33rd Annual INCOSE international symposium

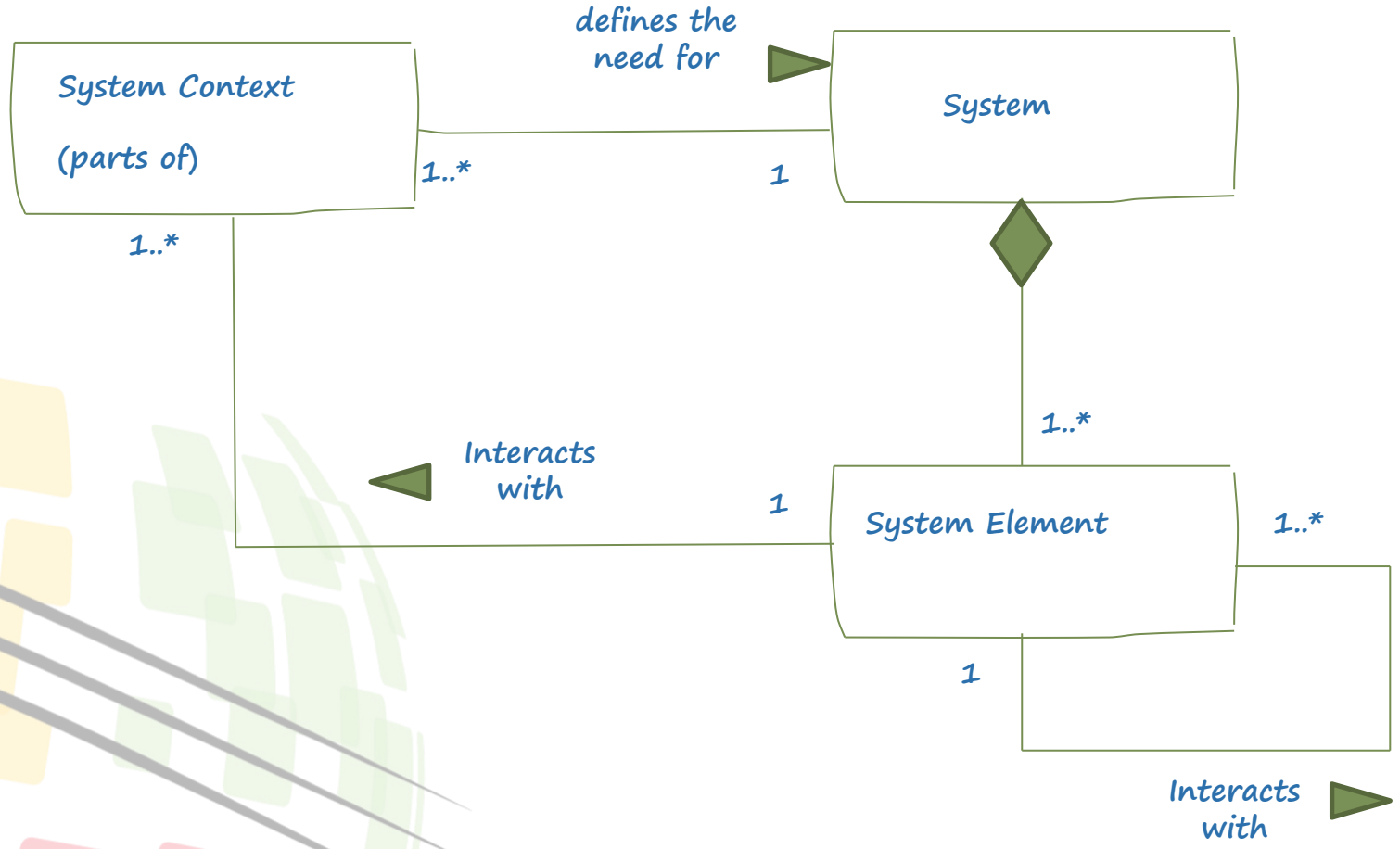
hybrid event

Honolulu, HI, USA
July 15 - 20, 2023

