Some prior knowledge of systems and software engineering and project management required

Let's just stop writing requirements, we can't write them properly anyhow

Prof Joseph E Kasser, DSc, CEng[2], CM, FIET, FIES, VK5WU, G3ZCZ

https://therightrequirement.com

Webinar presentation to the BCS

Postponed from Wednesday 21 Sept 2022 at 5pm London time

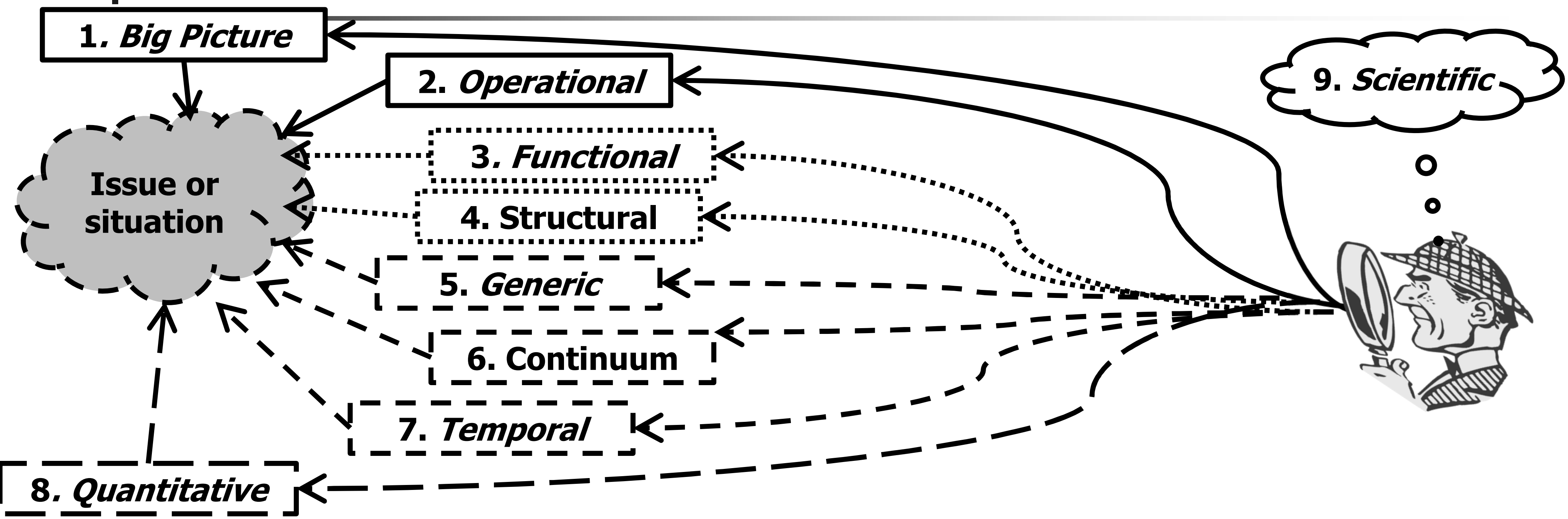
Hidden (notes) slides in handout

Version 1.0.2

Topics

- The perennial problem of poor requirements exists despite numerous efforts to tackle the problem
  - In systems, software engineering and IT projects
- The complex and complicated requirements generation process
- Applying systems thinking to introduce an innovative solution
  - Object-Oriented Systems Engineering and how it might work

Systems thinking : The Holistic Thinking Perspectives*



External	Internal	Progressive	Remaining
1. Big picture	*3. Functional*	*5. Generic*	*8. Quantitative*
2. Operational	*4. Structural*	*6. Continuum*	*9. Scientific*
		7. Temporal	

Attributes of a Requirement*

- **F**easible
- **A**tomic
- **C**omplete
- **T**raceable
- **S**ufficient
- **V**erifiable (NOT necessarily testable)
- **A**dequate**
- **C**onsistent
- **U**nderstandable**
- **U**nambiguous
- **M**anageable

- Known for 40+ years
- So many poorly written requirements are still being produced

* Compiled from various sources

Big Picture Background

- The perennial problem of poor requirements still exists
- Ivy Hooks presented ‘Writing Good Requirements’ at the INCOSE International Symposium in 1993
- International Requirements Engineering Board (IREB)
  - Founded in Fürth in Germany in October 2006
  - Offers the Certified Professional for Requirements Engineering (CPRE) qualification
- INCOSE Requirements Working Group rediscovering the 20^{th} Century world of requirements
- Lots of Standards, books, classes, trainers, workshops, etc.
- INCOSE’s Model Based Systems Engineering (MBSE) is building complex and complicated models to produce a better set of poorly worded requirements
- Various requirements management tools that can’t tell us if the requirement is poorly-written

Sample of Standards and guides for requirements documents (1987-2003)

- MIL-STD 961E, 2003
- IEEE Std 830-1984 Software Requirements
- BS 6719:1986 Specifying user requirements for a computer-based system
- CSA Z243.15.4-1979 Basic Guidelines for the structure of Documentation of System Design Information
- DOD-STD-2167 Defense System Software Development
- DOD-STD-7935A DOD Automated Information Systems Documentation Standards (15/2/1983)
- The STARTS Guide, Second Edition 1987

Lessons learned include

- The requirements process has built-in failure elements
- People can’t always communicate their needs
- People don’t always know the cost of meeting a stated need
- People state requirements in their language
- Analysts, customers, users, developers, and other stakeholders don’t always speak the same language.
- Requirements from different sources may conflict
- Organisational and political issues drive programmatic and system requirements
- Stated requirements may not be real
  - hidden agendas, or not thought through
- Requirements change over time
- Priorities change over time
- There is a difference between what stakeholders ask for and what they really need

The real requirement (need)

- People don't always state the real requirement (need)
- If he'd had some acceptance criteria or a model of the solution situation this situation would never have appeared

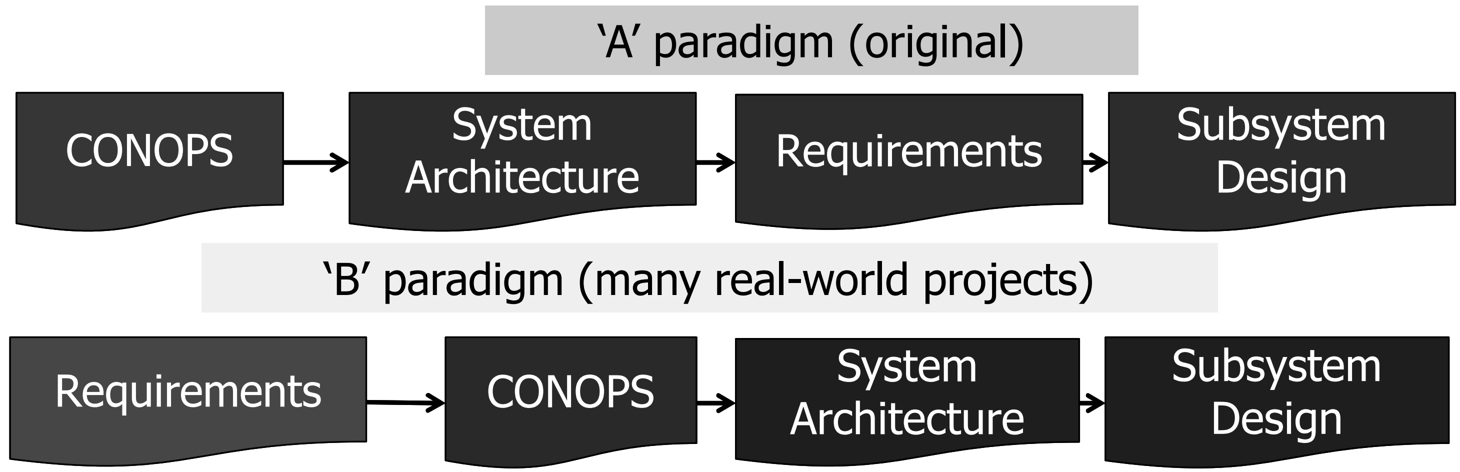
Requirements Management 2022

- We ignore situations in which something cannot be specified by requirements and use alternatives
  - E.g. kitchens, tastes
- The perennial problem of poor requirements is still with us
- **Nothing has changed in 40+ years**
  - **Other than tools have become more subjectively complicated**
- "The definition of insanity is repeating the same mistakes over and over again and expecting different results, …"
  - Attributed to Albert Einstein
  - Narcotics Anonymous, Literature subcommittee, World Service Conference of Narcotics Anonymous, 1981, page 11, https://web.archive.org/web/20121202030403/http://www.amonymifoundation.org/uploads/NA_Approval_Form_Scan.pdf, accessed 2 Dec 2021.

