Modular and Evolutionary Safety Cases

Process Developed by:

Industrial Avionics Working Group

Agenda

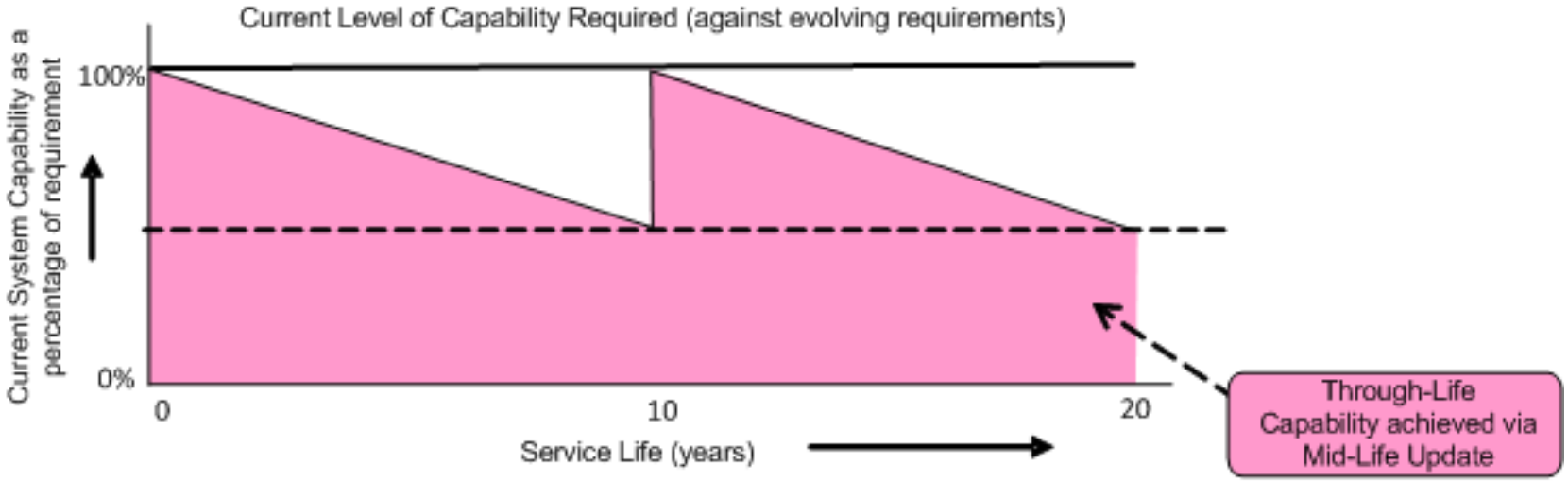
- Motivation
- Basic Concepts
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- Maturity of MSSC
- Deciding to Use MSSC
- Where to Find Out More

Agenda

- **Motivation**
- Basic Concepts
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- Maturity of MSSC
- Deciding to Use MSSC
- Where to Find Out More

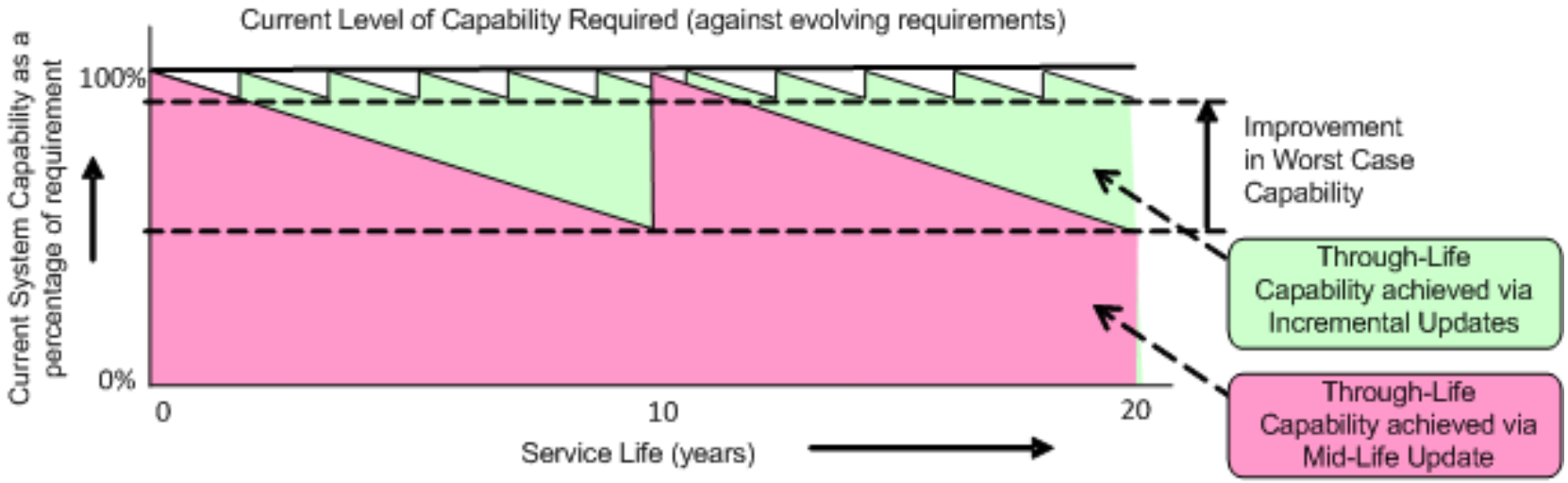
Motivation – Maintaining Peak Military Capability