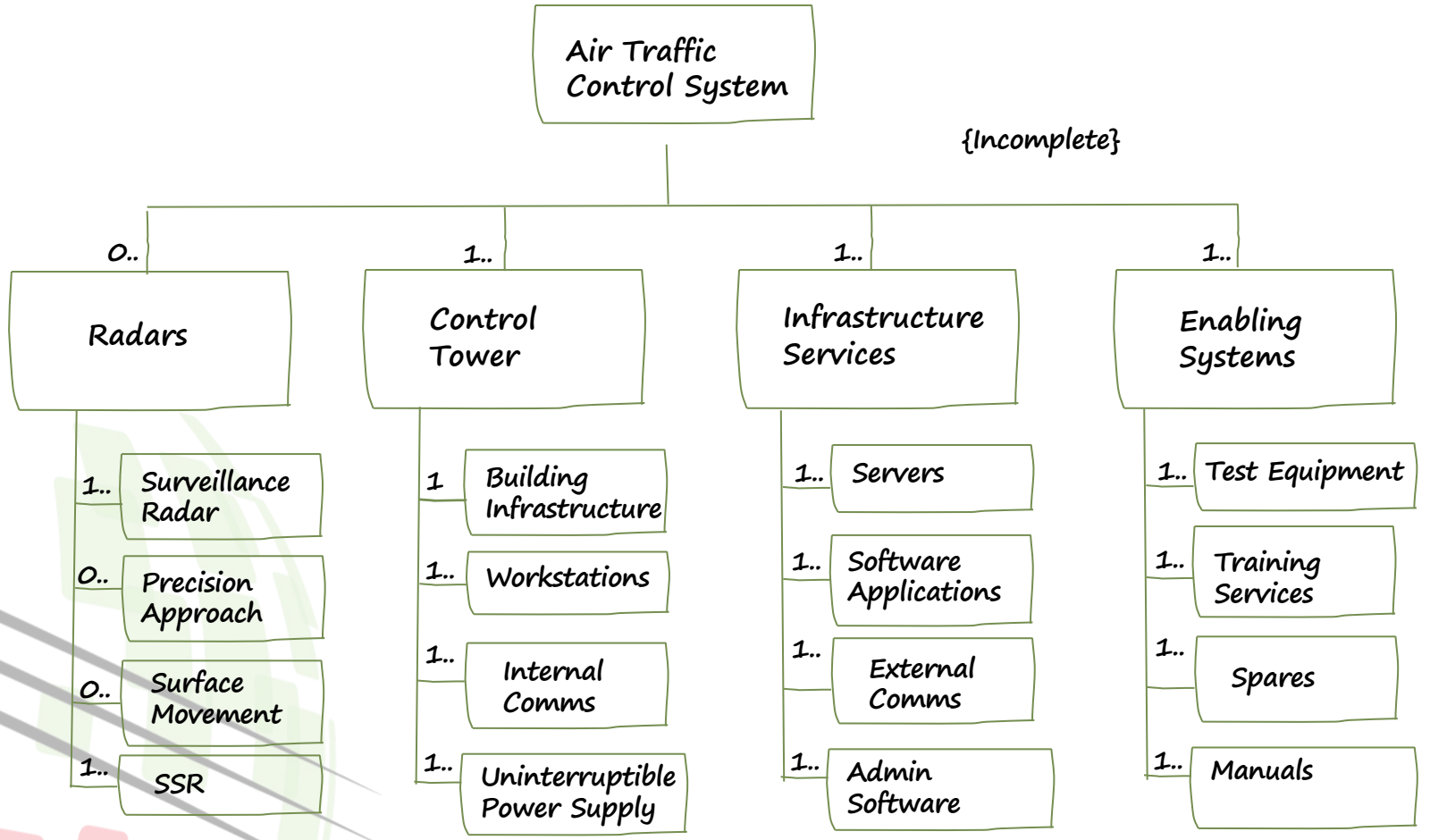
Paul Davies, thesystemsengineer.uk

Get yourself tested!

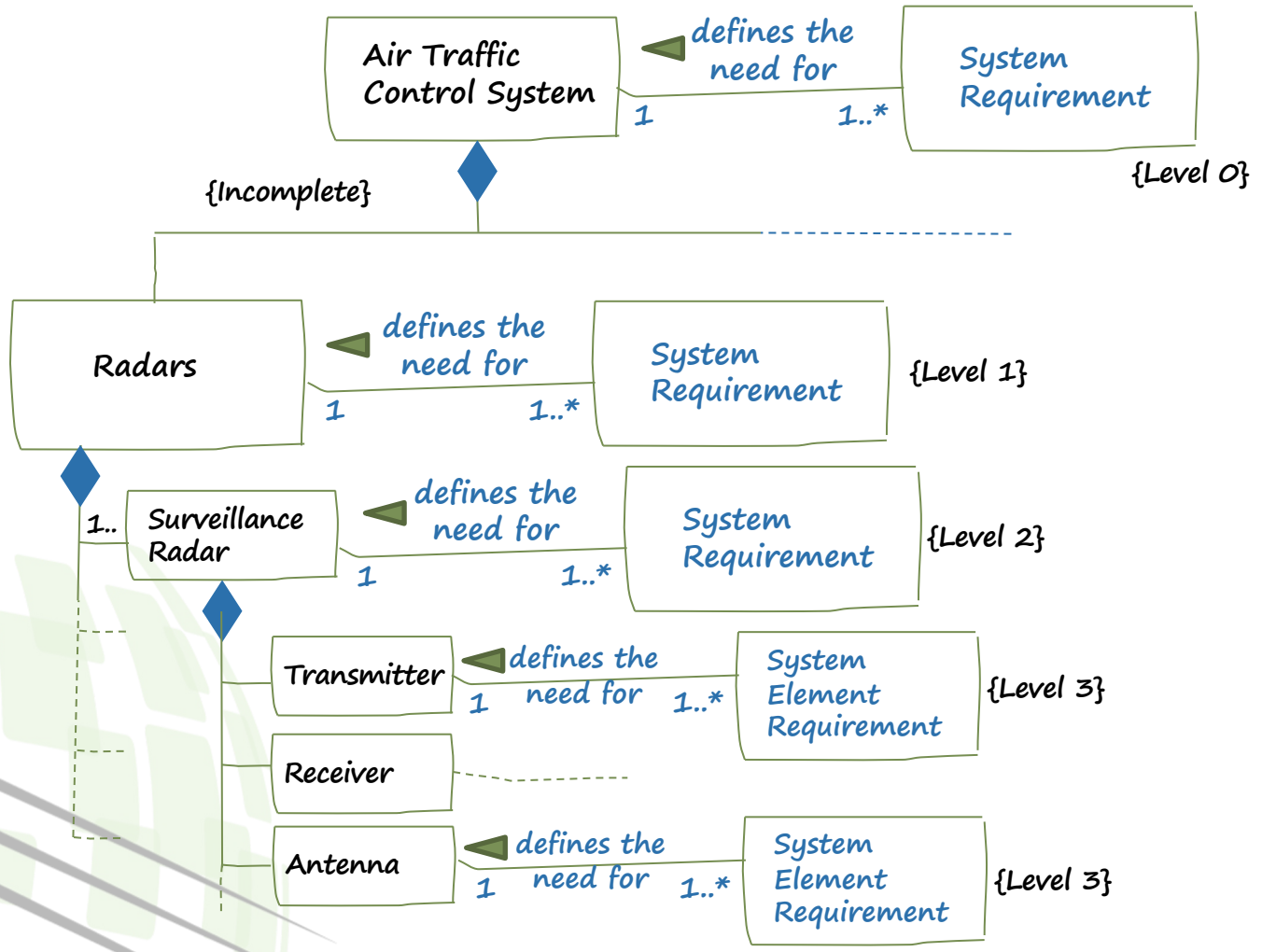
What are we testing?