Two requirements paradigms

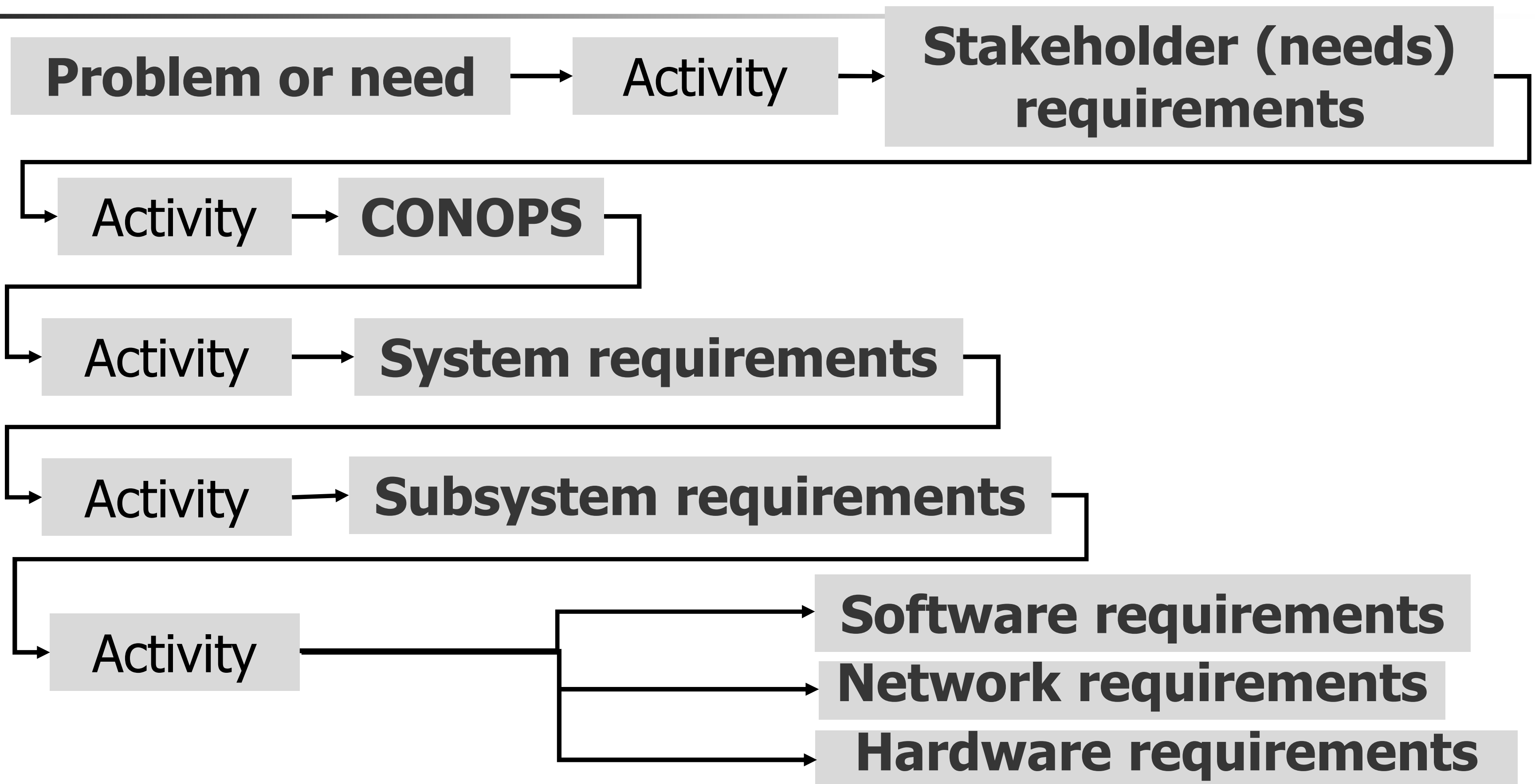
- The “A” Paradigm
  - Understands the problem and create a CONOPS or OCD or MODEL representing the solution
  - Then create a matched set of system and subsystem specifications
  - Original systems engineering of the 60’s
    - Successful projects characterized by common vision of future desirable situation
  - Creates/architects a process to realize the solution
    - Biemer and Sage, 2009, page 153, Kasser and Palmer, 2005
      - The SEMP or Project Plan
- The “B” paradigm
  - Requirements are one of the inputs to the ‘systems engineering process’ (SEP)
  - The SEP produces a CONOPS or MODEL from the requirements
  - Taught in most systems engineering courses
    - (Martin, 1997) page 95), (Eisner, 1997) page 9), (Wasson, 2006) page 60) and (DOD 5000.2-R, 2002), pages 83-84)
  - Follows the SEP

‘A’ and ‘B’ paradigms



- Shown as a linear flow for educational purposes
  - E.g. an infeasible requirement or an approved change may modify the CONOPS which would be shown as a confusing feedback loop
- Constraints (legal, etc.) should also drive CONOPS and system architecture in both paradigms
- System architecture may change during subsystem design

Producing requirements [-]



Ask any reliability engineer

- What is the reliability of serial communications path (R)?
  - $R = R_1 * R_2 * R_3 * \ldots *R_n$
- How to improve reliability?
  1. Strengthen the reliability of the elements in the path
     - Education and training not having much effect
     - Model-based Systems Engineering (MBSE) helping to visualize the needed solution but does not contribute to producing well-written requirements
  2. Shorten the path without losing information
     - Concurrent engineering
  3. Use a different path to carry the information
     - Higher reliability?
     - Fewer opportunities for errors in communication?

Essence of systems engineering *

"The essence of systems engineering is in choosing the right parts, bringing them together in the right way, causing them to interact in the right way, and in orchestrating those interactions to create a unified whole that performs with optimum effectiveness in its operational environment, so solving[1] **the problem** that prompted its creation."

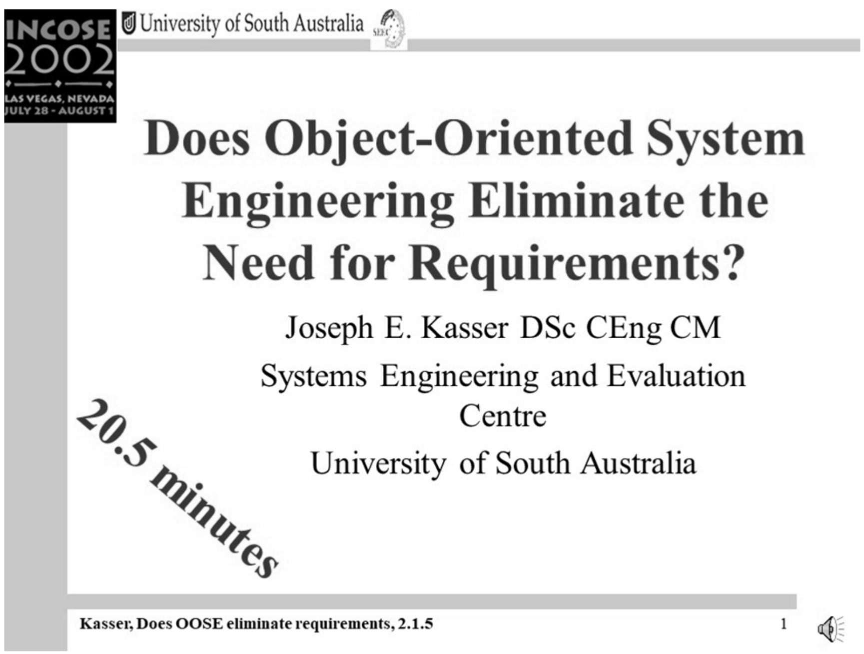
[1] Replace 'solving' with 'remedying'

- Problems can be solved, resolved, dissolved or absolved (Ackoff, R)
- Remedy can be any one or a combination of any of Ackoff's four ways

* Hitchins, Derek.

