34th Annual INCOSE international symposium

hybrid event

Dublin, Ireland
July 2 - 6, 2024

Requirements—Why Bother?

Dr Mike Ryan, Co-Chair, INCOSE Requirements Working Group (RWG)

Requirements—Why Bother?

- System developments fail in embarrassing numbers by embarrassing amounts of time and cost. Something must be done.
- We are interested here in requirements, but the role of requirements in systems development is perhaps one of the most contentious issues in modern systems engineering. There are two broad views:
  - Some believe that project failure is largely due to an unnecessary focus on formal requirements, particularly text-based requirements, which are no longer necessary as part of modern development methodologies.
  - Yet, others point out that it is actually the lack of appropriate requirements that is one of the principal causes of project failure.
- So, who is correct?

Requirements—Why Bother?

- It is an important issue. Why would a project team bother to expend all that effort and angst in developing formal requirements if they aren't necessary?
- Let's look at:
  - The evidence.
  - The two broad schools of thought as to the role of requirements in avoiding project failure—that is, we will address the question: *why bother with requirements?*

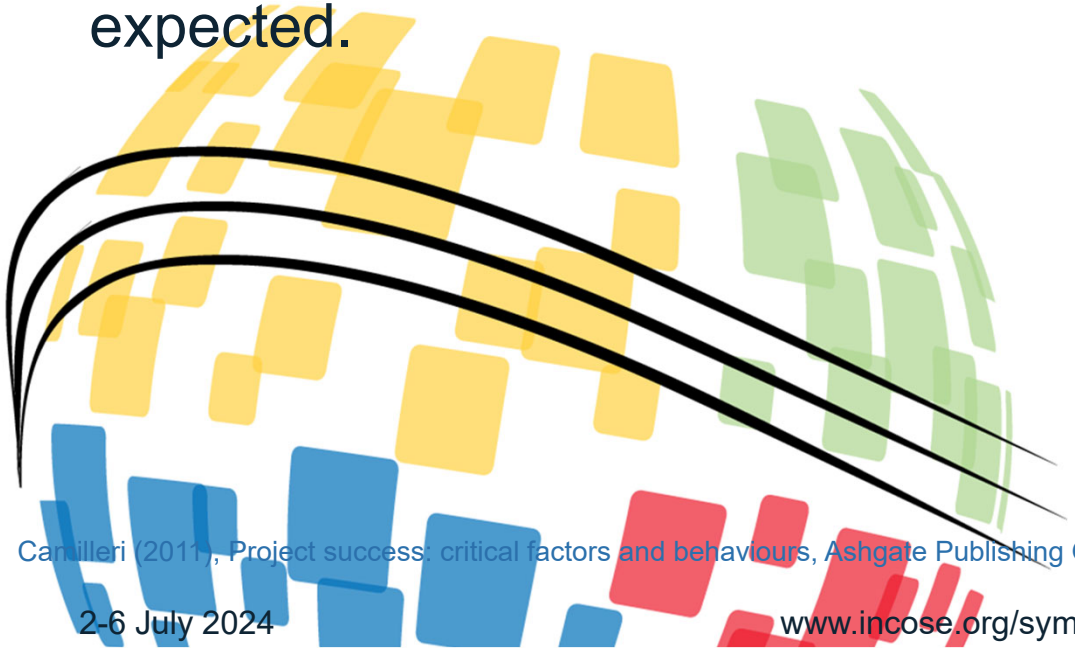
The Evidence

- The US DoD loses approximately $150 million / day due to delayed and cancelled production.
- Current US DoD system developments are overrunning 78% in cost and 63% in schedule.
- Boeing 787 and Airbus 380 are both two years late with the with a cost overrun of $10 billion and €2 billion respectively.
- Pentagon's 80 largest weapons systems were collectively $480 billion over their original cost estimates, while the average weapons program was over two years delayed in delivering initial operating capability.
- Chevy Volt has doubled in manufacturing cost during design.
- NASA's projects have an average cost growth of between 47-61%

The Evidence

- Research indicates that up to 85 to 90 per cent of projects fail to deliver:
  - on time,
  - on budget, and
  - to the quality

  expected.



Camilleri (2011), Project success: critical factors and behaviours, Ashgate Publishing Group, 2011, p. 10.

Requirements—Why Bother?

- So, what to do?
- Well, based on the same evidence, the two views broadly are:
  - Requirements—Why (would you) Bother?
    - You don't need requirements.
      - There is no point.
  - Requirements—Why (should you) Bother?
    - You do need requirements.
      - You have missed the point.

Why (would you) Bother?

Requirements—Why (would you) Bother?

- Application domain information is not always collected in one place and may involve specialist terminology.
- People who understand the problem to be solved are often too busy to help develop a solution.
- Organisational issues and political factors may influence system requirements to satisfy personal agenda.
- Stakeholders often do not help—they often don't really know what they want, they have no desire to compromise their unrealistic demands when faced with cost, and they have different requirements that are often expressed in different ways.
- Requirements development takes place in economic and business environments that are continually changing.

Requirements—Why (would you) Bother?

- Different people have different perspectives—as key positions change, the new incumbent may have a significantly different view of requirements.
- Not every stakeholder is a supporter of the new system.
- Stakeholders do not understand the requirements-development process and often do not participate.
- Stakeholders will often not commit to a written set of requirements for fear that it may constrain them.
- Stakeholders may insist on adding/modifying requirements after the set has been endorsed.
- Stakeholders may not be willing (or able) to understand the technologies and processes involved.

Requirements—Why (would you) Bother?

When you assemble a number of [people] to have advantage of their joint wisdom, you inevitably assemble with those [people] all their prejudices, their passions, their errors of opinion, their local interests and their selfish views. From such an assembly can a perfect production be expected?

Benjamin Franklin

Over time, every way of thinking generates important problems that it cannot solve.

Russell Ackoff

Without changing our patterns of thought, we will not be able to solve the problems we created with our current patterns of thought.

Albert Einstein

The difficulty lies, not in the new ideas, but in escaping from the past.

John Keynes

Requirements—Why (would you) Bother?

- *The critical failing of user interviews is that you're asking people to either remember past use or speculate on future use of a system.* Jakob Nielsen
- *Customers don't know what they want. It's very hard to envision the solution you want without actually seeing it.* Marty Cagan
- *Customers don't know what is possible. Most have no idea about the enabling technologies involved.* Marty Cagan
- *You can't just ask customers what they want and then try to give it to them. By the time you get it built, they'll want something new*. Steve Jobs
- *If I had asked my customers what they wanted, they'd have said a faster horse.* Henry T. Ford

Requirements—Why (would you) Bother?

- In large software developments—of the features required by stakeholders (Jan Bosch, INCOSE IS 2015):
  - *7% always used,*
  - *13% often used,*
  - *16% sometimes used,*
  - *19% rarely used, and*
  - *45% never used.*
- That is, only 20% of features are always or often used.
- Why would you bother wasting time finding out about the other 80% when all that development work (which then overruns) is going to be wasted?

Why (should you) Bother?

Remember this evidence:

- In large software developments—of the features required by stakeholders (Jan Bosch, INCOSE IS 2015):
  - *7% always used,*
  - *13% often used,*
  - *16% sometimes used,*
  - *19% rarely used, and*
  - *45% never used.*
- Only 20% of features are always or often used.
- Perhaps we should have collected the requirements properly and not have wasted time eliciting requirements that are not necessary (and, at the same time, probably missed many that were)?

Remember this evidence

- Research indicates that up to 85 to 90 per cent of projects fail to deliver:
  - on time,
  - on budget, and
  - to the quality

expected.