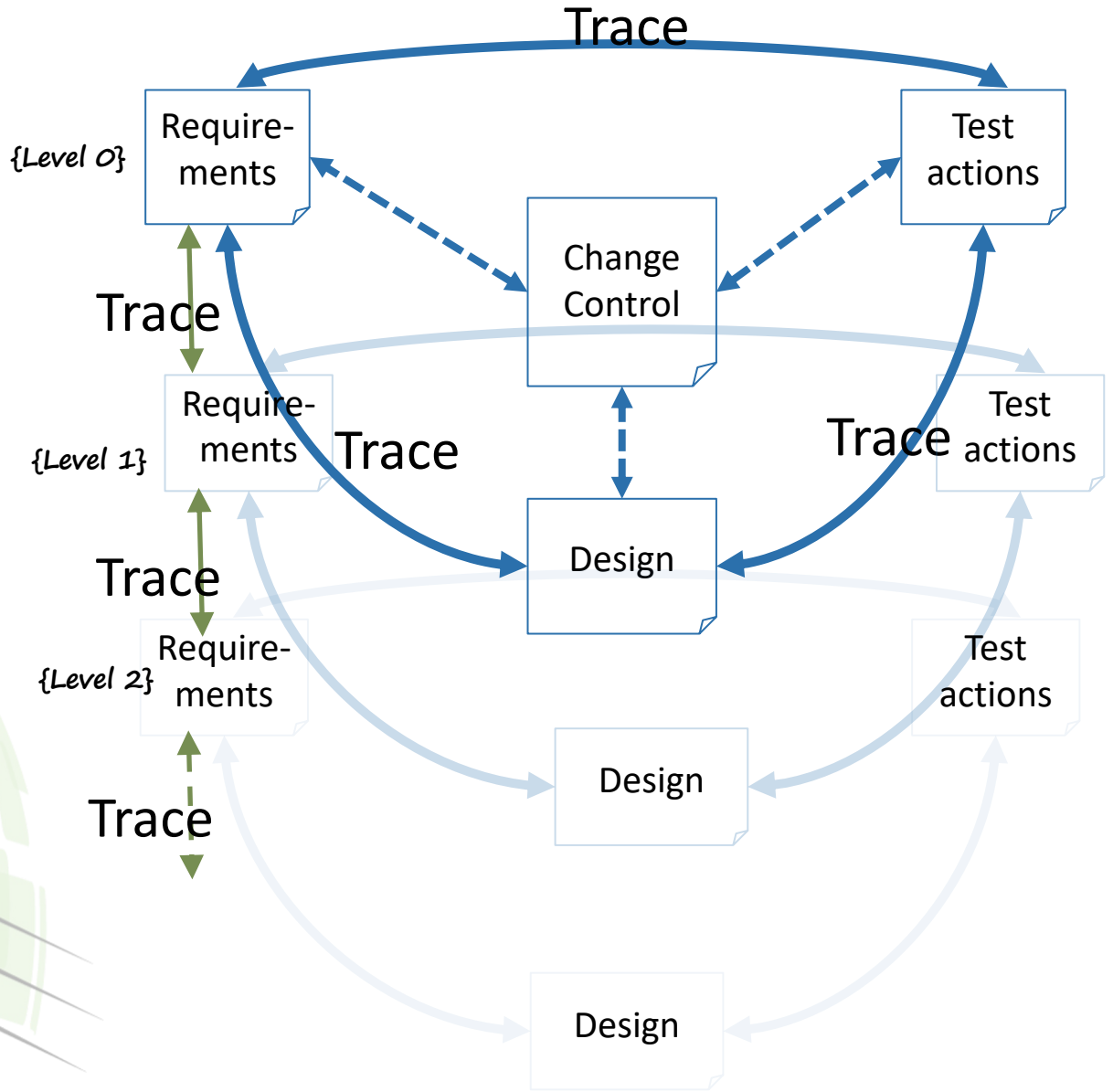
The System Breakdown Structure



Traceability to requirements



3-way hierarchical Traceability



Key Concepts 1

Key Concept: System

A set of interacting System Elements organised to satisfy one or more System Context.

Key Concept: System Element

A basic part of a System, which may itself be a System if it satisfies the above condition.

Key Concept: System Context

The definition of the ‘why’ of a System: what is its purpose, and what does it need to do to achieve that purpose. System Context also includes the external environment in which the System must operate, external Systems with which the System must interact, and Enabling Systems which support the System without direct interaction.

Key Concept: Interface

A shared boundary between two (or more) Systems or System Elements, defined by characteristics pertaining to functional or physical exchanges between them.

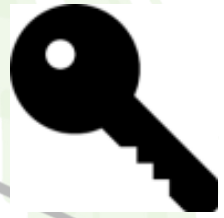
Key Concepts 2

Key Concept: Progressive Assurance

The step-by-step process of compiling evidence that each System Element, each interface, each aggregate of System Elements and finally the whole synthesised System work as intended in its System Context.

Key Concept: Technical Debt

The promise to complete a technical shortcoming in the future while **declaring it complete today.**



Key Concept: Regression Testing

Repeat conduct of a test process on a modified or corrected System or System Element.

Key Concepts 3

Key Concept: Integration

Progressively assembling the implemented System Elements (hardware, software and operational resources) into a realised set of products and/or services that compose the System of Interest (SOI), and verifying the correctness of the functionality, performance, and static and dynamic aspects across Interfaces.

Key Concept: Verification

A process applied to a SOI, or any System or System Element that compose it, to establish that it has been "built right". To provide objective evidence that it fulfils its specified requirements and characteristics.

Key Concept: Validation

A process applied to a SOI, or any System or System Element that compose it, at the appropriate points in the lifecycle stages to provide confidence that the right System (or System Element) has been built. To provide objective evidence that the System, when in use, fulfils its business or mission objectives and stakeholder requirements, achieving its intended use in its intended operating environment.

Key Concept: Acceptance

An activity such that the acquirer (or a representative stakeholder) can decide that the system is ready to change ownership from supplier to acquirer.

Why do we test (so much)?