So what's the alternative?

Dissolve the problem



Introducing Object-Oriented Systems Engineering

and project management

Bringing MBSE into the 21st Century

and

systems engineering

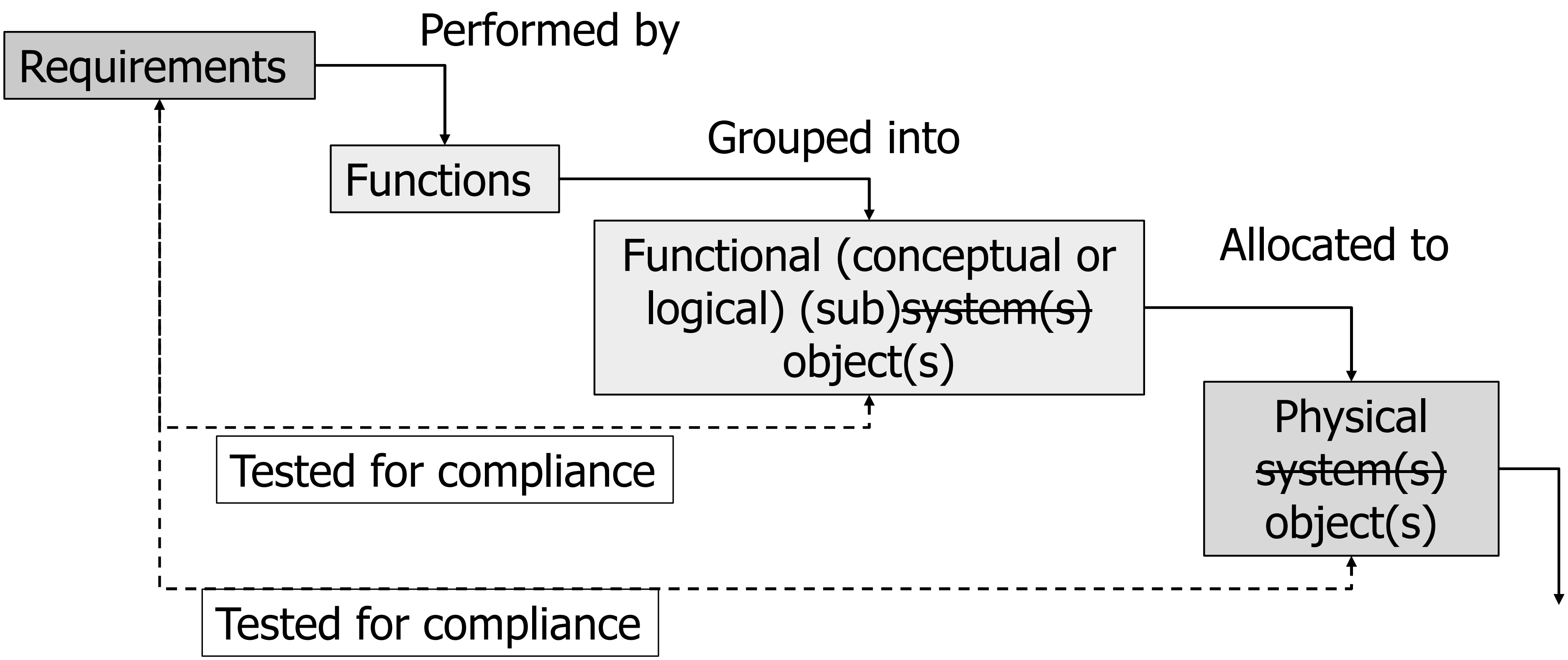
back

into the A paradigm

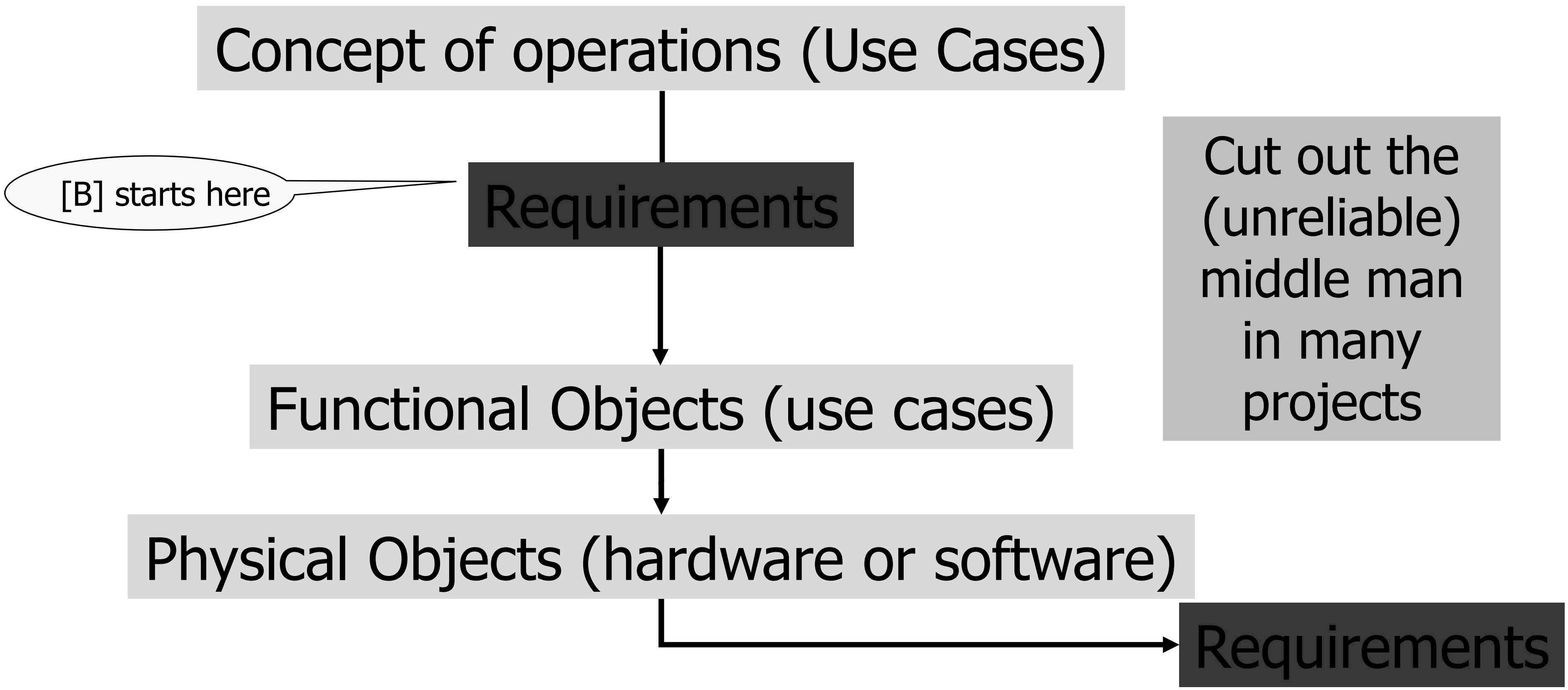
Replacing requirements

- Text-mode requirements are a communications tool adopted in the stone age
- Many current problems are more complex and involve multiple stakeholders
- Computers have provided tools and techniques that can serve as alternate tools
  - Example, kitchen design
- Use an object-oriented approach to replace text-mode requirements with properties

Traditional design paths [B]



Traditional design paths [A]