So, project failure relates not to what we do, but to what we **plan** to do—that is, one could argue, we fail because we don't start with the appropriate set of requirements.

Remember, 80% of the requirements we collected were unwanted and we would also have missed many that were actually necessary.

Camilleri (2011), Project success: critical factors and behaviours, Ashgate Publishing Group, 2011, p. 10.

Remember these observations:

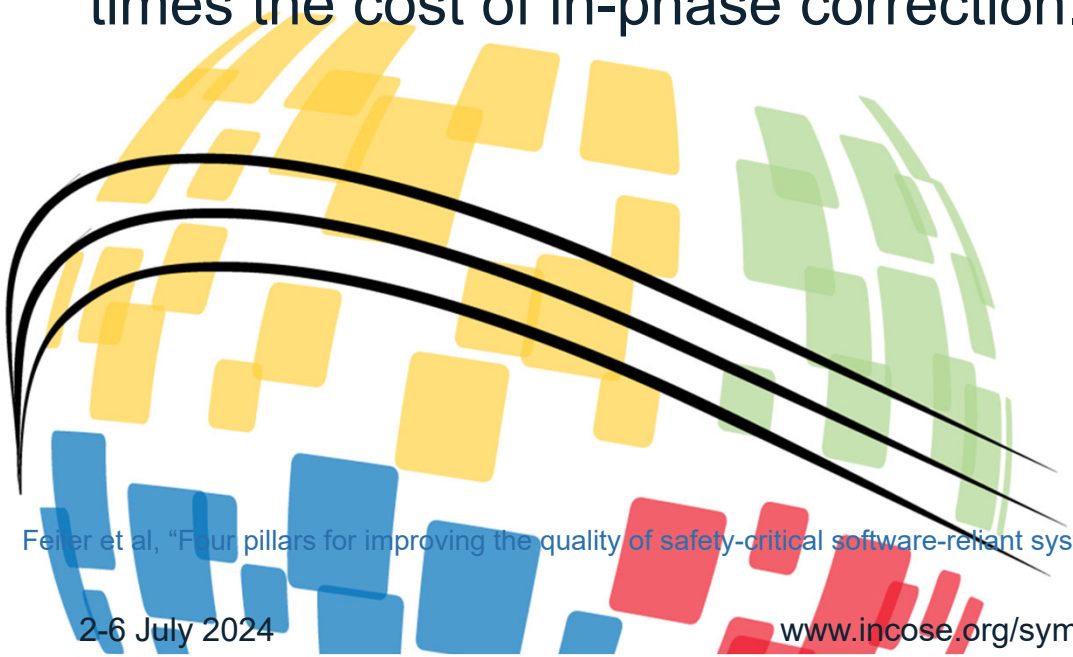
- *The critical failing of user interviews is that you're asking people to either remember past use or speculate on future use of a system.* Jakob Nielsen
- *Customers don't know what they want. It's very hard to envision the solution you want without actually seeing it.* Marty Cagan
- *Customers don't know what is possible. Most have no idea about the enabling technologies involved.* Marty Cagan
- *You can't just ask customers what they want and then try to give it to them. By the time you get it built, they'll want something new*. Steve Jobs
- *If I had asked my customers what they wanted, they'd have said a faster horse.* Henry T. Ford

Perhaps we should be more careful when eliciting requirements and not expect to just write down and implement anything a stakeholder says.

(Remember only 20% of requirements elicited from stakeholders are useful, and what about what they don't say—missing requirements?)

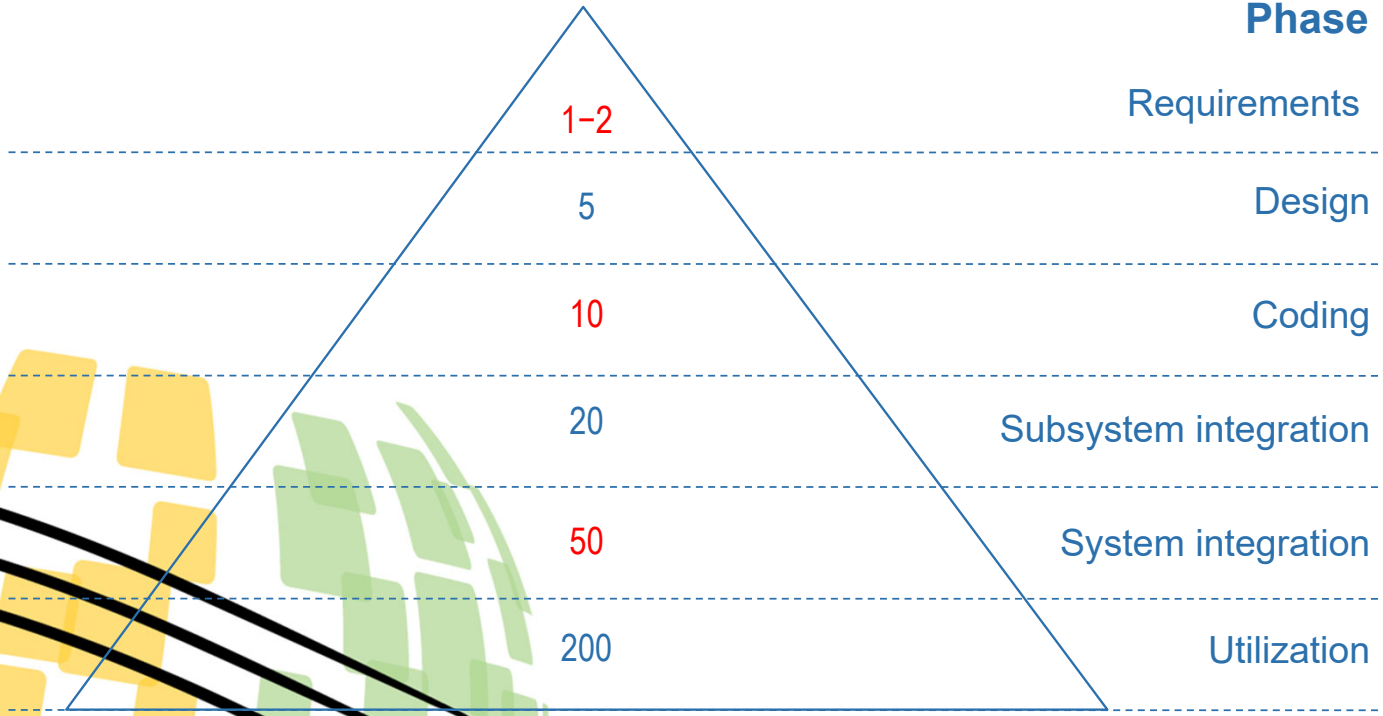
Requirements—Why (should you) Bother?

- Some 70% of software faults are introduced in early design: 35% during requirements development, and 35% during architectural design.
- Some 80% of those faults are discovered at systems integration or later.
- Rework effort to correct a fault in later phases can be as high as 300-1,000 times the cost of in-phase correction.



Feiler et al, “Four pillars for improving the quality of safety-critical software-reliant systems”, SEI, 2013.

Requirements—Why (should you) Bother?