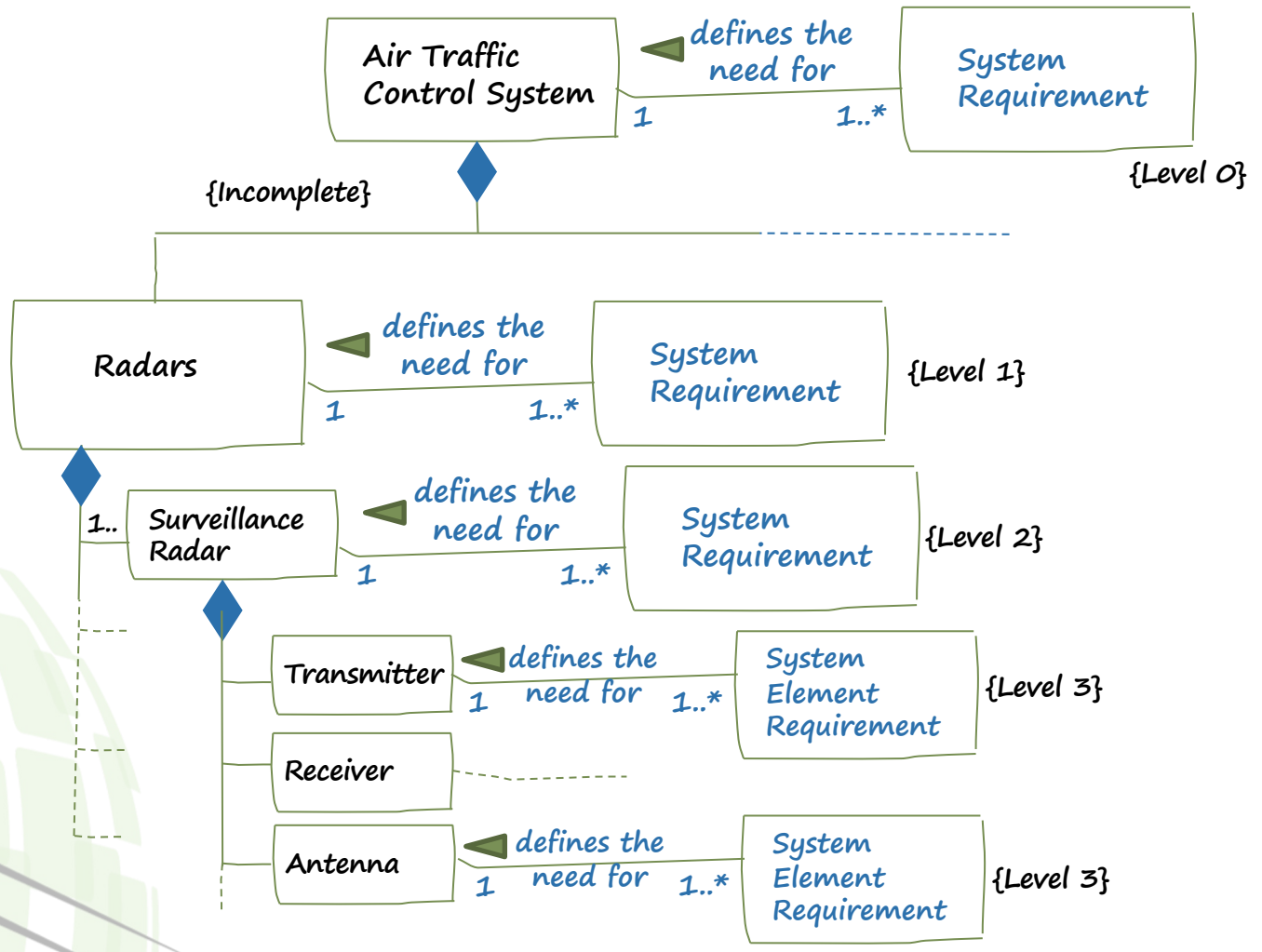
- All Systems have latent errors
- Latent errors mean technical debt
- The sooner you test what you can, find the errors and fix them, the cheaper it is to finish

Cost of Fixing Errors in Projects

Phase at which Error is detected and fixed	Cost to Fix
Requirements	X1 (reference)
Design	X5
Build	X12
Test	X40
Operations	X250

How do we test?

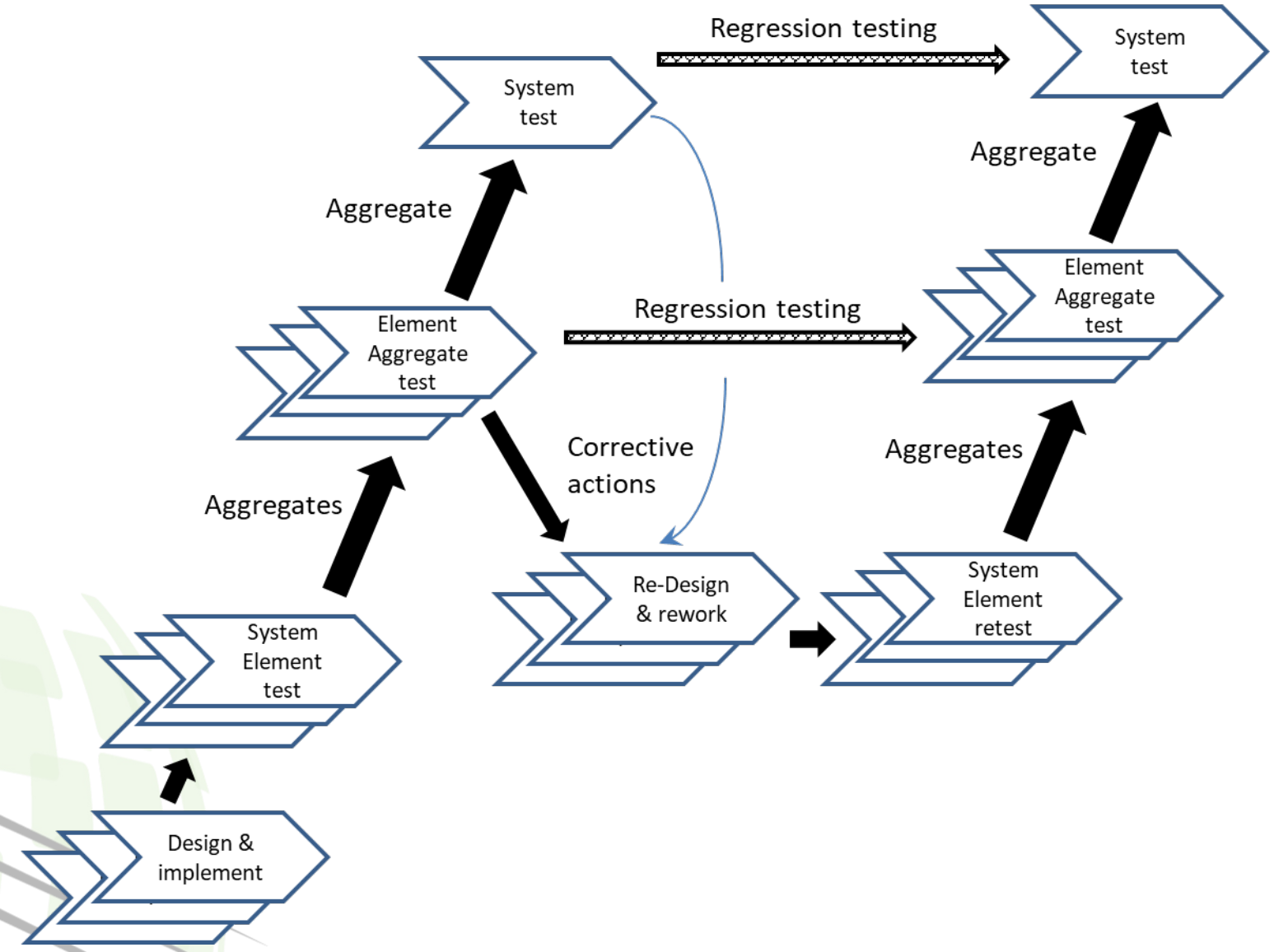
- Generally, start bottom-up
- But see the Integration approaches in the INCOSE Handbook (V4 Section 4.8)
- Don’t forget the interfaces!
- Use simulators, models etc for the elements not done yet