- High cost of changing Defence Systems has resulted in changes typically being delayed until major upgrade programmes
  - e.g. aircraft mid-life update
- Significant intervening deterioration in capability



Motivation – Maintaining Peak Military Capability (2)

- If costs could be reduced, frequent, smaller 'incremental' changes could be incorporated
- 'Worst case' capability would be significantly improved
- Safety (re)certification is a significant contributor to change costs



Defence System Safety

- In parallel, demonstrating that safety issues around the operation of Defence System have been handled correctly has become a high priority activity

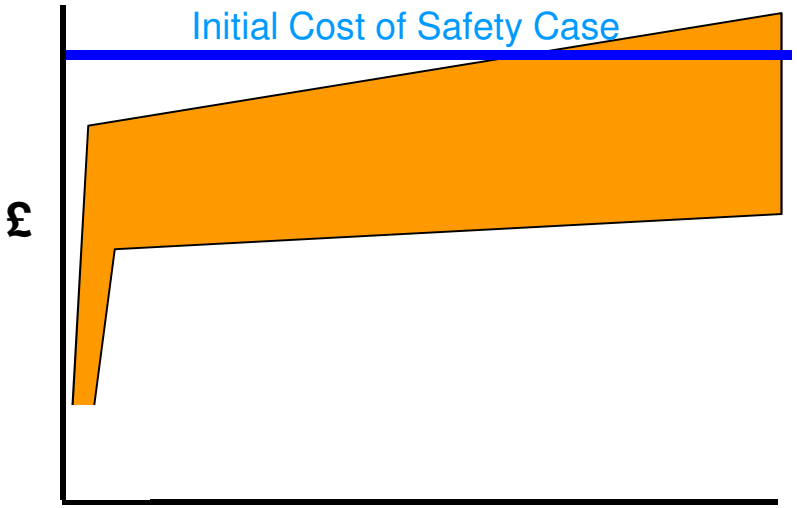
- Safety Cases are currently required for all Defence Systems

- A Safety Case is described as:

*"The **Safety Case** shall consist of a structured argument, supported by a body of evidence, that provides a compelling, comprehensible and valid case that a system is safe for a given application in a given environment."*

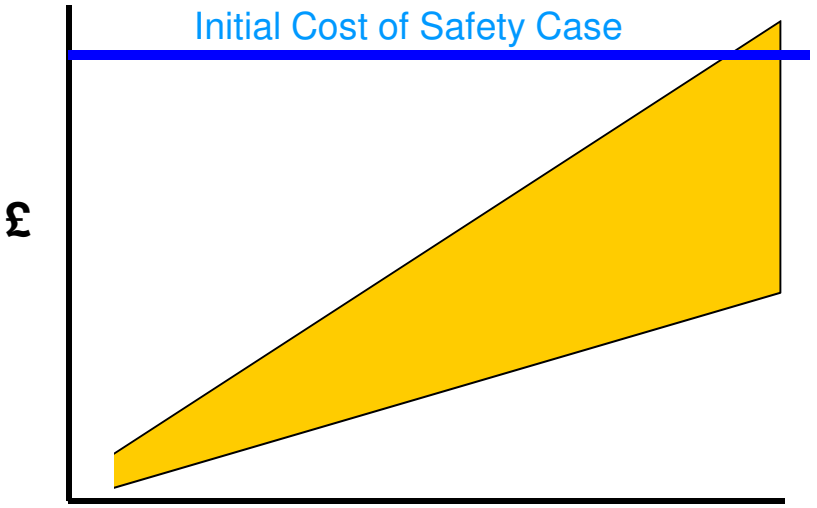
Typical Cost Relationships for Safety Cases