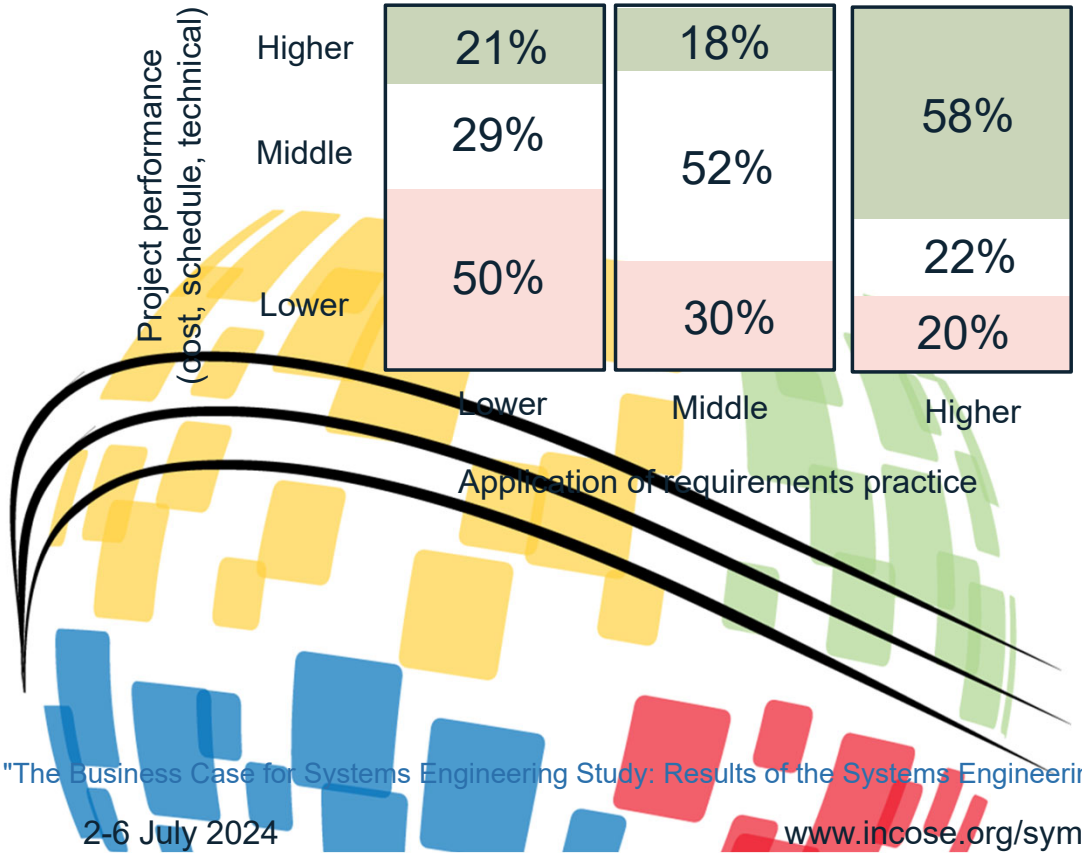


Boehm, B., *Software Engineering Economics,* Englewood Cliffs, NJ: Prentice Hall, 1981.
Davis, A.M., *Software Requirements: Objects, Functions and States,* Englewood Cliffs, NJ: Prentice Hall, 1993.

Keys to Project Success

- The *Chaos Summary* identifies annually the top ten project success factors:
  - user involvement,
  - executive management support and buy in,
  - clear statement of requirements,
  - proper planning,
  - realistic expectations,
  - smaller project milestones,
  - competent staff,
  - ownership,
  - clear vision and objectives, and
  - hard-working focused staff.

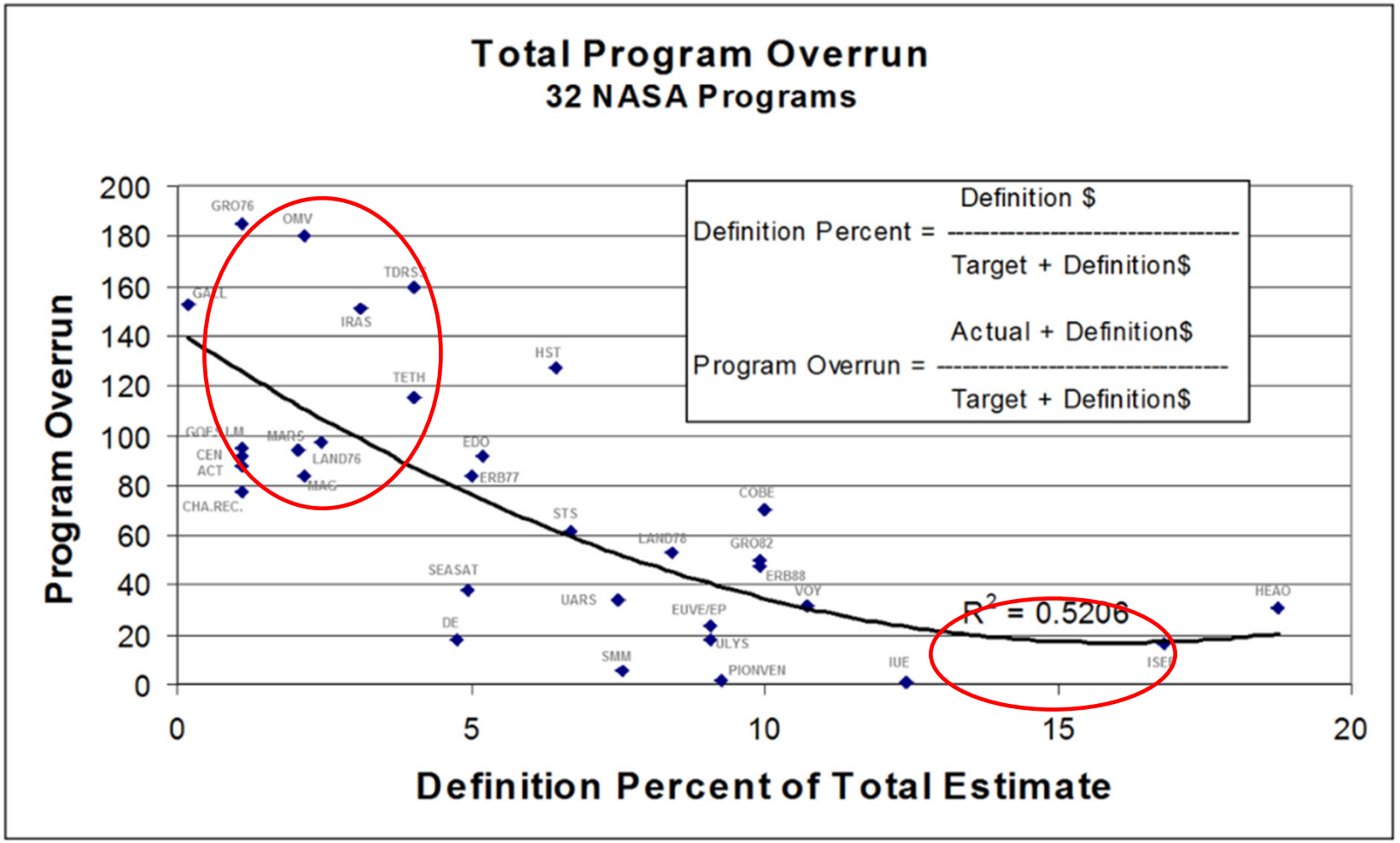
The Value of Good Requirements



The number of projects with higher performance almost triples with better requirements practice.

"The Business Case for Systems Engineering Study: Results of the Systems Engineering Effectiveness Survey," D. Goldenson and J. Elm, Software Engineering Institute, 2012.