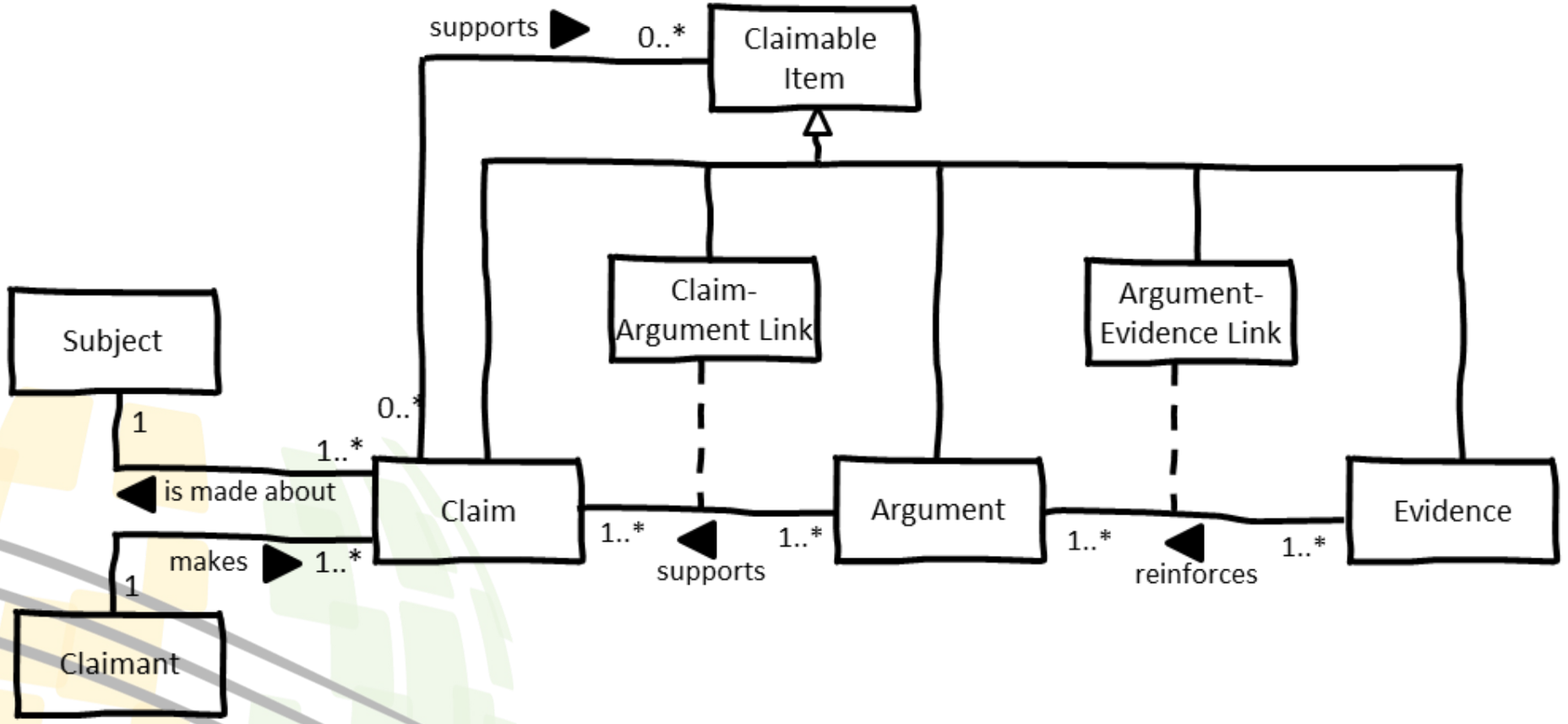
Integration Approaches

Integration Approach	Summary	Application
Global	Also known as 'big bang' – all System Elements assembled in one phase, then tested.	Only if the System is a minor increment of a precedented version. Otherwise, failures are too difficult to diagnose.
With the stream	Integrated into Aggregates as System Elements become available; in any order.	Very hard to plan. Limited to well-controlled developments with low technology risk.
Incremental / bottom-up	Each new Aggregate introduced to the Integration setup adds only one or two extra System Elements.	Best for diagnosing faults, especially where there are multiple levels and many unprecedented System Elements. Can be slow.
Subset	Similar to Incremental, but with parallel subset Aggregates with little cross-connection between Aggregates. Then the subsets are integrated and tested. Can be combined with 'reorganisation of coupling matrices"	Faster than bottom-up, but at some risk on Integration between subsets across interfaces. Works well for Iterative/Incremental life cycles + well-chosen subsets.
Criterion-driven (*sic*)	Critical features (risky, complex, central functionality, immature technology, or other criterion) are integrated and tested first.	Allows longer to detect, diagnose and correct faults; but may involve more complex simulations, test equipment and environments. Very difficult for Systems with multiple levels and long functional chains.

