations

1.2.1 The buses shall operate on their specific routes according to the daily schedule shown in Table 1

- Sunset times are defined in a separately identified document with version number (and date)
- Even harder to misinterpret
 - Each day is a sub number of 1.2.1.

Table 1

ID	Day	Start	Finish
1	Sunday	0530	2300
2	Monday	0530	2300
3	Tuesday	0530	2300
4	Wednesday	0530	2300
5	Thursday	0530	2300
6	Friday	0530	1 hour before sunset
7	Saturday	1 hour after sunset	2300

How to write good requirements

0701-111

Spelling and grammar

■ Spelling

- What
 - The correct arrangement of letters to form the words you mean to use to communicate concepts
- Why
 - Accurate spelling is essential for effective communication, as incorrect spelling can lead to misunderstandings and confusion

■ Grammar

- What
 - The structure and rules of how words are combined to form sentences and convey meaning
- Why
 - Proper grammar ensures that sentences convey the intended message

How to write good requirements

0701-112




Exercise 7-1

1. In exercise 5.2 you wrote a set of well-written requirements
2. Rewrite the requirements to convert them to better well-written requirements that comply with the requirements for writing requirements
3. Add acceptance criteria to each of the requirements
4. Prepare a <5 minute presentation containing
 1. The well-written requirements (before **and** after performing Step 2)
 2. The acceptance criteria for the requirements
 3. The exercise problem formulated per COPS problem formulation template
 4. A compliance matrix for the exercise
 5. Lessons learned from exercise (<3 min)
 6. Why you think this exercise was included in the lesson
5. Save as a PowerPoint file in format Exercise7.1-abcd.pptx
6. Post/email presentation as and where instructed

How to write good requirements

0701-113



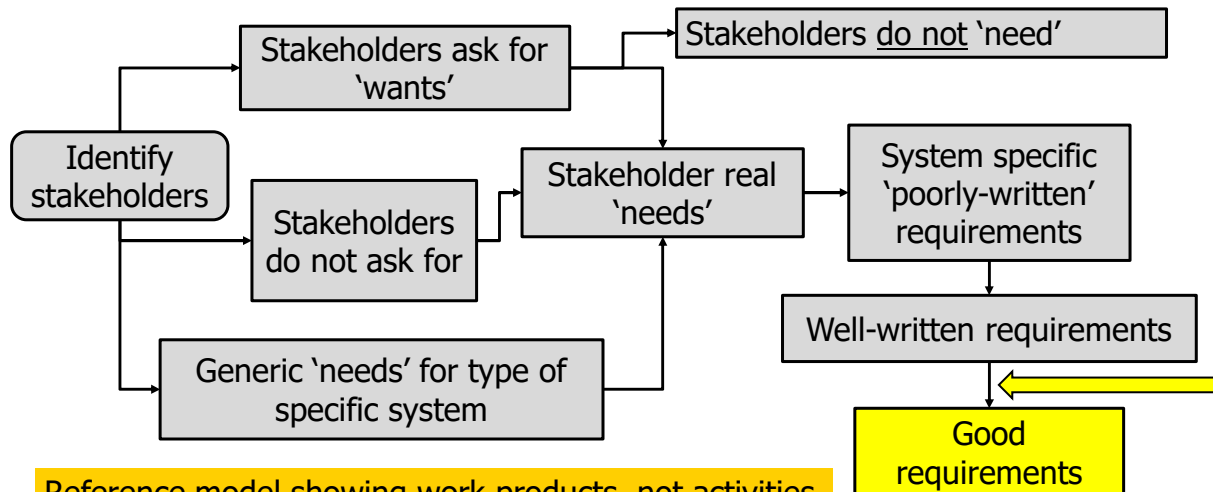
How to write good requirements Module 8 of 10 **Converting well-written requirements to good requirements**

Version 1.0.2

How to write good requirements

0801-114

Gap analysis for writing good requirements



How to write good requirements

0301-115

Module 8 objectives

#	Objectives	Met
1	To explain the additional attributes of good requirements	14-32
2	To explain the requirements for good specifications	34-35
3	To explain the systems approach to writing good requirements	37-41
4	To explain two reasons for requirements changes and prevent one of them	43-45
5	To practice what has been taught	47
6	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	47