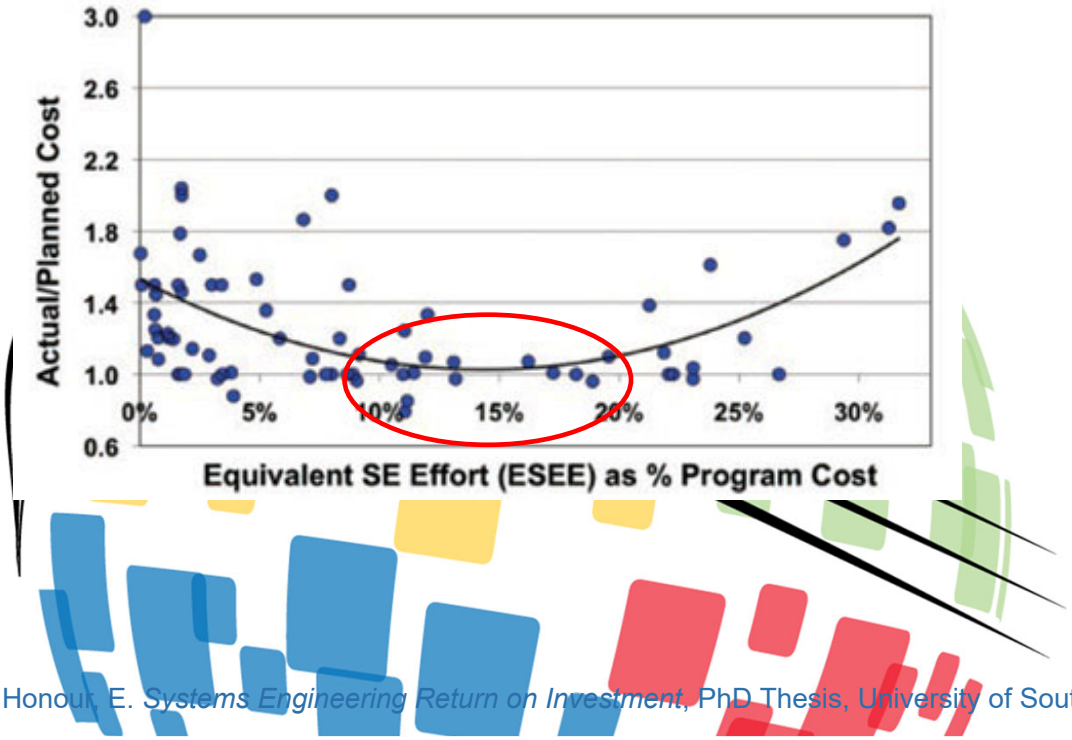
The Importance of project definition

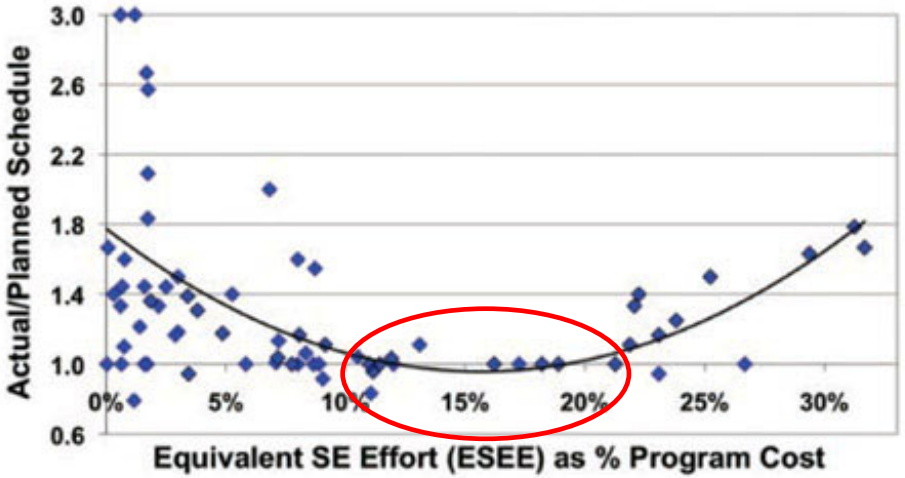


Source INCOSE (Werner Gruhl NASA Comptroller's Office)

SE Return on Investment

- Taking a little licence and using SE effort as an analogue for requirements development effort:





Honour, E. *Systems Engineering Return on Investment*, PhD Thesis, University of South Australia, 2013

So, what's the point?

We do need requirements because:

- We do need to be able to define a scope for the project / phase / activity / increment / evolution / spiral / sprint. How else would we know what we are collectively to do in that time period?
- We do need to ensure that everyone involved has had input and those various points of view have been (and continue to be) analysed and reconciled.
- We do need to be able to guarantee that we have set the work up for success with a complete set of requirements that are unambiguous, complete, non-conflicting, and so on.
- We do need to trade off functionality for cost—which implies that we understand the required functionality, as well as associated priorities, costs, and so on.
- We do need to be able to manage changes in requirements for a wide variety of reasons.

We need requirements management because:

- We do need to be able to justify any expenditure of funds or effort based on the contractual/ethical/moral obligation to achieve an endorsed set of requirements that return value to the organisation.
- We do need to be able to contract for the completion of a set of requirements, particularly if we are using a firm-fixed pricing structure, regardless of whether that is for a phase, an increment, an iteration, a build, a spiral, or a sprint.
- We do need to be able to report on progress in order to meet our obligations (and to justify when payment should be made).
- We do need to be able to tell when we are finished in order for our obligations to have been met (on both sides of a contract).

We need business leadership because:

- Remember the *Chaos Summary's* top ten project success factors:
  - user involvement,
  - executive management support and buy in,
  - clear statement of requirements,
  - proper planning,
  - realistic expectations,
  - smaller project milestones,
  - competent staff,
  - ownership,
  - clear vision and objectives, and
  - hard-working focused staff.

We need business leadership because:

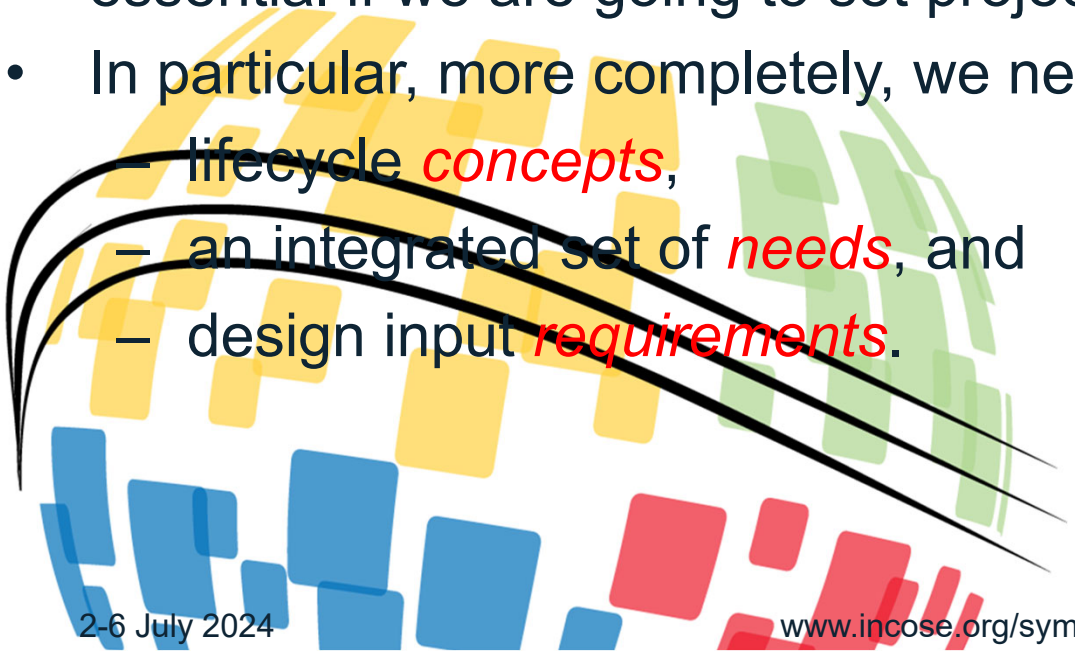
- Despite the ‘engineering’ aspect of systems engineering, requirements engineering is fundamentally a political process.
- Many complex negotiations are required involving many stakeholders and actors with many, often competing, perspectives (many of which should be discounted).
- Conflict is therefore inherent in both the requirements themselves and in the process to elicit and elaborate them.
- This fundamentally social aspect of a project must be addressed by the business—that is, by business management.
- It is also vitally important that the process is ‘led’—that is, there is a chief stakeholder who is most likely also the project champion.