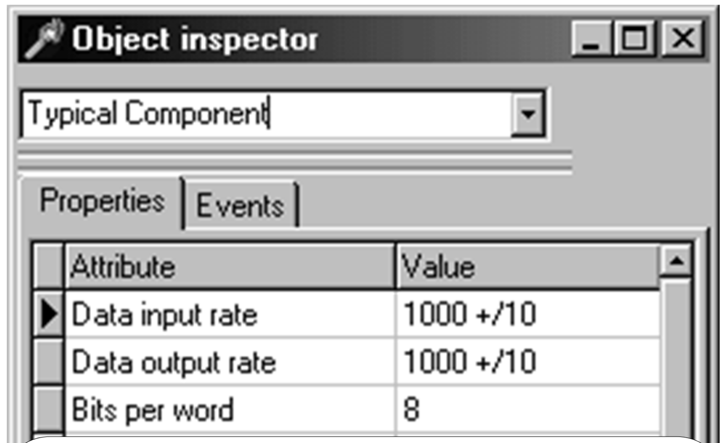


Properties of a communications function

A function with numbers is a requirement

A requirement is a function with numbers

Communications function with numbers

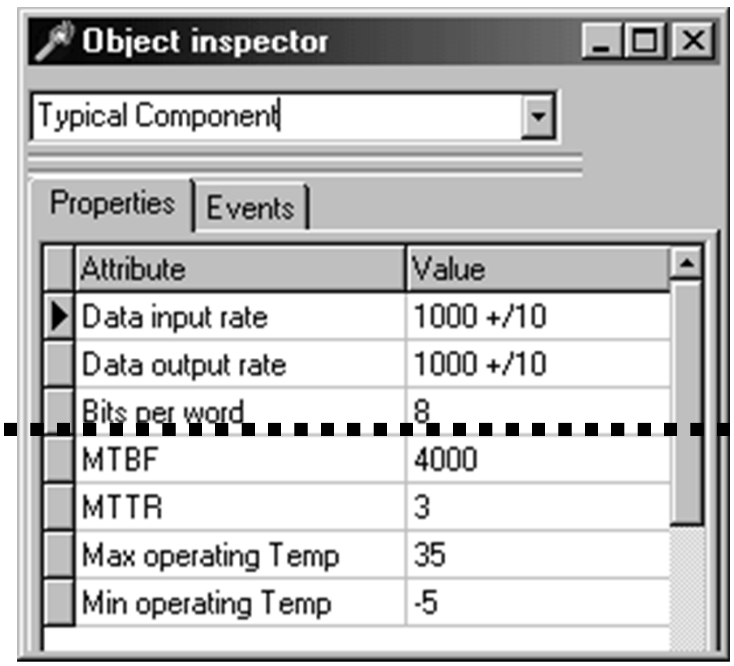


- Add non-functional properties to the function
- i.e. Priority, MTBF, Reason function is being done, verification method, risk

Object-component perspective

Functional

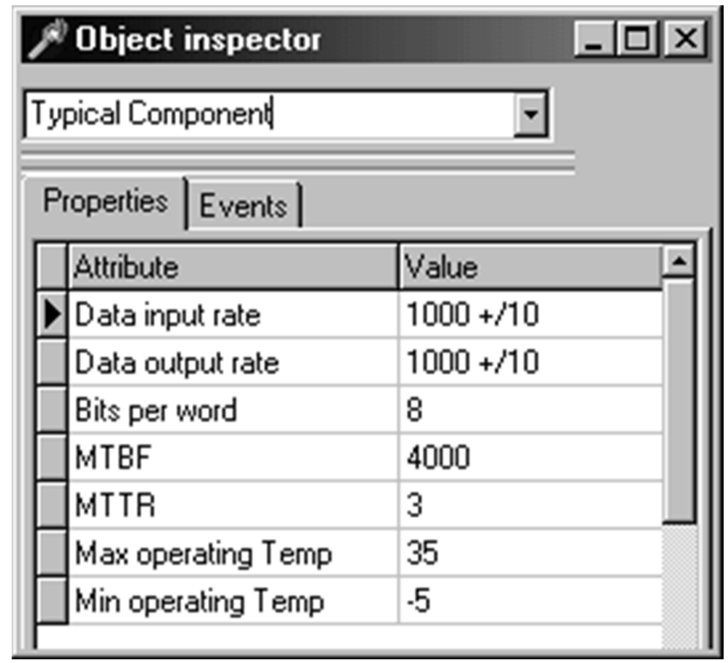
Non-functional



Could an expanded version capture the entire performance of the component?

Attributes of a Component

- **F**easible (achievable within constraints)
- **A**tomic
- **C**omplete
- **T**raceable
- **S**ufficient
- **V**erifiable (NOT necessarily testable)
- **A**dequate**
- **C**onsistent
- **U**nderstandable**
- **U**nambiguous
- **M**anageable



* Compiled from various sources

Specific and generic properties

Stakeholder
(wants and needs)

Solution specific

Models (use part of database)

Other sources

1. Government mandates
2. Lessons-learned documents (experience)
3. Similar systems (*Generic* perspective)
4. Legacy system user manuals
5. Analysis and extrapolation
6. Other appropriate places

- Solution generic
- Inheritable
- Improves completeness and probability of meeting the needs

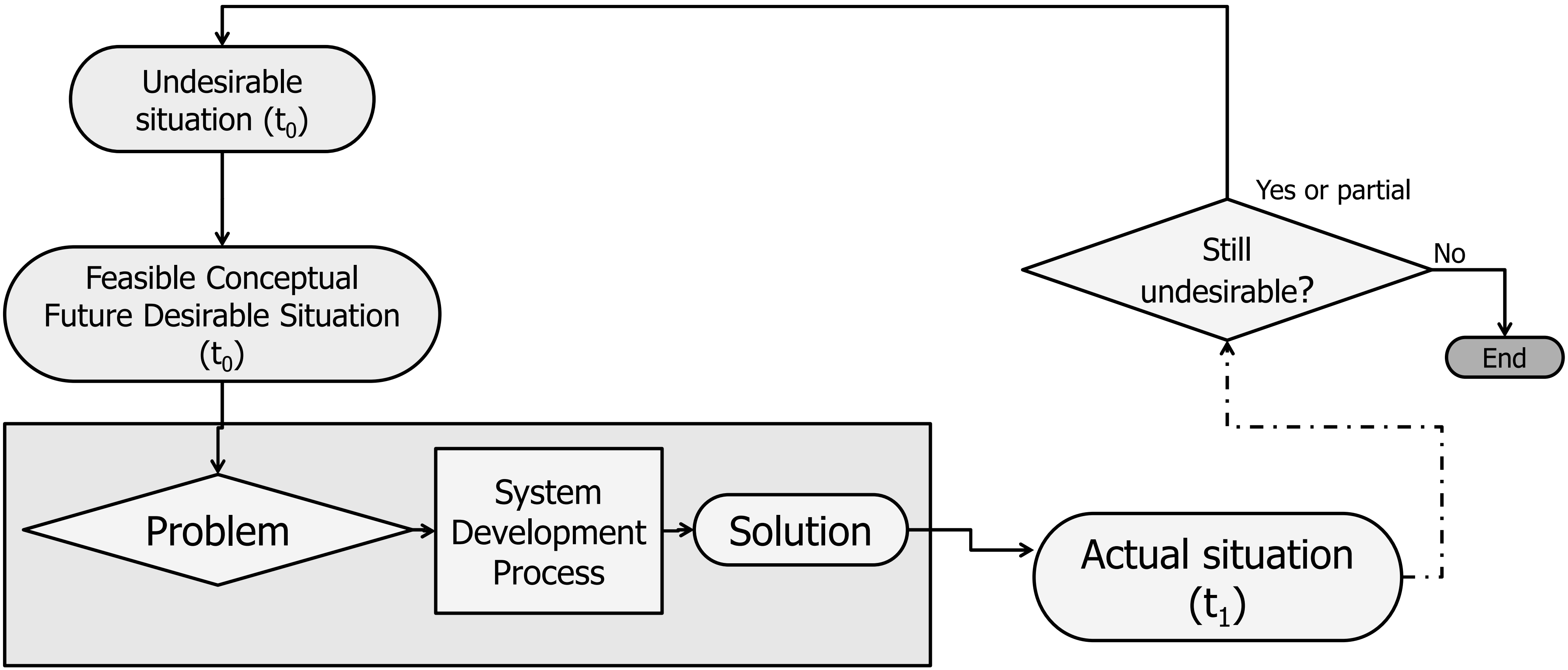
Object-Oriented Systems Engineering

- MBSE on steroids
- Text-mode requirements-free
- Models are only a part of OOSE using data an Integrated Information Environment (IIE)
- All personnel and tools access information in the same database
- Integrates product and process information in the IIE
- Can readily provide smart features not available in current system, software and IT development paradigms
- Tremendous potential for tool creators and vendors
- Follows the extended problem-solving process