How to write good requirements

0801-116



13 attributes of a good requirement

1. Well-written (inherited) – the text sentence (imperative)
2. Real (inherited from need)
3. Acceptance criteria (inherited from need scenario)
4. Agreement on how compliance to the requirement will be verified
5. Traced or linked to a source (owner) via a scenario
6. Prioritized
7. Reason (inherited from need)
8. Unique
9. Consistent
10. Grouped in document (by ID numbers and keywords in database)
11. Technically feasible (inherited from scenario)
12. Affordable (estimated) (inherited from scenario)
13. Achievable by the need-by date (estimated) (inherited from scenario)

How to write good requirements

0801-117



3. Acceptance criteria

- Answers the question "How will we know when the requirement has been fulfilled"?
 - Identifies both acceptance criteria and real need
 - Helps to write both requirement statement and acceptance criteria
 - Can be used to clarify existing poorly written (ambiguous) requirements when planning tests
- If linked from need scenario, then answered the question "How will we know when the need has been met?" (Module 4)

How to write good requirements

0801-118

4. Agreement on how compliance will be verified

- **Stakeholder request**
 - 45.1 Image quality of rangefinder will permit me to see if my car will fit between the posts in time to avoid them
- How shall we know when the requirement request is met?
- Some discussion takes place
 - E.g. how fast are you going, is it on or off road? (ideally in scenario), reaction time
- **Stakeholder's agreed acceptance criterion**
 - An image resolution of $\pm R$ cm at a distance of D meters
- **System requirement**
 - 45.1 The rangefinder shall have an image resolution of $\pm R$ cm at a distance of D
 ± 0.02 meters
 - **Notice how words did not change**
- **How compliance will be verified**
 - By demonstration

How to write good requirements

0801-119

The systems approach to writing good requirements

1. Understand the current situation

1. Understand what is undesirable in the current situation
2. Understand what additional changes, improvements and additions the stakeholders want/wish for

2. The assumptions

1. The description is of an ideal process, the real world is different, so the process will need to be tailored

3. The FCFDS


1. The customer signs off on the System Requirements Specification (SRD) after the System Requirements Review (SRR) **with no changes**

4. The problem (to develop the FCFDS working back from the SRR)

1. At the end of the SRR, the customer signs off on the System Requirements Specification (SRD) **with no changes**
2. Hold the SRR

How to write good requirements

0801-120




The systems approach to writing good requirements

4. The problem (cont.)

3. Circulate the draft SRD and summaries of other documentation to the stakeholders for agreement prior to the SRR
4. Prepare other appropriate documentation for the SRR (e.g., updated plans)
5. Create the draft SRD
6. Verify or ensure the well-written requirements are also good requirements
7. Verify or ensure the requirements are also well-written requirements
8. Verify or ensure the non-functional requirements are written
9. Verify or ensure the other appropriate generic requirements not derived from scenarios are written
10. Write the requirements to perform the quantified functions in the CONOPS scenarios
11. Tag the quantified functions with the properties and attributes of good requirements
12. Expand the scenarios in the CONOPS into atomic quantified functions
13. The customer signs off on the CONOPS at OCR and agrees to proceed with the project

How to write good requirements 0801-121



The systems approach to writing good requirements

4. The problem (cont.)

14. Hold the Operations Concept Review (OCR) or equivalent milestone review
15. Adjust draft CONOPS until customer agrees it is feasible, affordable and can be delivered by the need-by date
16. Verify or ensure the appropriate non-functional attributes are added to the functional and performance scenarios in the draft CONOPS
17. Transform the conceptual reference representation into a draft CONOPS
18. Create a conceptual reference representation of a solution system that will remedy the problem (meet the stakeholder's need)
 1. Without the undesirability in the current situation
 2. With the additional changes, improvements and additions the stakeholders want/wish for
 3. The transition process

Continuum HTP
 Numerology 18 = 'n
 (Chai) = Life (English)

5. The solution is how you implement each step

How to write good requirements 0801-122

Two reasons for requirements changes

1. The stakeholders' , "I forgot to mention"
 1. The [baseline] needs at the start of the project
 1. "get all the [baseline] requirements up front"
2. The need has changed
 1. Since the project (system development process) began
 1. The need has changed
 2. A better understanding of what the need really is
 2. Since the system was placed into service
 1. Latent defects
 2. The need has changed
 3. A better understanding of what the need really is

How to write good requirements

0801-123

Preventing the stakeholders' "I forgot to mention"