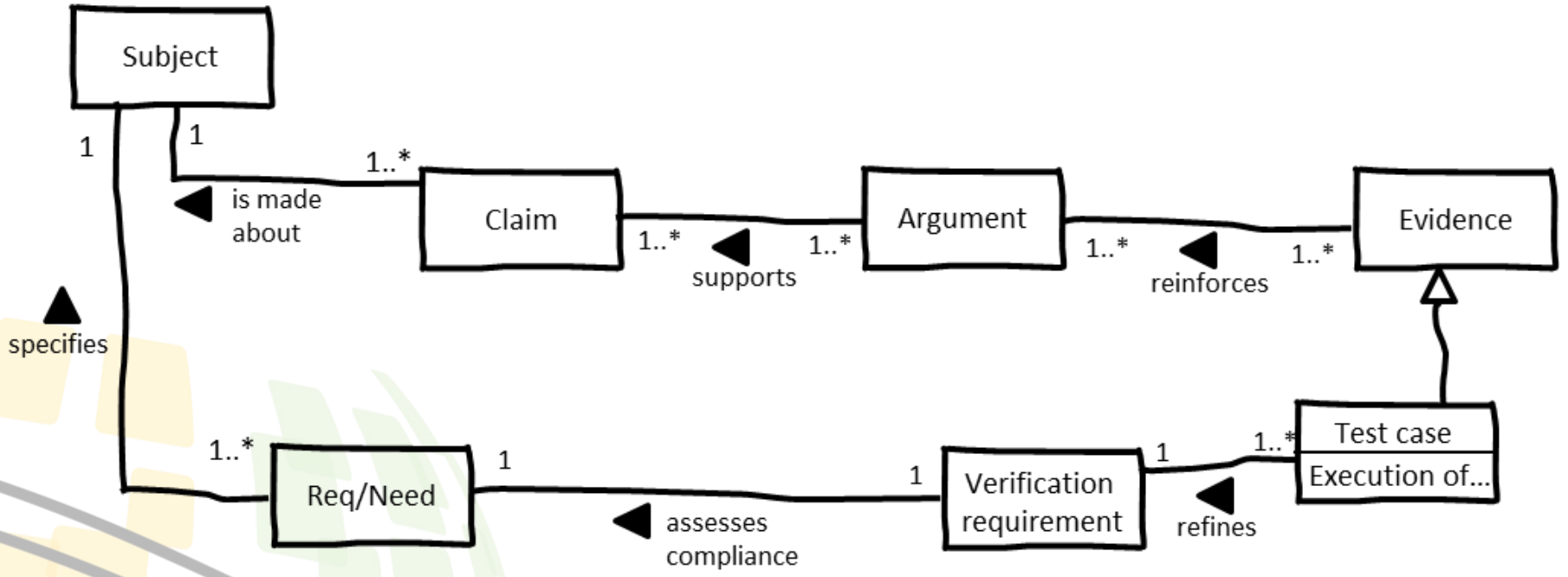
The staircase of Integration



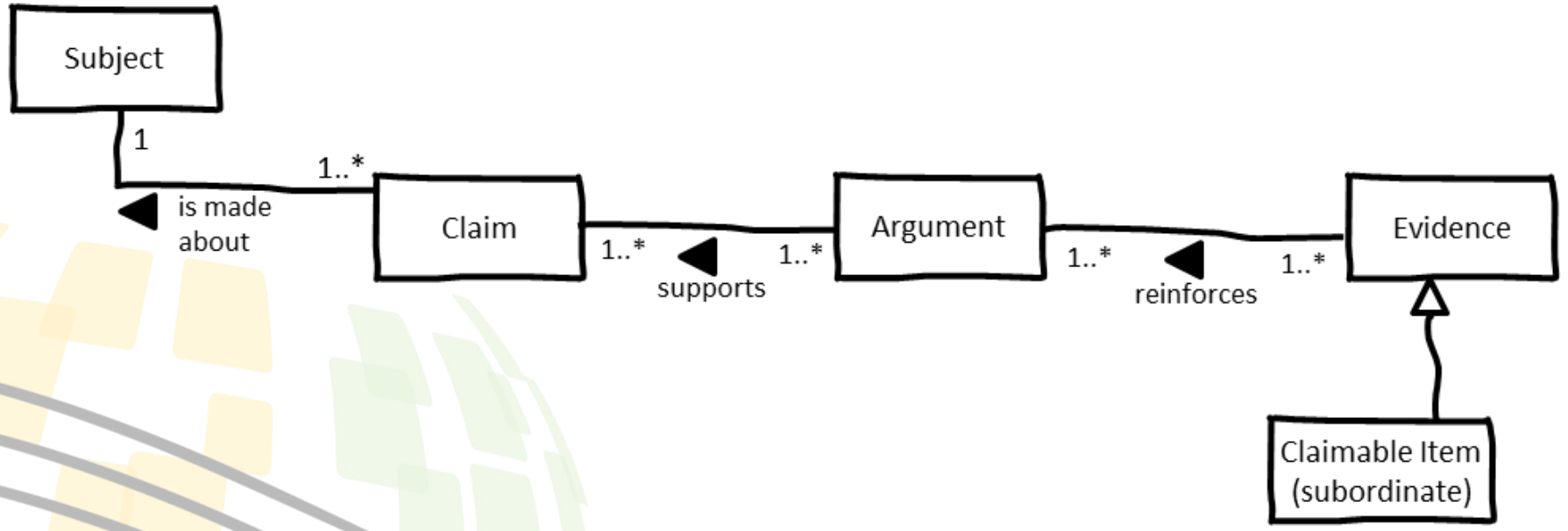
The Evidence Pattern



Relationship to Requirements



Progressive assurance: Claimable Items



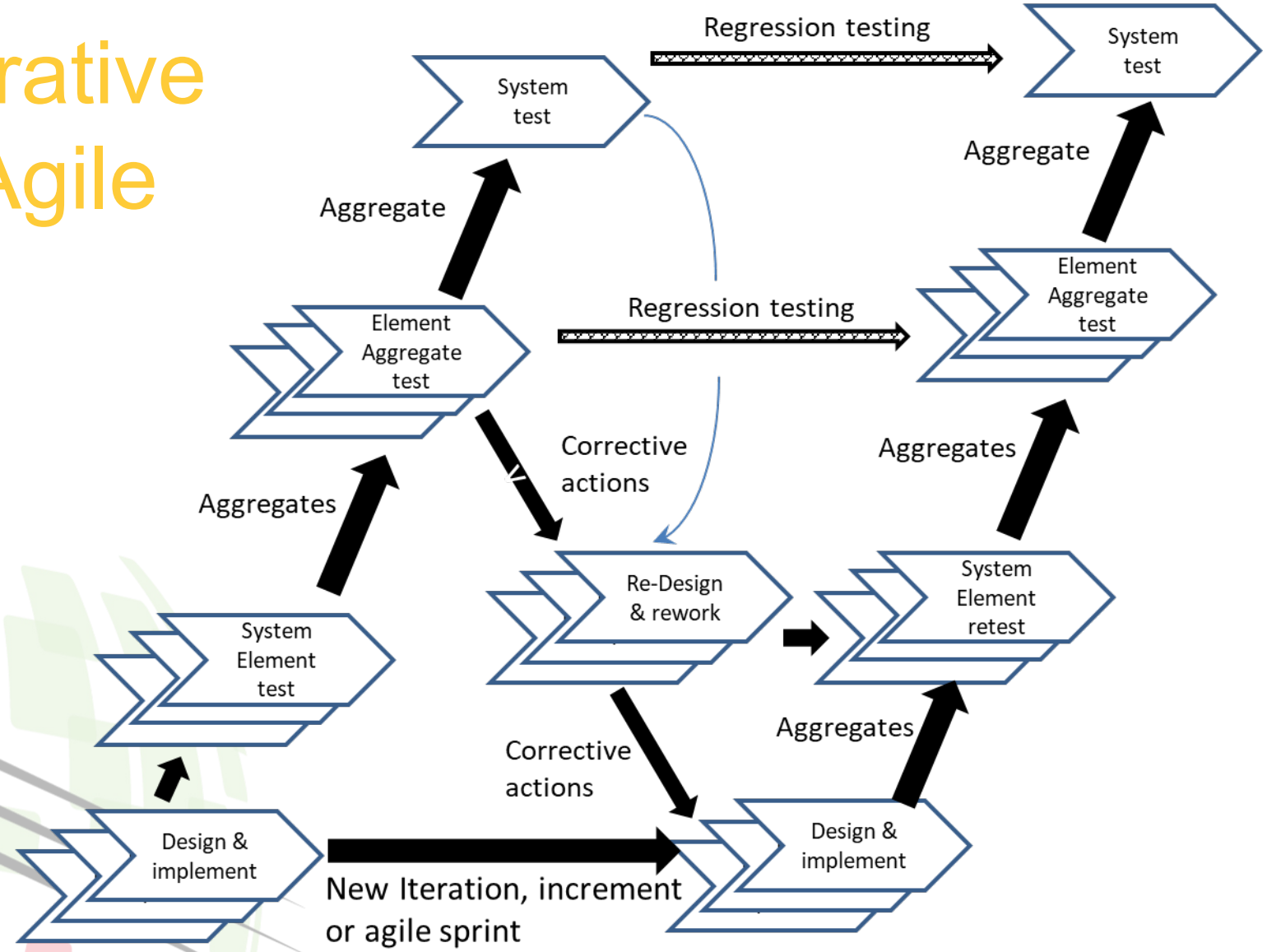
Or Digital Twin result?

Variations on a theme

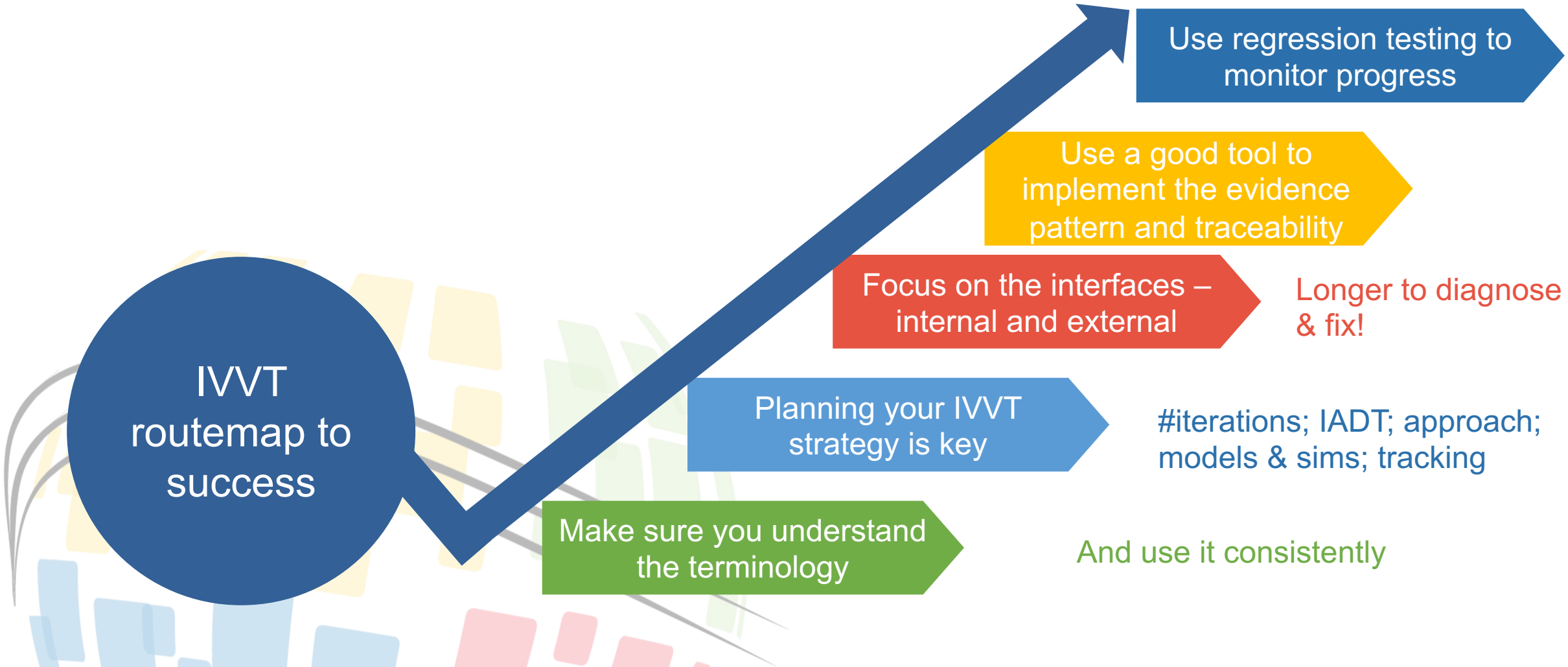
Context	Effects on the Claim – Argument process
IADT	The label 'Test case' in the diagram above is used generically; the test action may also be Inspection, Analysis or Demonstration. This should be referenced in the corresponding Verification Requirement(s), and the Test case itself should comprise a detailed procedure for conducting the action, whether it be I, A, D, or T.
Free play testing	Operation of a System without a test script, with random inputs and operator interaction, can be an extremely efficient way of finding new faults, which can provide evidence for counter-claims.
Envelope testing	Typically, Acceptance is based on the satisfaction of each Requirement (particularly the most difficult conditions) individually. Especially during Integration, it is worth exploring the behaviour of the system under combinations of the most difficult conditions – the 'corners of the envelope' – even if these are not bound to Acceptance. In continuous operation, if these conditions could be imagined to occur, then they will. The capture of such Evidence, in advance, is advantageous.
First article testing versus production	For the first instance of a produced System, it is usually the case that all Requirements need to be proved. For series production, it is usually sufficient to test only a subset, to show that it has been built correctly. The first article results are a set of Claimable Items covering the aspects not subject to repeated or Regression Testing.