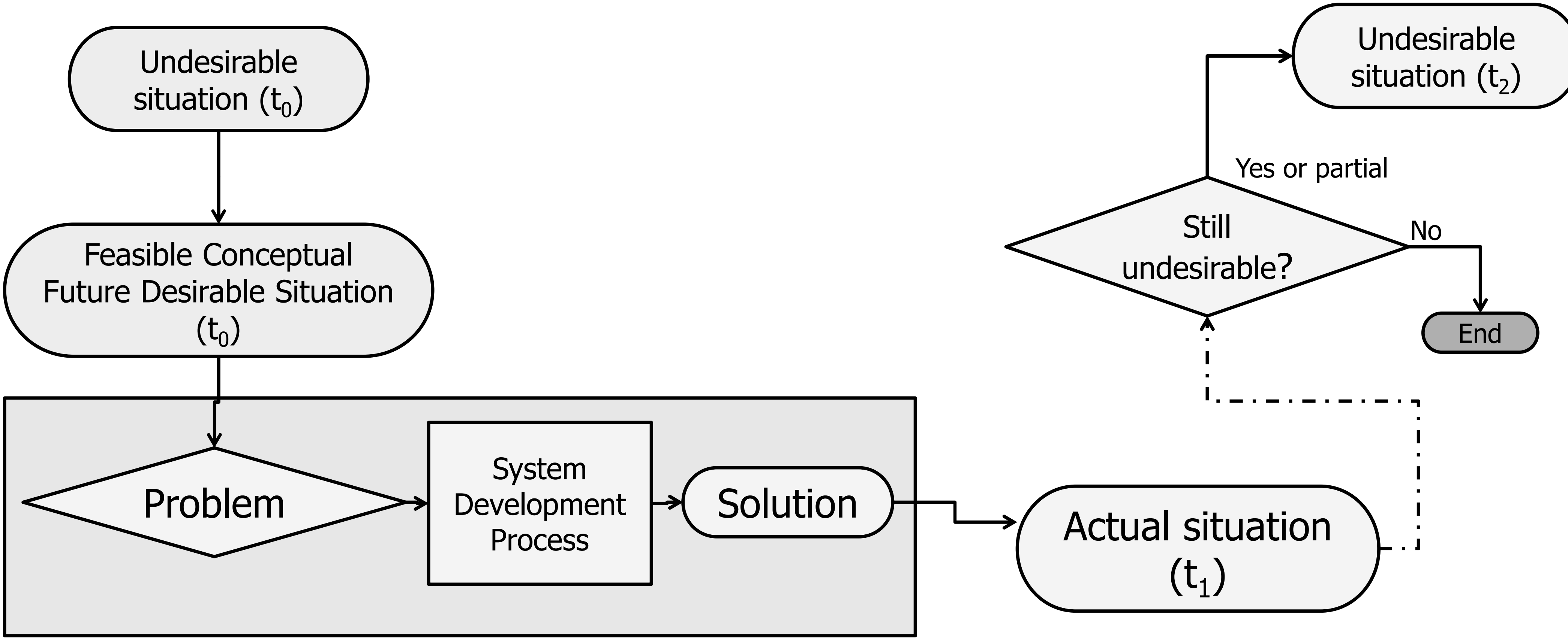
MBSE on steroids

- **S**ystems
- **T**hinking
- **E**liminates
- **R**equirements
- **O**bjectively
- **I**ncreases
- **D**elight
- **S**atisfaction

The extended problem-solving process (*Functional*)

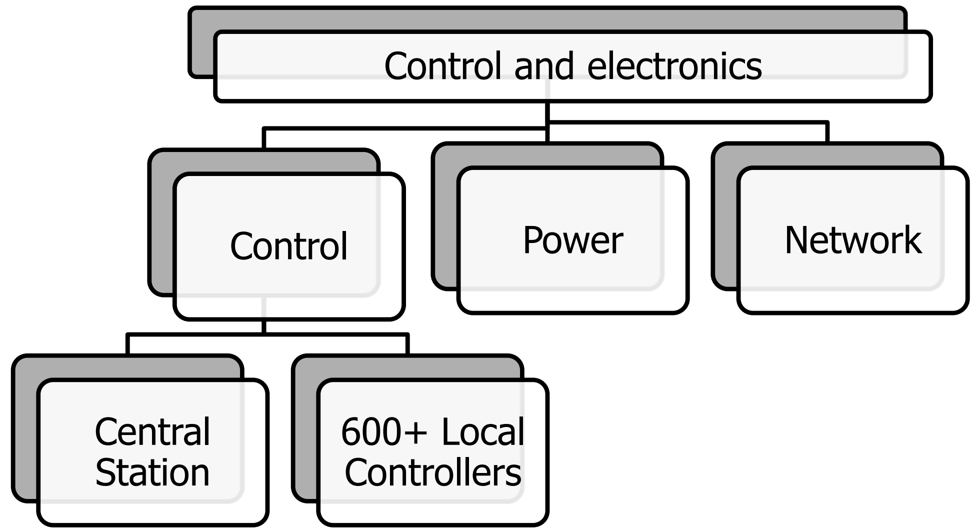


The extended problem-solving process (*Temporal*)



System physical architecture

- Traditional SE
  - Matched set of requirements
    - With linkage traceability
  - Product Based Structure (PBS) ~ Work Breakdown Structure (WBS)
- OOSE
  - Matched set of object properties
    - With linkage traceability
  - Product Based Structure (PBS) = Work Packages (WP)
  - No gap between system and software architectures



Object-Oriented Systems Engineering

- Can also be taught using the Waterfall educational view or baseline project Gantt chart
- The four early states in the OOSE System Development Process (SDP)
  - A. The Problem Discovery State
  - B. The Solution Conceptualization State
  - C. The Preliminary Architecture State
  - D. The Architecture Selection State

Problem with meaning of “problem”[h]

1. A question proposed for solution or discussion (dictionary.com, 2013).
2. Any question or matter involving doubt, uncertainty, or difficulty (dictionary.com, 2013) For example:
   - ***An undesirable situation***. You might hear someone end a sentence with “… and that’s the problem” when they should be saying “… and that’s the undesirable situation”.
   - ***The underlying <u>cause</u> of an undesirable situation***, usually a failure of some kind.
     - “*my phone stopped working; the **problem** was a discharged battery”*.
     - the **cause** of the phone stopping working was a discharged battery;
     - the **symptom** or effect was that the phone stopped working.
3. The need to determine the necessary sequence of activities to perform the transition from an undesirable situation to a FCFDS (Schön, 1991).

A: The Problem Discovery State

- "Problem" - meaning undesirable aspect of the situation
- Creates an "as-is" model of the undesirable (problematic) situation
  - Identifies the symptoms of the undesirability
- Documents properties of functions performed by entities in the model
- Checkland's Soft System Methodology is a useful tool in this state
- Terminates at a Problem Understanding Review (PUR)

A: The Problem Discovery State[h]

- Creates an "as-is" model of the undesirable (problematic) situation
  - Viewed from the appropriate eight descriptive Holistic Thinking Perspectives (HTP)
  - Focus on the *Big Picture*, *Operational*, *Functional* and *Structural* perspectives
  - Identify the symptoms of the undesirability
- Documents properties of functions performed by entities in the model
- Does not put too much effort in determining root causes of undesirability at this time (*Scientific* perspective)
  - Stakeholders may disagree on them
- Documents known or suspected causes, if any, as assumptions
- Documents any other assumptions
- Identifies
  - Amount willing to pay for solution (may or may not be realistic at this time)
  - Need-by operational date for operational solution
- Terminates at a Problem Understanding Review (PUR)

B: The Solution Conceptualization State

- Creates a "(would-like) to-be" functional model of a Feasible Conceptual Future Desirable Situation (FCFDS) or CONOPS or Operations Concept Harbinger (OCH)*
- The undesirable situation
  - Without the undesirability
    - This is where the root causes need to be determined
  - With improvements
  - With additional functionality
- Documents properties of functions performed by entities in the model
- Terminates at a Solution Concept Review (SCR)

* Kasser J.E., Cook S.C., Scott W, Clothier J., Chen P., Introducing a Next Generation Computer Enhanced Systems Engineering Tool: The Operations Concept Harbinger, *proceedings of the Systems Engineering Test and Evaluation (SETE) Conference,* Sydney Australia, 2002.