- Systemically and systematically identifying all relevant stakeholders
 - Tools include the Nine System Model
- Developing an understanding of the stakeholders' real needs
- Developing an understanding of the stakeholders' domains
 - (1) Problem, (2) implementation and (3) solution domains
 - Needed for identifying generic requirements
 - Needed for identifying 'requests' ('wants') that are not real 'needs'
 - Needed for identifying needs that were not requested
- Holding a dialogue
- Converting assumptions into facts (true or false (reason for false))

How to write good requirements

0801-124

Preventing the stakeholders' "I forgot to mention"

- Creating the draft generic CONOPS reference model of needed solution system
- Converting the draft CONOPS reference model into the specific CONOPS
- Tools include
 - Short restricted attendance meetings (less than 1 hour)
 - Meeting agendas (provided ahead of time)
 - Active brainstorming
 - Active listening
 - The five why's
 - KISS
 - The Principle of Hierarchies
 - Concept maps, Checkland's rich pictures and other sketches
 - Modelling tools

How to write good requirements

0801-125

Exercise 8-1

- In exercise 5-2 you wrote a set of well-written requirements, in exercise 6-1 you ingested them into Tiger Pro, in exercise 7-1 you rewrote them and added acceptance criteria to the requirements
- 1. Verify or ensure the acceptance criteria are in the Tiger Pro database
- 2. Add the priority and risk (both probability and severity) attributes to each of the 7-1 requirements (make some guesses, or select random numbers)
- 3. Display and save the attribute profile graphs, for priority, risk probability and severity
- 4. Create a traditional requirement Risk Matrix for the 7-1 requirements
- 5. Prepare a <5 minute presentation containing
 1. The three attribute profiles for the requirements
 2. The requirement risk matrix
 3. The exercise problem formulated per COPS problem formulation template
 4. A compliance matrix for the exercise
 5. Lessons learned from exercise (<3 min)
 6. Why you think this exercise was included in the lesson
- 6. Save as a PowerPoint file in format Exercise8.1-abcd.pptx
- 7. Post/email presentation as and where instructed

How to write good requirements

0801-126

How to write good requirements

Module 9 of 10

The use of requirements in the rest of the system development process



Version 1.1.2

How to write good requirements

0901-127

Module 9 objectives



#	Objectives	Met
1	To summarize the System Development Process (SDP) and the System LifeCycle (SLC)	12-21
2	To expose participants to the consequences of poorly-written requirements in the SLC,	23-24
3	by explaining the use of requirements in the Needs Identification and System Requirements States	26-27
4	by explaining the use of requirements in the System Design State	29-31
5	by explaining the use of requirements in the Subsystem Construction and Subsystem Test States	33-47
6	by explaining the use of requirements in the System Integration and System Test States	49-51
7	by explaining the use of requirements in the System Operations, Maintenance and Upgrade States	53-62
8	To explain the requirements and change management processes once the initial set of system requirements have been accepted	66-72
9	To discuss the place of requirements in the Agile paradigm (software)	75-79
10	To practice what has been taught	81-82
11	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	81-83

How to write good requirements

0901-128

The Hitchins-Kasser-Massie-Mabelo Framework - (HKM²F)

Complexity	Layer of complexity		A	B	C	D	E	F	G	H
	Global (Planetary)	7								
	Regional	6								
	Socio-economic	5								
	Supply chain	4								
	Business (multiple)	3								
	System (single)	2	System Development Process							
	Product	1								
	Component	0								

Lifecycle States

A – Customer Needs Identification	B – System Requirements	C – Subsystem Design	D – Subsystem Construction	E – Subsystem Testing
F – Systems Integration and Test	G – Operations and Maintenance	H – System Disposal		

How to write good requirements

0901-129

States of the System LifeCycle (SLC)

	State
A	Customer Needs Identification
B	System Requirements
C	Subsystem Design
D	Subsystem Construction
E	Subsystem Testing
F	System Integration & System Test
G	Operations, Maintenance & Upgrade
H	System Disposal

How to write good requirements

0901-130

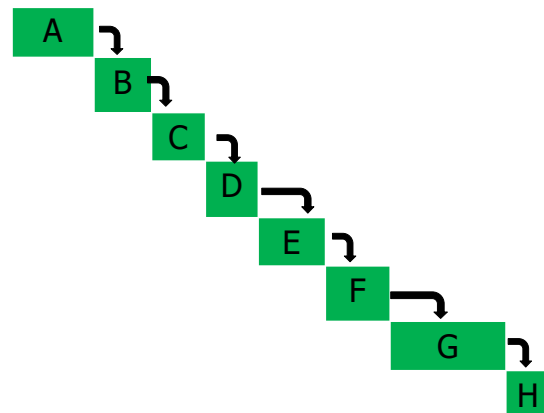
States of the SLC (Gantt chart)