So, what is the point?

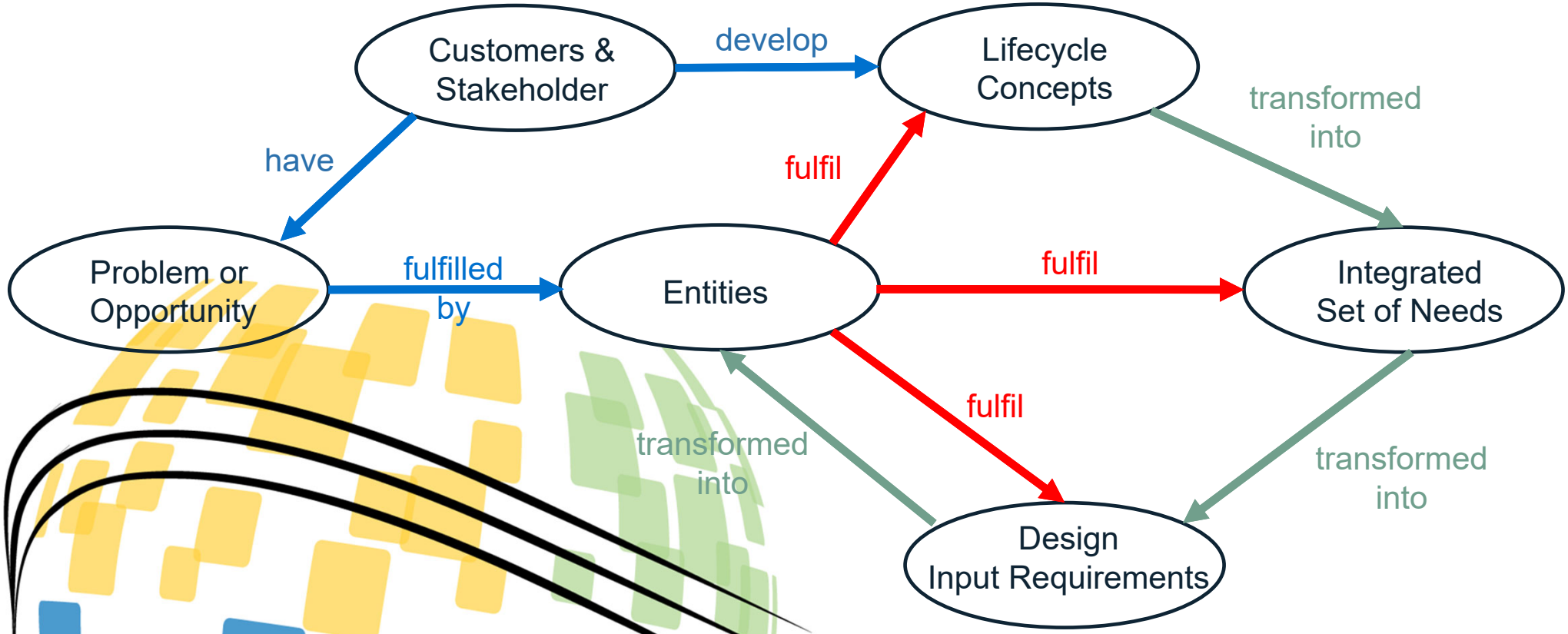
- We should bother with requirements.
- We do need to be more agile, but we need to become so by adapting our development methodology (linear sequential, incremental, iterative, evolutionary, spiral, or Agile), not our view of whether we should bother with requirements.
- Whichever development methodology we select, we still need a complete set of unambiguous non-conflicting requirements at the beginning of each phase, increment, iteration, spiral, build, or sprint (for all the reasons we just discussed).

Something a little more urbane

- Now, in taking advantage of your patience and using a lot of exaggeration for emphasis, I have been using the vernacular in that I have only made use of a single term: *requirements*.
- I am obliged now to be more careful, since a more sophisticated ontology is essential if we are going to set projects up for success.
- In particular, more completely, we need to consider:
  - lifecycle *concepts*,
  - an integrated set of *needs*, and
  - design input *requirements*.