B: The Solution Conceptualization State [h]

- Creates a "(would like) to be" functional model of a Feasible Conceptual Future Desirable Situation (FCFDS) or CONOPS or Operations Concept Harbinger (OCH)*
- The undesirable situation
  - Without the undesirability
    - This is where the root causes need to be determined
  - With improvements
  - With additional functionality
- Viewed from the appropriate eight descriptive Holistic Thinking Perspectives (HTP)
  - Focus on the *Big Picture*, *Operational* and *Functional* perspectives
- Documents properties of functions performed by entities in the model
- Prioritizes the functions performed by the model
- Documents all assumptions
- Terminates at a Solution Concept Review (SCR)

C: The Preliminary Architecture State

- Creates more than one candidate physical architecture
  - At least two, ideally three – but depends on scope of project
- Maps properties of functional object to properties of physical objects
- Physical objects contain hardware and software
  - No gap between system and (OO) software architectures
- Traditional systems engineering activity
  - Does NOT produce a matched set of requirements
  - Does produce a match set of objects together with properties
- Terminates at a Preliminary Architecture Review (PAR)

C: The Preliminary Architecture State [h]

- Create more than one candidate physical architecture
  - At least two, ideally three
- Maps properties of functional object to properties of physical objects
- Traditional systems engineering activity
  - Analysis of Alternatives (AoA) part 1
  - Does NOT produce a matched set of requirements
  - Does produce a match set of objects together with properties
- Create a rough estimate (guess) of cost and schedule to realize each desired physical object
- Create selection criteria to select a candidate
  - E.g., cost, need-by date, solution risk profile, constraints due to need to interface with the existing systems when this system is deployed, production risks
- Perform rough feasibility study on each architecture to identify risks, ensure feasibility of asked-for properties
- Terminates at a Preliminary Architecture Review (PAR)

D: The Architecture Selection State

- Selects a candidate solution
  - Standard decision making process
- Creates
  - Systems Engineering Management Plan (SEMP) or
  - Project Implementation Plan (PIP)
  - Test and Evaluation Master Plan (TEMP)
  - for the production and verification of the physical objects (subsystems)
- Terminates at a Selected Architecture Review (SAR)

D: The Architecture Selection State [h]

- Selects a candidate solution
  - Standard decision making process
- Analysis of Alternatives (AoA) part 2
- Creates Systems Engineering Management Plan (SEMP) or Project Implementation Plan (PIP) and Test and Evaluation Master Plan (TEMP) for the production of the physical objects (subsystems)
  - Cost based on willingness to pay for functionality
    - Too expensive, remove low priority functions
  - Schedule based on meeting needed operational date
    - Working back from date using just-in-time calculations
- Ending milestone review
  - Reviews properties of the set of objects to see if they meet the need as of the time of the review
- Terminates at a Selected Architecture Review (SAR)

Rest of system life cycle (SLC)

- Traditional
  - but manages properties not requirements

E. Subsystem Construction States

F. Subsystem Testing States

G. System Integration and Testing State

H. Operations and Maintenance (O&M) State

1. Placement into service
2. In-service O&M change management
3. In-service upgrades

I. System Disposal State

SDP design

- Stakeholder needs are converted to functions with properties
- Design for more than one iteration depending on complexity and technology availability
  - Extended problem-solving process
  - Traditional software Build Planning
  - Better risk management
- Each iteration satisfies parts of the need
  - Due to budget, need-by date, political constraints, etc.
  - Allows for changes during SDP
- Can always complete ahead of schedule

Variations on a theme

Properties	Wanted		

- Helps with exceeding expectations

Property Traceability Matrix (PTM)

Properties	Wanted	Willing to pay (WTP)	Actual (Verified)

- Properties
  - Identified in the Problem Discovery State
- Wanted
  - Identified in CONOPS or model of FCFDS (influences decisions during SDP) in the Solution Conceptualization State
- Willing to pay for
  - Identified by selection of candidate in the Architecture Selection State
- Actual (Verified)
  - After Operational Test and Evaluation (OT&E) in operational environment

Properties of system objects (partial)

Identification number	Reason function is being done in solution system
Name of function performed by object	Inputs to function
Priority	Willingness to pay for function (higher levels)
Narrative of function performed	
	Traceability forward to (component, document)
Products (outputs)	Risks (probability, seriousness)
Acceptance criteria for objects	Traceability back to (specific need or general)
	Lower level object ID's (if any)
	Assumptions not stated elsewhere

- Contains both systems engineering (products) and project management (process) information
- Integrated Information Environment (IIE)

Work Packages to create system objects (partial)

Identification number	Reason activity is being done in SDP
Name of Work Package	Prerequisites (products or milestones)
Priority	Resources (people, equipment, material)
Narrative of activity	Internal key milestones (if any)
Schedule (+ accuracy)	Decision points (if any)
Products (outputs i.e. system objects)	Risks (probability, seriousness, mitigation WP ID)
Acceptance criteria for objects	Traceability (source of work)
Estimated cost of activity	Lower level Work Package ID's (if any)
Accuracy of cost estimate	Assumptions not stated elsewhere

- Work packages are process objects corresponding to physical or logical objects
- Contains both systems engineering (products) and project management (process) information
- Integrated Information Environment (IIE)

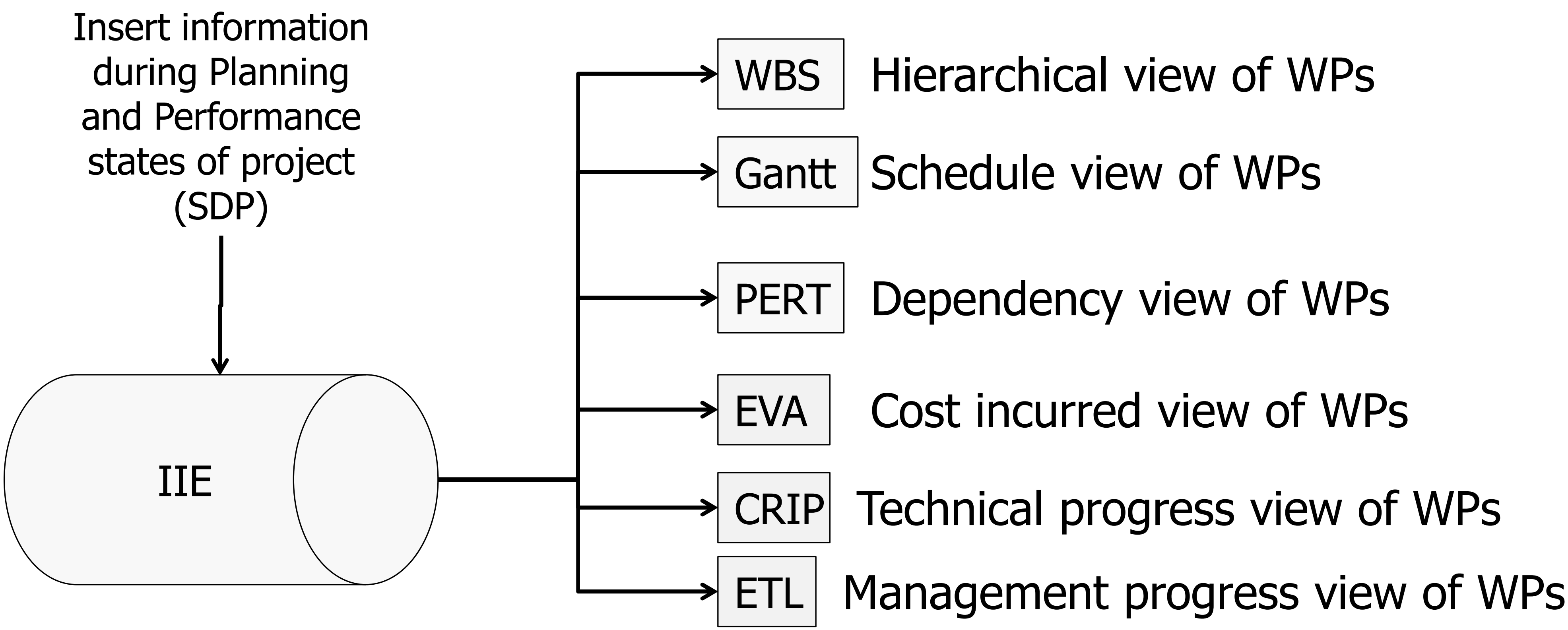
More effective process

- Tight coupling and correspondence between product objects and process objects (WP)
- Each process object (WP) produces a product object (component, subsystem)
- Integrates aspects of project management and engineering
- Integrated information environment (IIE)
  - Single integrated project database
    - Process (management)
    - Product (engineering)
  - Read only access to information provided to in-process stakeholders
  - Textual, visual (graphical and animated) and audio outputs as appropriate
- Fixes a few other defects in the current paradigm
- Scope for smart next generation integrated project management/engineering tools