STATE		T _A	T _B	T _C	T _D	T _E	T _F	T _G	T _H
A	Customer Needs Identification								
B	System Requirements								
C	Subsystem Design								
D	Subsystem Construction								
E	Subsystem Testing								
F	System Integration & System Test								
G	Operations, Maintenance & Upgrade								
H	System Disposal								

How to write good requirements

0901-131

States of the SLC (Waterfall)



How to write good requirements

0901-132

States of the SLC (N² chart)

A	X						
	B	X					
		C	X				
			D	x			
				E	x		
					F	x	
						G	x
							H

How to write good requirements

0901-133

States of the SLC (V view)

A						G	
	B				F		
		C		E			
			D				

How to write good requirements

0901-134

Summary of SDP Milestone reviews

- (Start of project)
- **OCR** at the end of the Needs State
- **SRR** at the end of the Requirements State
- **PDR** at the end of the Preliminary design sub-State of the Design State
- **CDR** at the end of the Design State
- **TRR** at the end of the Construction State
- **IRR** at the end of the Subsystem-Testing State
- **DRR** at the end of the System Integration and Test States
- (End of project)
- Engineering
 - Technical reporting
 - Progress, problems and risks
- Management
 - Progress, problems and risks
 - Short term future in detail
 - E.g. SRR to Next Formal Milestone
 - Distant future in less detail
 - E.g. SRR to completion
 - Must be feasible
 - Tasks, what, who, where, when
 - Process risks
 - Excuses
 - Etc.
- Lessons learnt

How to write good requirements

0901-135

CRIP charts

- In the traditional paradigm, we can tell the customer
 - How much money has been spent and if we expect to overrun
 - How much time has elapsed and if we expect to overrun
- BUT, we can't tell the customer how much of the project has been completed
- CRIP charts provide a way to answer the question "*how much of my project has been completed?*"
- Show change in the state of requirements as the SDP progresses
- Reading/video 0902

Range	Identified				In process				Completed				In test				Accepted			
	P	E	A		P	E	A		P	E	A		P	E	A		P	E	A	
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				

How to write good requirements

0901-136

Meetings

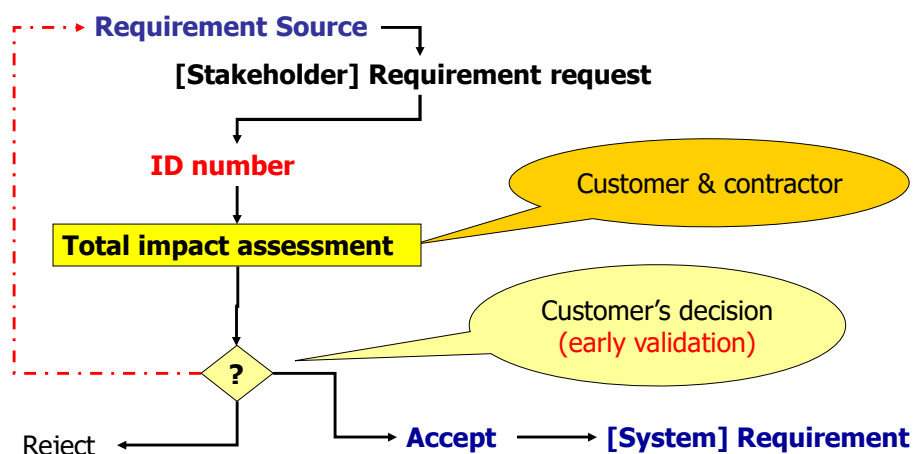
- To discuss the meaning of the poorly-written requirement
- Each meeting
 - Takes one hour
 - Has five people (average)
- Multiply that by the number of poorly-written requirements
- Cost of each meeting = five people hours * (salary + overhead)
- Multiply that cost by the number of meetings over the SDP to discuss all the poorly-written requirements
- Note those people are not working on what they should be working

How to write good requirements

0901-137



Requirements processing



How to write good requirements

0901-138

Example 2 [ST-DADDS]

"204.1 DADS shall automatically maintain *statistics* concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive" (ST-DADS 1992)

- The requirement contains:
 - **Undefined terms:** statistics, piece of media.
 - **Multiple requirements:** and, two sentences.