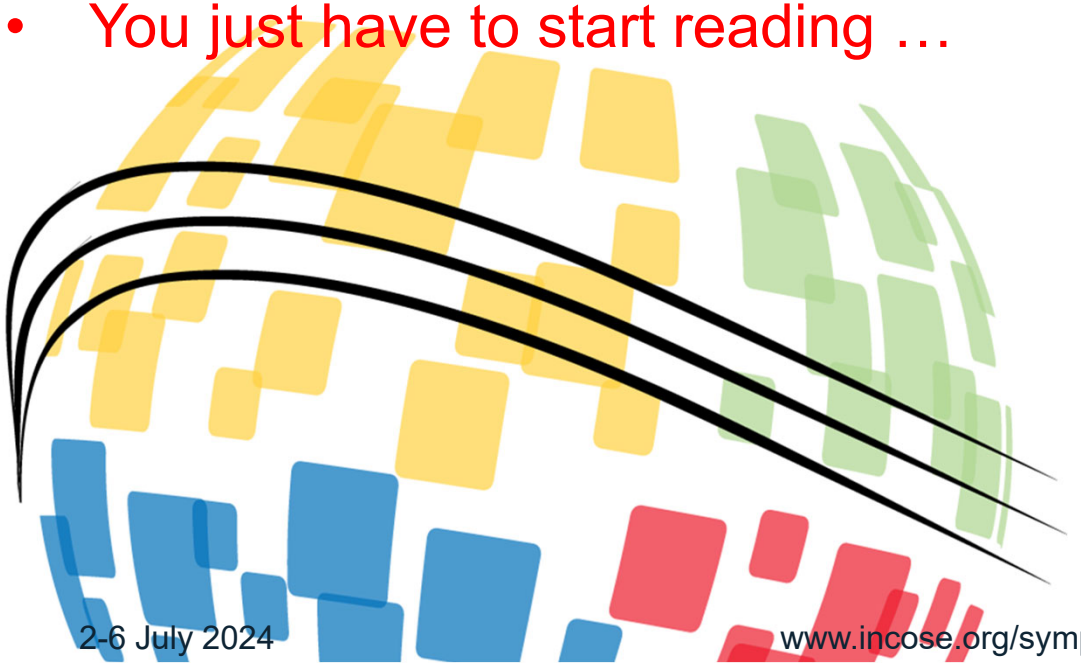
Concepts, Needs and Requirements



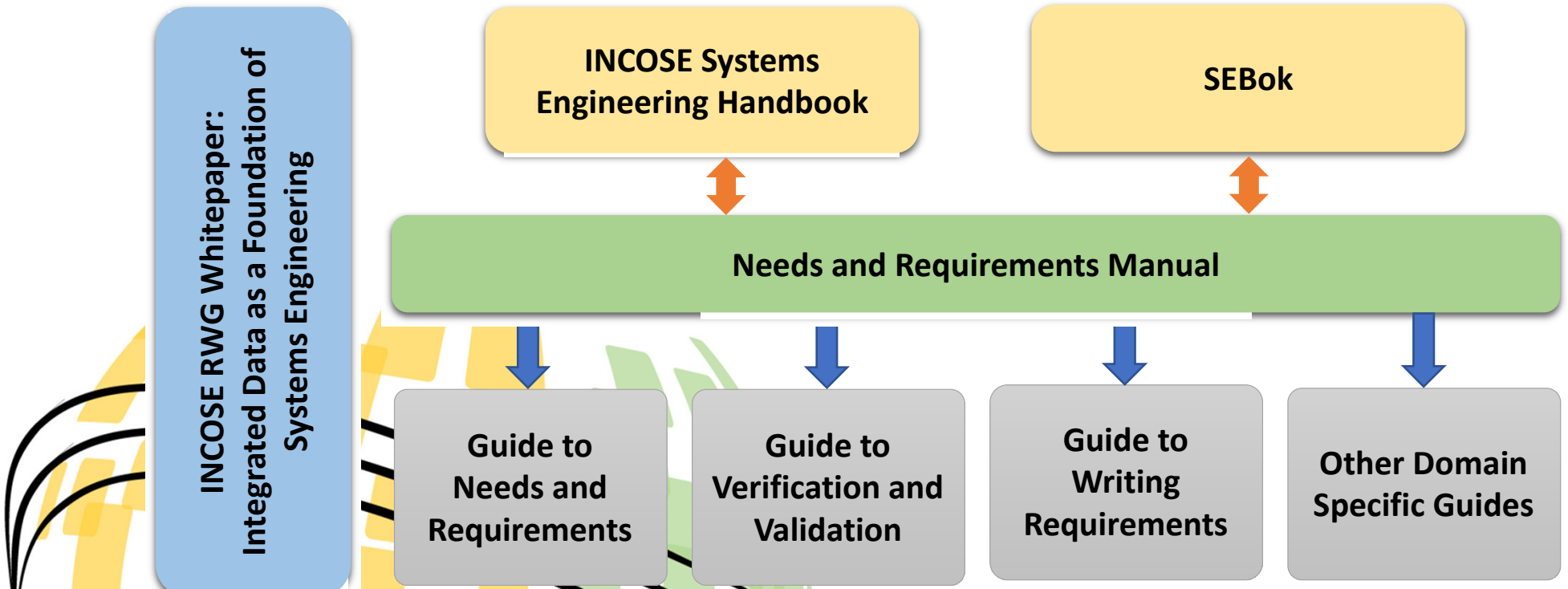
INCOSE Needs and Requirements Manual, Wheatcraft, Ryan and Katz.
Original figure created by M. Ryan an L. Wheatcraft. Usage granted per the INCOSE Copyright Restrictions.

So, finally:

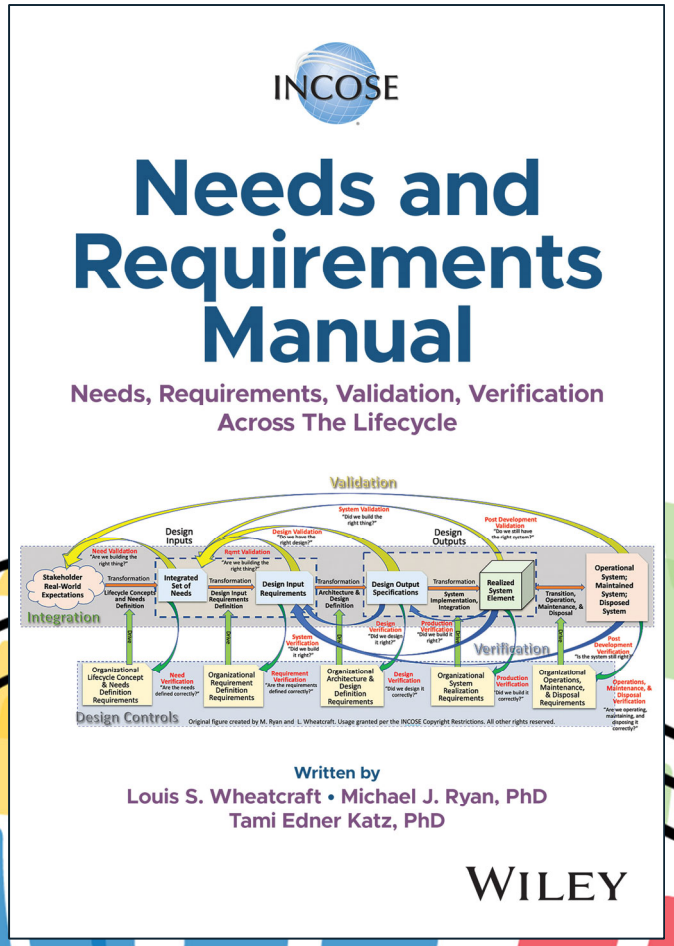
- Just because something is hard to do, doesn't mean it should be avoided.
- Just because it didn't seem to work in the past, doesn't mean it isn't worth doing—it just means that we weren't doing it properly in the past.
- We do know how to do it properly, so there is no excuse.
- You just have to start reading …