Additional WP properties added over time [h]

- Used in providing management information
- Facilitates producing visual graphics in meetings
- Properties include
  - Cost of work performed
  - Actual time taken
  - Additional risks (problems) anticipated, being managed, overcome
  - Lessons learned

Views of project data

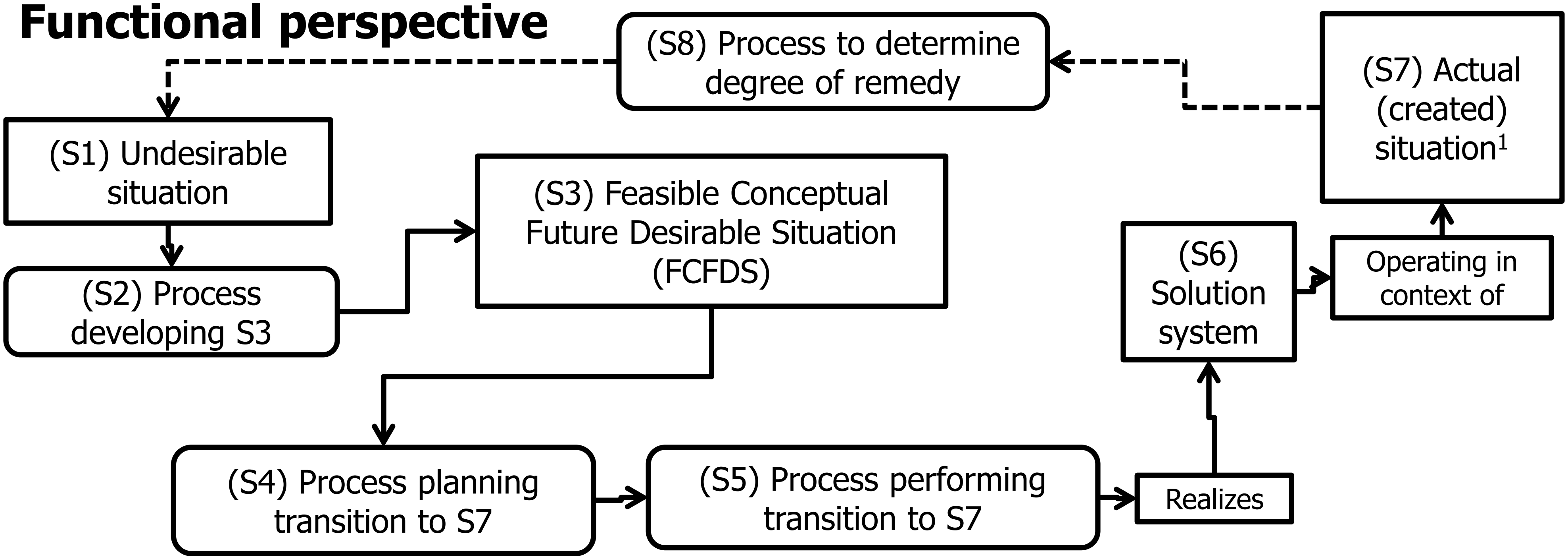


Benefits of OOSE include

- Significantly lower costs and improved probability of success
- Removes a major cause of project failure
  - Poorly written requirements
    - Nobody is being paid to write them
    - Nobody is being paid to figure out what they mean in the later states of the SDP
- Simplifies and shortens the time taken by the SDP
- Risk management is an integral part of the process, not a separate activity
- Broad scope and new markets for MBSE tool vendors
- Changes the focus of the up-front customer dialogue
  - From what do you want or need
  - What do we change in this (generic part of the model) to provide what is needed
- Can be used without consensus on underlying cause of the problem
- Maps into the extended problem-solving process and the Nine-Systems Model (Kasser, J. E. and Zhao, Y.-Y., INCOSE 2014), (Kasser, J. E., Zhao, Y.-Y. and Mirchandani, C. J., INCOSE 2014)

The Nine-System model [h]

Functional perspective



1. The solution systems and the adjacent systems are subsystems in the actual situation

Promise of OOSE

- Place for a Professional Society OOSE working group to take the lead to work with tool vendors to establish OOSE IIE Tool interface standards up-front
- Future interfaces to creation tools
  - Software compilers
  - Hardware manufacturing tools (workshop and factory)
  - 3D printers and subsequent technologies
  - Addition of "smartness" using expert systems and/or artificial intelligence for both process and product
    - Custom domain
    - Generic

(IIE) Smart tool example

House	Need	Actual
Schedule	3	2
Cost	100	20
Thermal insulation	88	100
Number of occupants	3	3
…		
…		
Wind force resistance	73	5

Designer select physical component to meet needed properties

Smart tool fills in actual numbers and shows risks

Can't do this with present day tools

This is what tool vendors should be developing

Enhanced Traffic Light Chart (single project)

#	Projects	Last time	Current		Next
			Expected	Actual	
1	Project Ho-hum				

- Adds time element to traditional traffic light chart
  - Source Earned Value Analysis (EVA) box (*Generic* perspective)
- Enhances Management by Exception (MBE)
- Enhances Management by Objectives (MBO)
  - At various levels in the project or system-subsystem hierarchy
- Can show reason for colour
  - **B**udget, **S**chedule or **T**echnical

- Note **P**roblem changed to personnel, **T**echnical, suggested by Pascal Bohulu Mabelo , COST class session 2022/09/29

Enhanced Traffic Light Chart*

#	Projects	Last time	Current		Next
			Expected	Actual	
1	Project Ho-hum				
2	Project Oh oh	T B		T S	T S
3	Project Catching up	S	S	S	
4	Project Replace manager?	B S	B S	B S	B S
5	Project Very happy customer				
6	Project Completed				N/A
7	Project Promote manager	P S	P S		
8	Project Watch this person	B S			
9	Project No risk management		T B S	T B S	T B S
10	Project Manager took course in risk management?				T
11	Project Manager doing risk management? (no effect on B and S)	T	T	T	T

https://youtu.be/fwM_9otO0F0

* Systems Thinker's Toolbox Figure 8.24, added since YouTube presentation

CRIP Chart (Category X)

Planned **E**xpected **A**ctual

	Identified			In process			Completed			In test			Accepted		
Range	**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															

‘P’ links to ‘Work to be scheduled for **next reporting** period’
‘E’ links to ‘Work Scheduled at last milestone for **this reporting** period”
‘A’ links to ‘Work Performed in **this reporting** period’

https://youtu.be/5AUafacJ5AU

Movement in chart rows [h]

Partial CRIP Chart at CDR for a specific row

In process				**Completed**		
P	**E**	**A**		**P**	**E**	**A**
20	10	10		10		

Partial CRIP Chart at TRR (subsequent milestone) for the same row

In process				**Completed**		
P	**E**	**A**		**P**	**E**	**A**
0	20	10	Catch up (10+10)	20	10	

- P→E in same CRIP State in subsequent Charts
- Depending on work done
  - E→A same CRIP State
- Depending on work planned
  - A→P in subsequent CRIP State in same Chart

CRIP Chart for category X showing requirements creep at Preliminary Design Review (PDR)