How to write good requirements

0901-139

Creating four atomic requirements

204.1a DADS shall automatically maintain statistics concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive.

204.1b DADS shall automatically maintain statistics concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive.

204.1c DADS shall automatically maintain statistics concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive has been accessed.

204.1d DADS shall automatically maintain statistics concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive has been accessed.

How to write good requirements

0901-140



Four atomic requirements

204 Access to data sets

204.1a DADS shall automatically maintain **statistics** concerning the **number of times** that each **data set** has been accessed.

204.1b DADS shall automatically maintain **statistics** concerning the **most recent time** that each **data set** has been accessed.

204 Access to each piece of media

204.1c DADS shall automatically maintain **statistics** concerning the **number of times** that each **piece of media** in the DADS archive has been accessed.

204.1d DADS shall automatically maintain **statistics** concerning the **most recent time** that each **piece of media** in the DADS archive has been accessed.

How to write good requirements

0901-141



Four atomic requirements

204 Access to data sets

204.1a DADS shall automatically maintain statistics concerning the number of times that each data set has been accessed.

204.1b DADS shall automatically maintain statistics concerning the most recent time that each data set has been accessed.

Statistics

204.1c DADS shall automatically maintain statistics concerning the number of times that each piece of media in the DADS archive has been accessed.

204.1d DADS shall automatically maintain statistics concerning the most recent time that each piece of media in the DADS archive has been accessed.

How to write good requirements

0901-142

Luz SEGS-1 during constructions/installation



How to write good requirements

0901-143

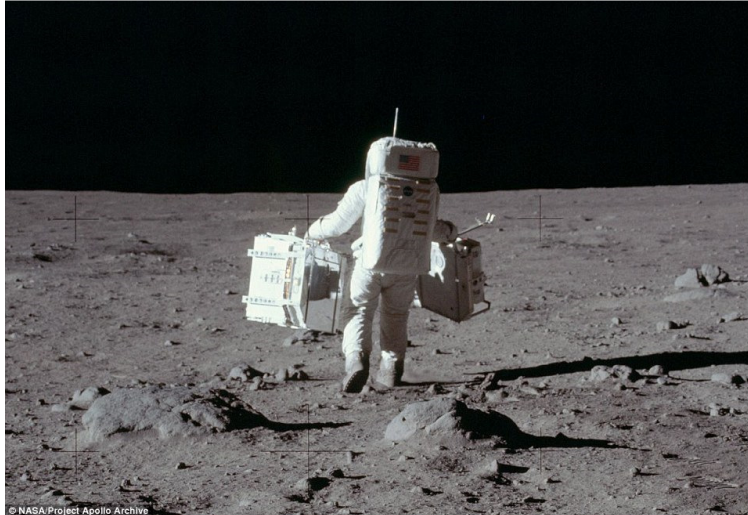
Requirements (written text statements)

- Mirror pointing accuracy = ± 0.2 degrees
- System level
 1. In operation, SEGS-1 shall generate more power than it uses
 2. SEGS-1 shall generate the maximum possible amount of power each day
- Subsystem level
 1. Cable interface specifications
 2. AC power

How to write good requirements

0901-144

Carrying the ALSEP to deployment location

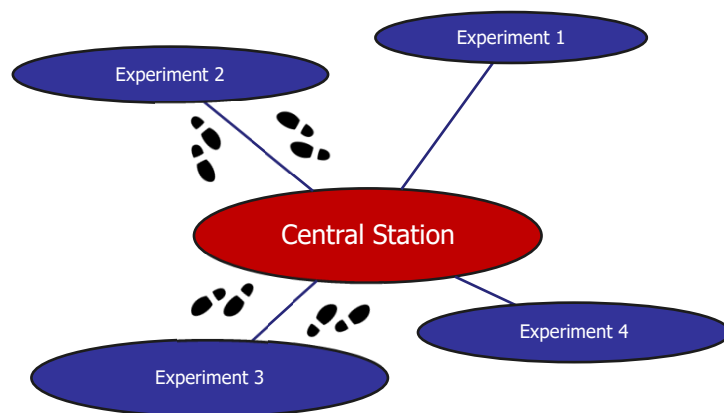


How to write good requirements

0901-145

ALSEP integration scenario

- Unpack from palette shown earlier
- Deploy experiments on surface location per layout plan
- Write procedure to connect cables in a sequence that avoids stepping over a connected cable
- Walk out on one side of cable, walk around (set up) experiment, walk back on other side
- Note simple layout plan representation for stakeholder communications