RWG Products

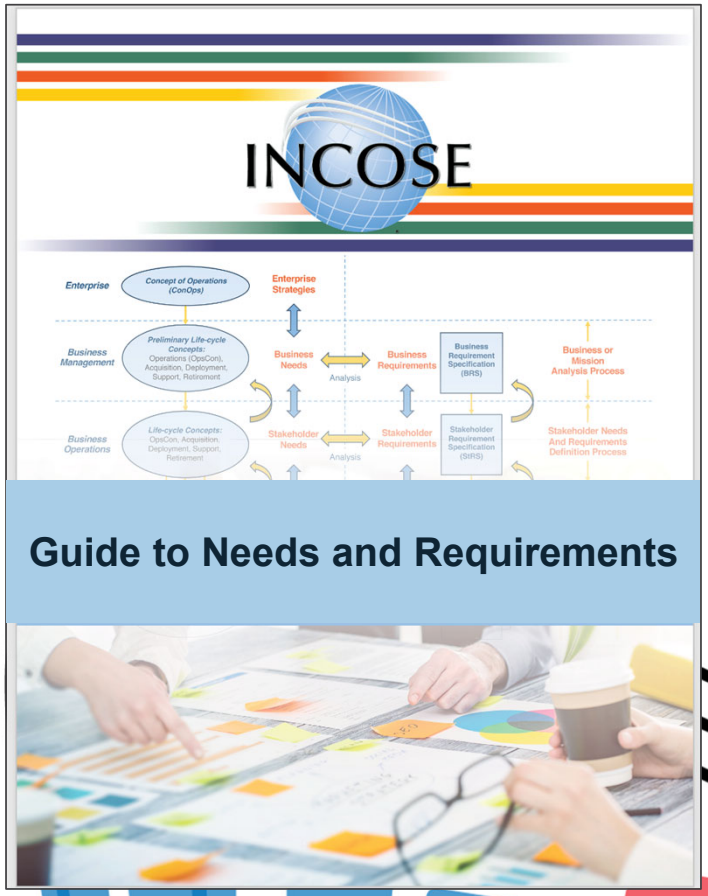


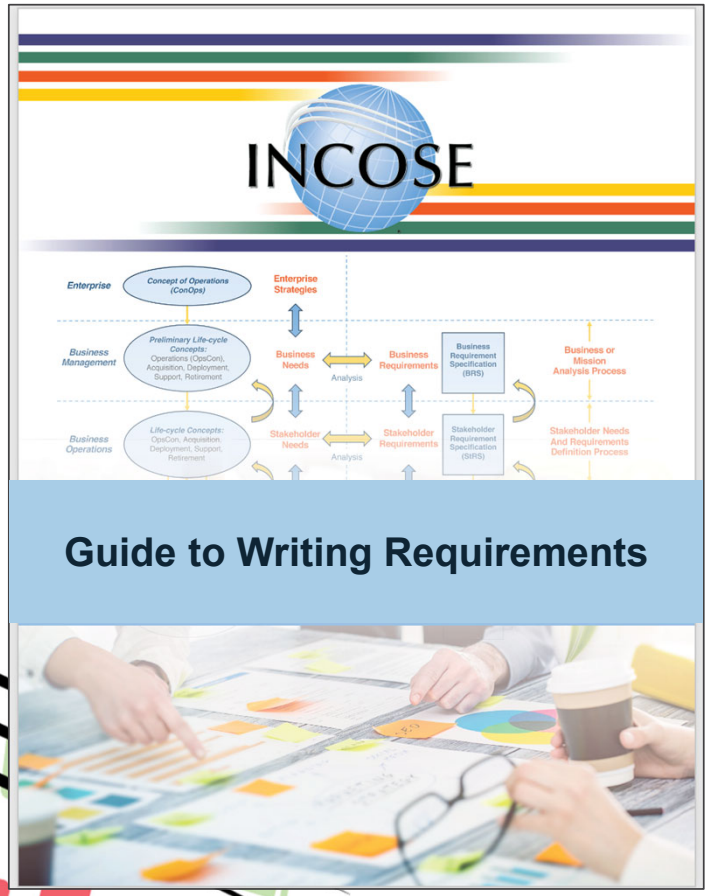
RWG Products

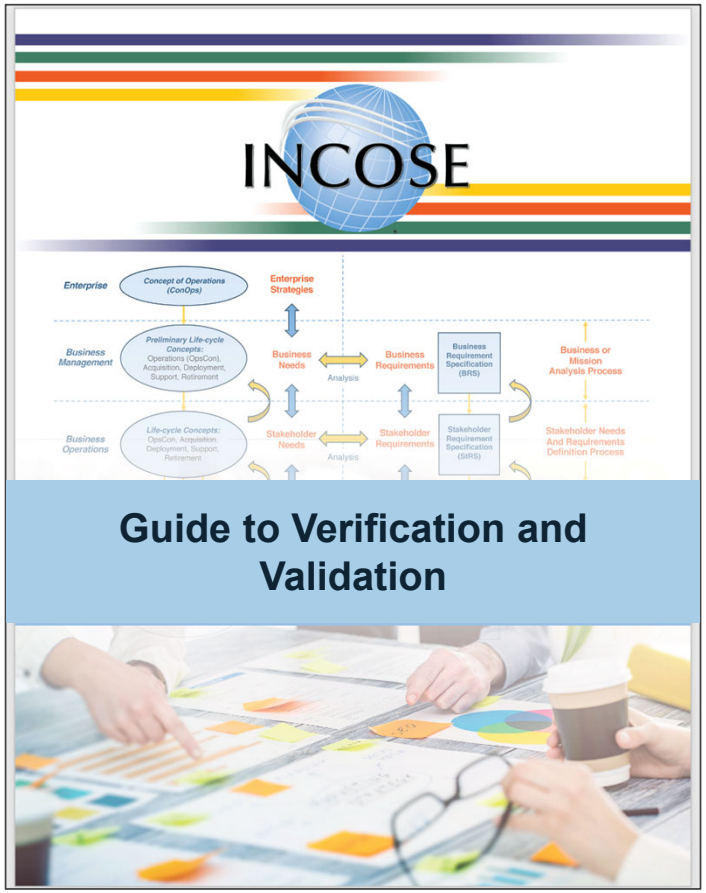


- The Needs and Requirements Manual (NRM) is the RWG flagship product.
- Content aligns with, and expands, the INCOSE SE Handbook version 5.
- To be published as a Wiley text by end 2024.