	Identified			In process			Completed			In test			Accepted		
Range	P	E	A	P	E	A	P	E	A	P	E	A	P	E	A
1	0	0	20	0	81	101	0								
2	0	0	0	0	78	78	0								
3	0	0	0	0	35	35	0								
4	0	0	0	0	30	30	0								
5	0	0	2	0	26	28	0								
6	0	0	0	0	20	20	0								
7	0	0	4	0	8	12	0								
8	**3**	0	0	0	7	7	0								
9	0	0	0	0	5	5	0								
10	0	0	0	0	2	2	0								
Totals	3	0	26	0	292	318	0								

Impact of additional work should show on EVA cost and Gantt schedule charts

https://youtu.be/5AUafacJ5AU

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

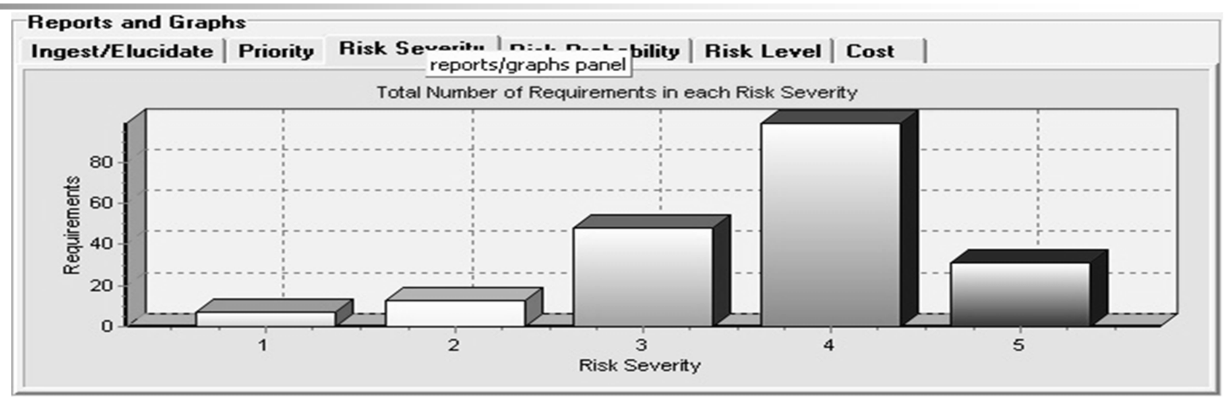
Severity of consequences (S)
(Impact)

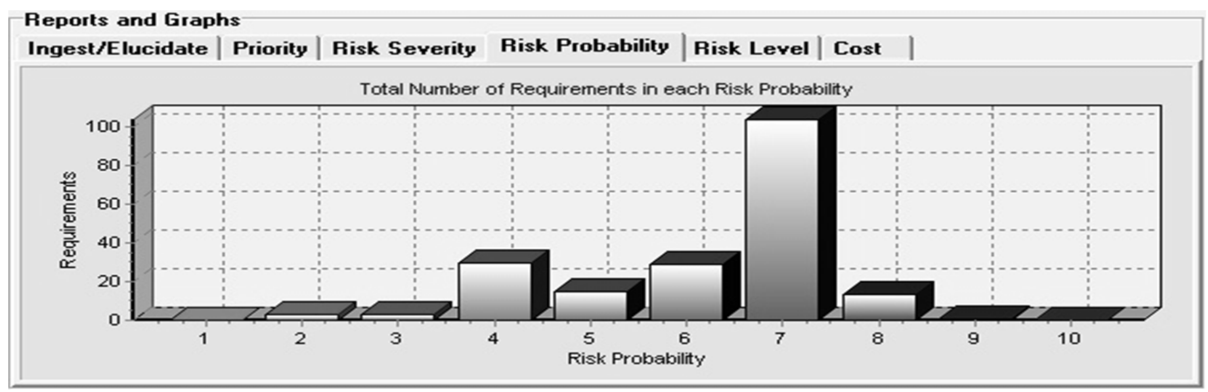
Based on one number (L*S)

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

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

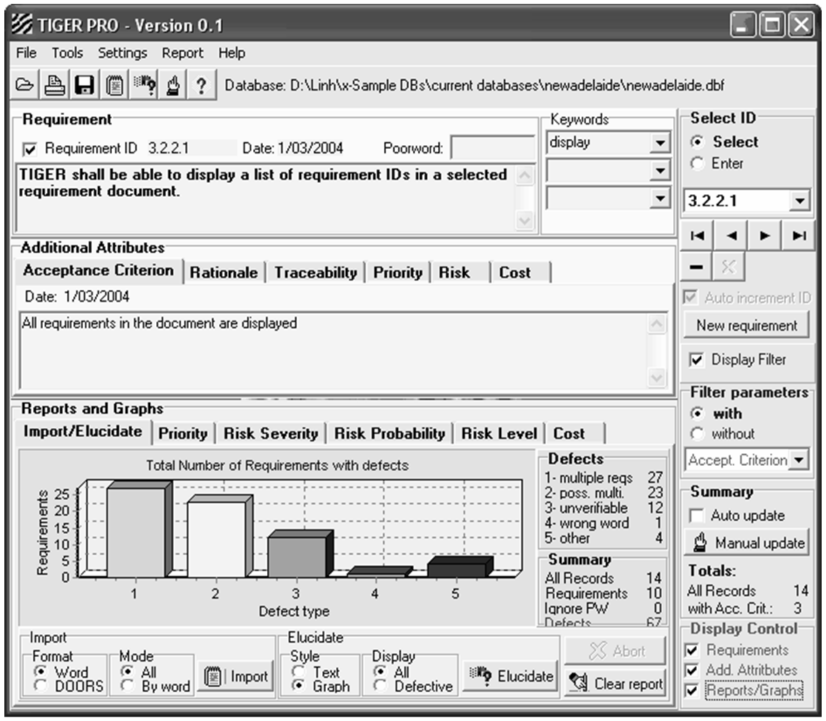
Project Risk profiles – a better way?





Risk: Are these reasonable?

Tiger Pro [h]



https://youtu.be/-Uegko4WQO8

- The First Requirements Elucidator Demonstration (FRED) Tool, 2004
  - Out-of-the-box solution from a word processor spelling chequer
- Detected many poor word in requirement statements
- **T**ool to **I**n**G**est and **E**lucidate **R**equirements
- Produces a Figure of Merit for a set of requirements
- Contains additional attributes of requirements
  - Acceptance criteria, priority, risk, etc.
- Free for educational and personal use
- Workplace single user license is $50

Computer Enhanced Systems Engineering

- Not MBSE, but CESE in application language*
  - i.e. Operations Concept Harbinger (OCH)*
- Integrated Information Environment (IIE)
- Vendor flexible templates for classes of projects
  - Product templates and process templates
- Requires interface Standards for AI/Expert System (ES) plug-ins for various domains
- Smart questions at both Architecture Reviews
  - e.g. based on comparing properties – i.e. (low) priority vs. (high) estimated cost
- AI in templates can do
  - a reality check on estimates – stops low bidding
  - a feasibility check on proposed process and product implementation
- AI and ES can overcome some loss of tacit knowledge as experts retire
- Risk profiles can compare designs or architectures

* Kasser J., Applying Total Quality Management to Systems Engineering, Artech House, 1995

Postscript

SE is all about ensuring that all the properties of the system as delivered (system capability) are at least equal to the properties of the system needed.

Thus it is requirements free.*

* Kasser, J.E., Does OOSE eliminate requirements, 2002

Do you have what it takes?

- Have you recognized the defects in the requirements paradigm?
- Are you interested in the OOSE paradigm
- Find out if you can make the transition
- Join me and a small select group for at least 30 days **FREE**
- Full access to two programs
- https://therightrequirement.com/

Questions and comments?

- Interested but don’t want to try the programs at this time
- Talk to me and others in the online Oasis Café using Zoom
  - Meeting ID: 873 4055 1993
  - Passcode: 557756
  - Daily; Monday to Thursday at 7 pm New York local time
- I’m also online for students and others
  - Mondays at 8 am Zurich local time

- My thanks to Raid AlQaisi, Martin Hugi, Niels Malotaux, Bruce Lerner, Gregory Olson, Shirley Tseng and Adrienna Zsakay who made useful comments in the online INCOSE Fir Tree and Oasis cafés and in one of my classes during the preparation of this presentation