How to write good requirements

0901-146

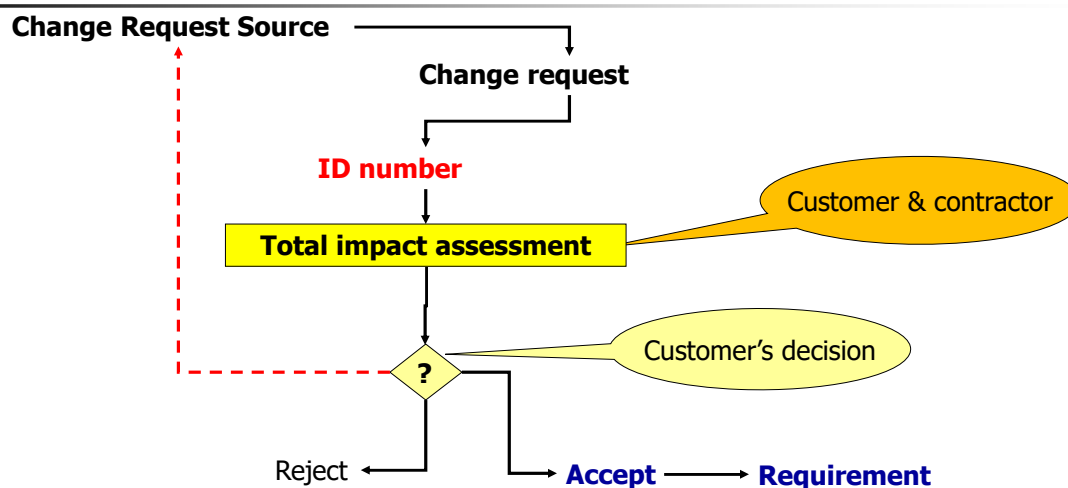
Requirements in System Disposal State?

- It depends
- Complexity of disposal project
- Options (different requirements)
 - Abandon
 - Walk away and leave in place
 - Dump
 - Transport to a facility that can salvage or store components
 - Sell
 - Find a third party who will purchase the system
 - Contractor
 - Pay someone to dispose of system
 - outsource problem
 - Others
 - Not mentioned above, combinations, etc.

How to write good requirements

0901-147

Change Processing

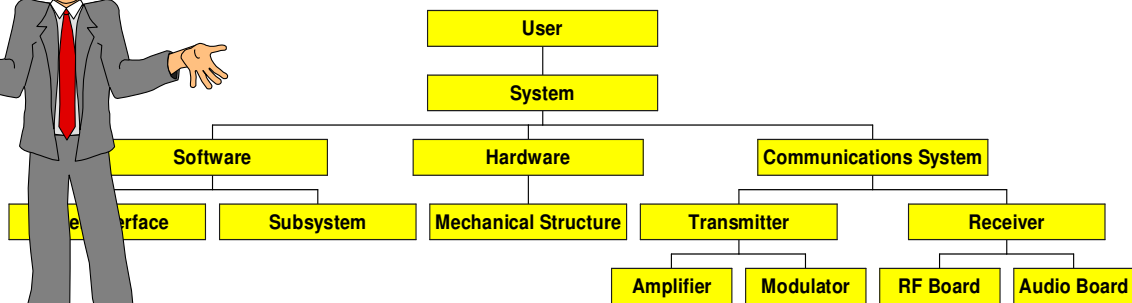
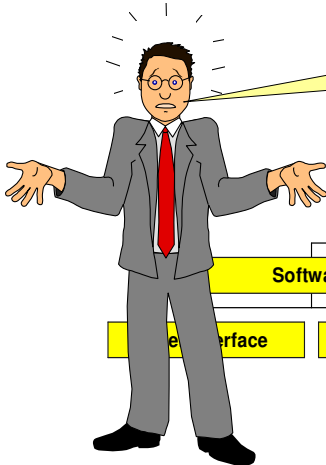


How to write good requirements

0901-148

Importance of the Requirements Traceability Matrix

Now what does that change impact?