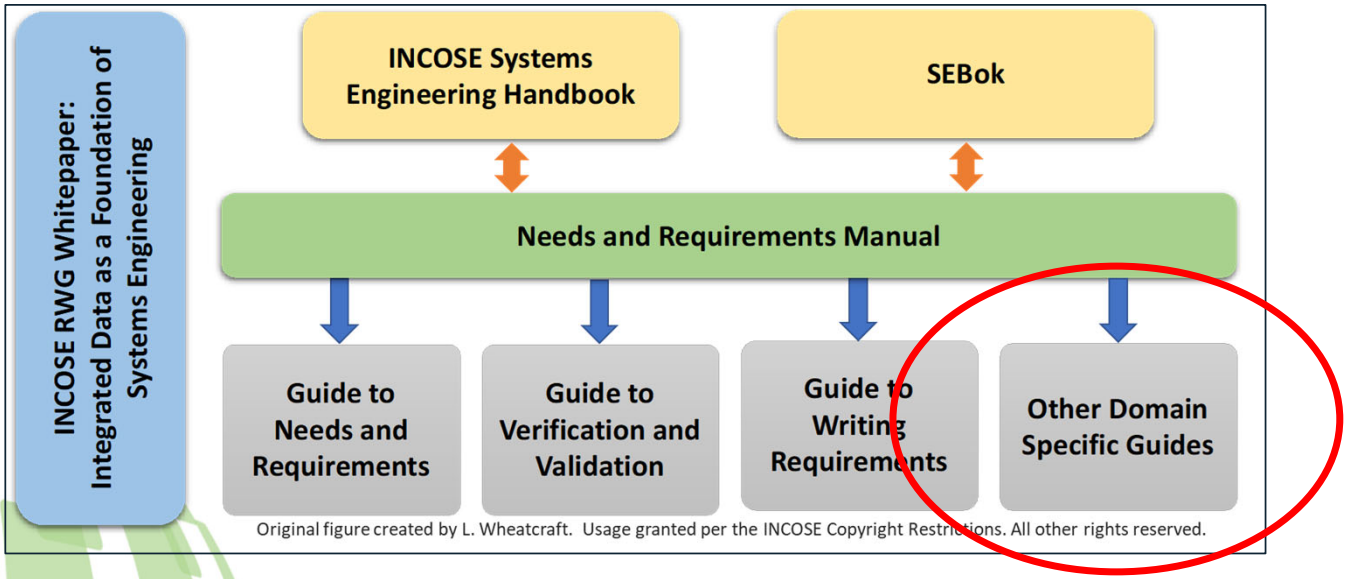
RWG Products are in the INCOSE Store





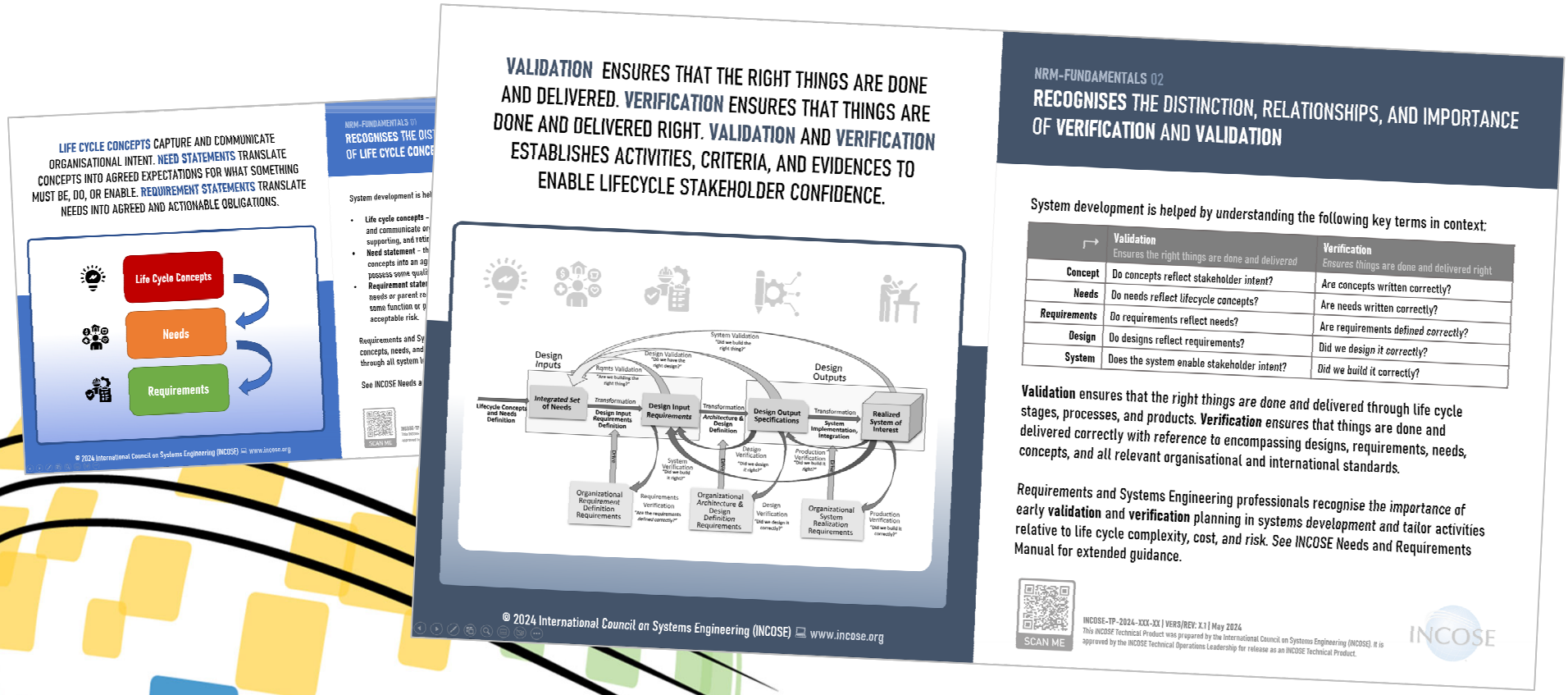


RWG Products



- Guide to Model-Based Needs and Requirements
- Guide to System Safety Needs and Requirements
- Guide to System Security Needs and Requirements
- …

NRM Fundamentals – Flash Card Illustrations



An exciting new RWG activity led by Rob Black and his team.

NRM Fundamentals – Draft List

01 - Recognises the distinction, relationships, and importance of life cycle concepts, needs and requirements
02 - Recognises the distinction, relationships, and importance of verification and validation
03 - Recognises the importance of, and key-enablers to, an integrated, collaborative project team
04 - Recognises the importance and key-enablers to effective communications
05 - Recognise the distinction of, and paired approaches required for, green field and brown field systems
06 - Recognises the importance of, and key-enablers for, avoiding technical debt
07 - Recognises the value of, and key enablers for, implementing an information-based needs and requirements definition and management (I-NRDM) approach
08 - Recognises the dynamic context of concepts, needs, and requirements management
09 - Recognises the importance of, and key enablers for, identifying, engaging, and managing life cycle stakeholders
10 - Recognises the importance of defining and managing systems context and associated interfaces
11 - Recognises the importance of defining and managing risk
12 - Recognises the value and application of functional analysis
13 - Recognises the importance and key-enablers for effective Needs, Requirements, Verification, and Validation Management (NRVVM) planning

34th Annual INCOSE international symposium

hybrid event

Dublin, Ireland
July 2 - 6, 2024

Requirements—Why Bother?

Dr Mike Ryan, Co-Chair, INCOSE Requirements Working Group (RWG)
Michael.Ryan@incose.net

34th Annual INCOSE international symposium

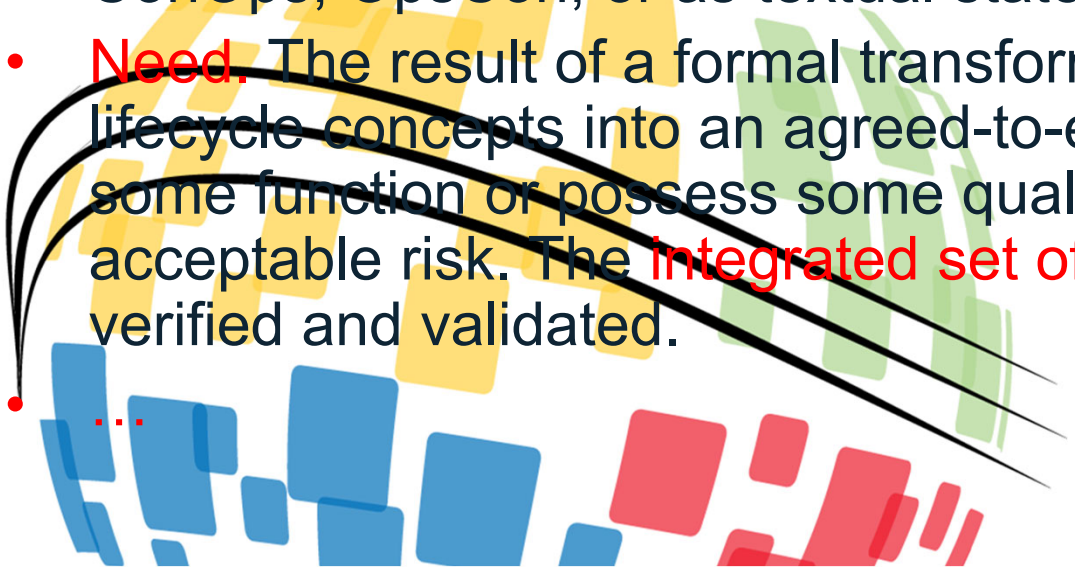
hybrid event

Dublin, Ireland
July 2 - 6, 2024

www.incose.org/symp2024
#INCOSEIS

Concepts, Needs and Requirements

- Concept. A textual or graphic representation that concisely expresses how an entity can fulfil the problem, threat, or opportunity, mission, goals, and objectives it was defined to address. The concept demonstrates how the entity provides a business capability in terms of people, processes, and products within constraints and acceptable risk. Concepts can be in the form of user stories, use cases, user scenarios, operational scenarios, ConOps, OpsCon, or as textual statements
- Need. The result of a formal transformation of one or more sources or lifecycle concepts into an agreed-to-expectation for an entity to perform some function or possess some quality within specified constraints with acceptable risk. The integrated set of needs is structured and can be verified and validated.
- …



Concepts, Needs and Requirements

- …
- Requirement. A requirement statement is the result of a formal transformation of one or more sources, needs, or higher-level requirements into an agreed-to-obligation for an entity to perform some function or possess some quality within specified constraints with acceptable risk. Requirements are transformed from needs through a process of requirements analysis (also called business analysis or mission analysis). Requirement statements can be verified and validated.

Concepts, Needs and Requirements