And use it consistently

Extension to Iterative / Incremental / Agile



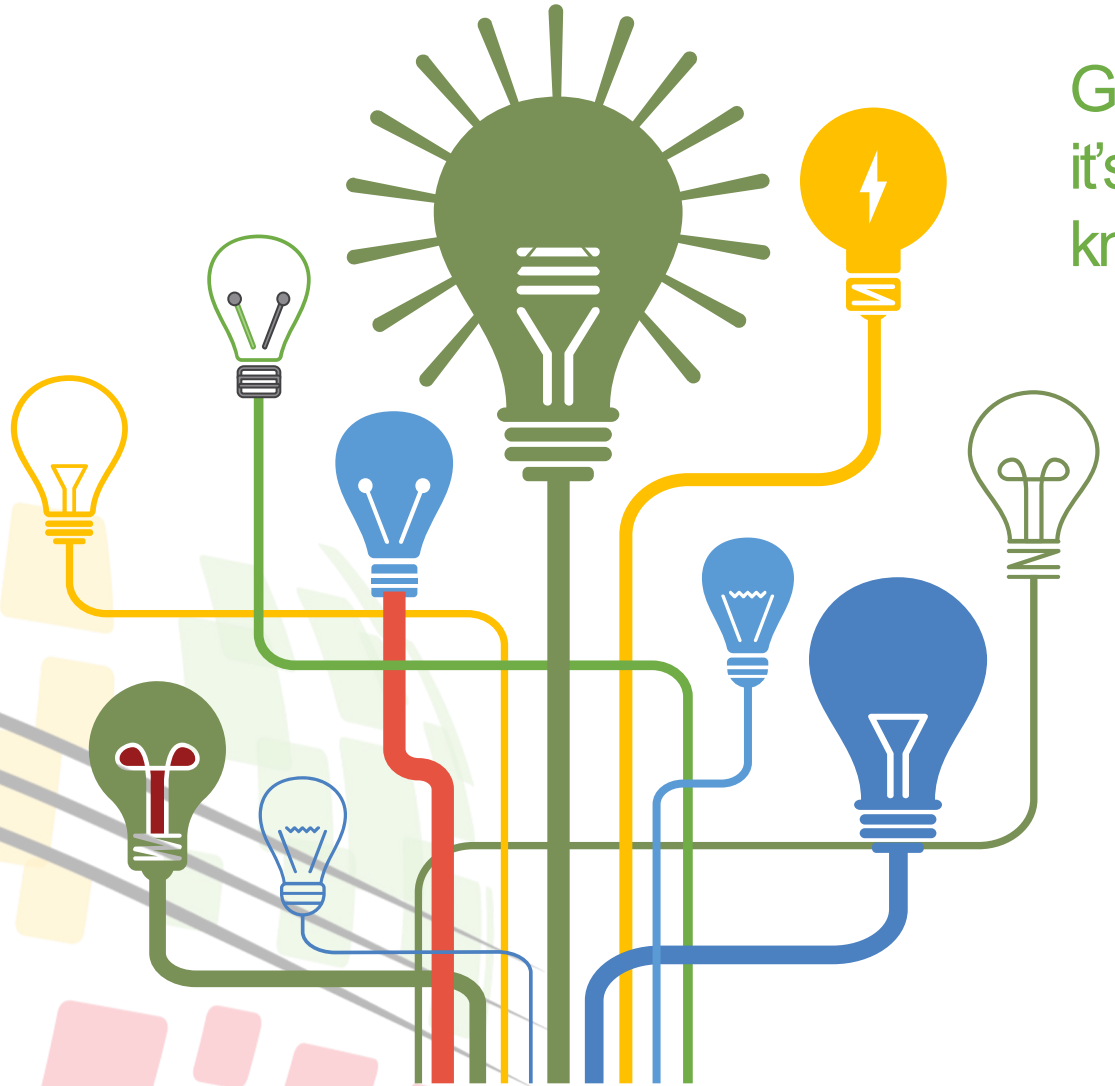
Take-away messages



A concluding thought on career progression

Further reading:

- INCOSE UK Don't Panic! guide 2023
  - "V, V & T of Engineered Systems" (Engel et al, Wiley)
- INCOSE Guide to V&V 2022



Get involved in IVVT – it's a great way to get to know the Systems.

Then (and only then) get involved in Requirements

These are the best stepping stones towards becoming a great System Architect

Get yourself tested!

Q&A…

Paul Davies

33rd Annual INCOSE international symposium

hybrid event

Honolulu, HI, USA
July 15 - 20, 2023

www.incose.org/symp2023
#INCOSEIS