How to write good requirements

0901-149

Waterfall vs. Agile (*Continuum* HTP)

Waterfall

- (Traditional SLC)
- Used in **bespoke (single customer)** system development
- Builds to **specification**
- Works well, when the baseline needs are well-defined **and** the needs do not change (or change slowly) during the SDP
- Some needs may change during O&M state
 - Waterfall is iterative once system is deployed (cataracts) in O&M state

Agile

- Used in **mass market (many customer)** (systems and) software product development
- Builds to **speculation** (hope people will buy it)
- **No O&M state once product is released**
- Instead, product is upgraded
 - To fix defects
 - Add functions (features)
- Kano model applies
- **May also be misused for physical system development**

How to write good requirements

0901-150

Exercise 9-1

1. Consider the two requirements in the previous page (503.1 and 513.2)
2. You are the Change Control Board (CCB), what is the impact of the change request?
3. Prepare a <5 minute presentation containing
 - 1) This slide
 - 2) The assumptions you made
 - 3) A summary of the impact of the change
 - 4) Accepted or rejected
 - 5) If accepted, the changed requirement(s)
 - 6) A compliance matrix for the exercise
 - 7) Lessons learned from exercise
 - 8) The problem posed by the exercise formulated per COPS problem formulation template
4. Save as a PowerPoint file in format Exercise9-1-abcd.pptx
5. Post/email presentation as and where instructed

How to write good requirements

0901-151

Module 10 topics

- Summary and highlights of Modules 0-9
- **Overall summary of course construction**
- Module 10 exercises



How to write good requirements

1001-152

Course architecture

- Optimized for maximum learning using the balanced classroom
- Designed for various learning styles
- Live lectures
- Live weekly discussion and question and answer Modules
- Readings for in-depth knowledge
- Practical exercises
- Knowledge reading exercises
- Lots of feedback to keep learning on track
- High participant workload in weekly Module format
- Minimal experiential time

Writing good requirements

1001-153

The exercises

- Lectures gave you the context
- Exercises gave you the experience
- Learning was in the exercises and comments
- In-depth analysis of exercises (time permitting) brought out experiential learning points, many of which will never show up in a regular class
- Most graduate and continuing education classes focus on either (a) remembering (and understanding) or (b) applying it, and © ignore thinking about it
 - Good instructor evaluations
 - Students are not taught what they need to know



How to write good requirements

1001-154

Workflow analysis perspective

- Reason for perpetuation of the perennial problem of poorly-written requirements is that courses do not teach requirements engineers what they need to know and do to create good requirements (*Scientific* HTP)
- The course filled gaps that other courses haven't identified, including
 1. A systemic and systematic approach to finding stakeholders
 2. The need for a reference model to maximize the completeness of requirements
 3. The three representations (1) current, (2) CONOPS (FCFDS) and (3) transition
 4. Well-written requirements can be useless or even bad
 1. E.g., self-destruct, rather than erase data
 5. The three domains, (1) problem, (2) solution and (3) implementation
 6. Generic and specific stakeholder needs
 7. Differentiation between baseline requirements and those resulting from changes in needs
- By applying systems thinking to the problem
- Workflow analysis build scenarios (CONOPS)

How to write good requirements

1001-155

Module 10 topics

- Summary and highlights of Modules 0-9
- Overall summary of course construction
- **Module 10 exercises**



How to write good requirements

1001-156



Exercise 10-1

1. Fill out the course evaluation form (1002)
2. Save as a Microsoft Word file
3. Post/email presentation as and where instructed

How to write good requirements

1001-157



Exercise 10-2 Self evaluation

1. Read the requirements in reading 0102
2. Comment on at least 15 requirements in the reading
 - Are they "good" or "bad", and why
3. Prepare <5 minute PowerPoint presentation containing
 1. Reformulated problem per COPS Problem Formulation Template
 2. The requirements you chose to comment on
 3. Your comments
 4. A compliance matrix for the exercise
 5. Lessons learned from exercise
4. Save as a PowerPoint file in format Exercise10.2-abcd.pptx
5. Compare it with the presentation you saved as Exercise1.2-abcd.pptx

How to write ~~good~~ requirements

0101-158



Meeting the objectives

#	Objectives	Met
1	To review the course Modules and the learning	7-155
2	To present an overall summary of the course construction	160-162
3	To request overall feedback via the Course Evaluate Form (1002)	171
4	To provide the opportunity to exercise the 5 levels of knowledge in the updated Blooms taxonomy	171-172

How to write good requirements

1001-159