Current



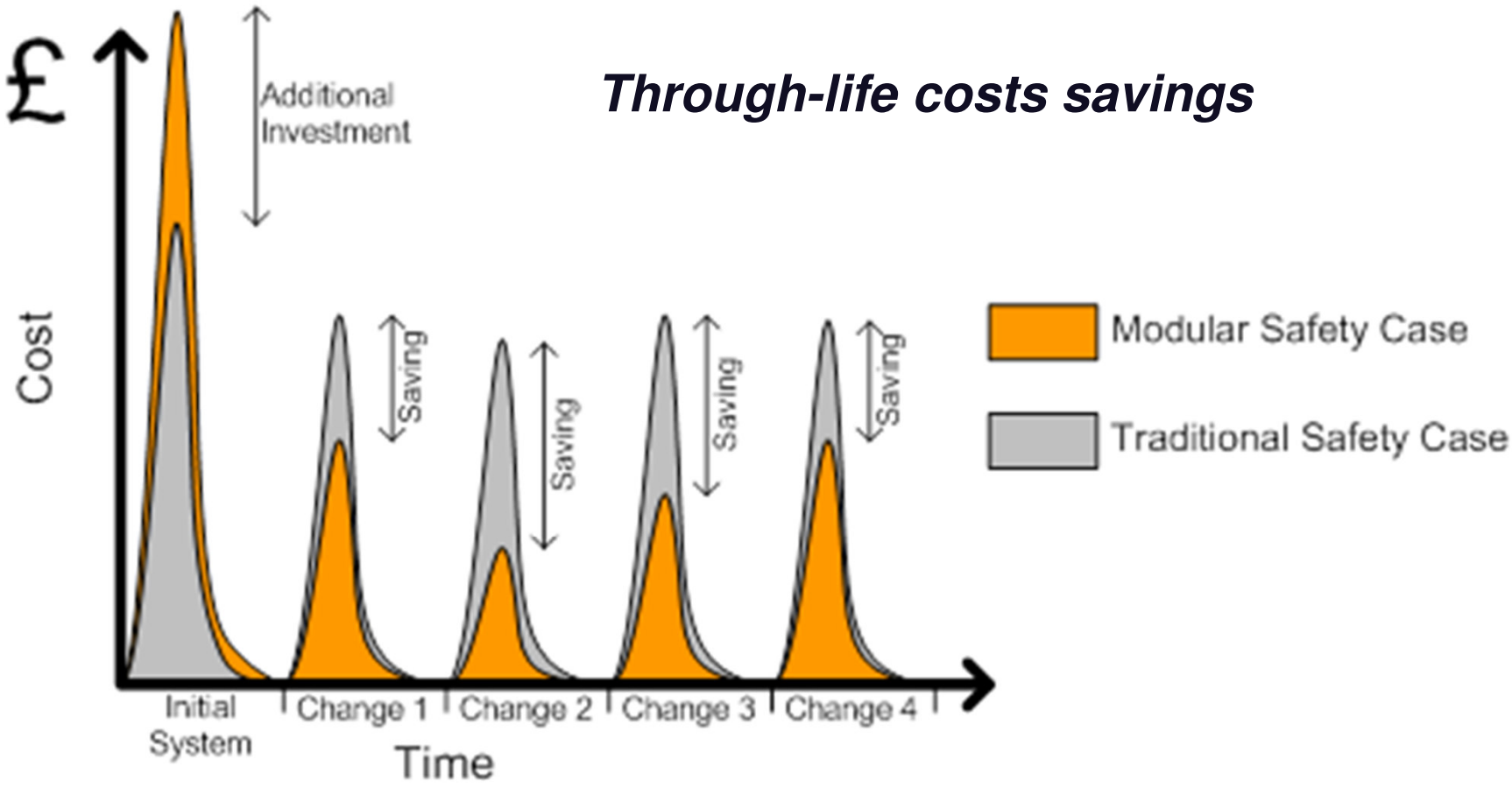
- Cost of re-establishing software safety case is NOT related to the size and complexity of the change.

Aim for the Future



- Cost of re-establishing software safety case is related to the size and complexity of the change.

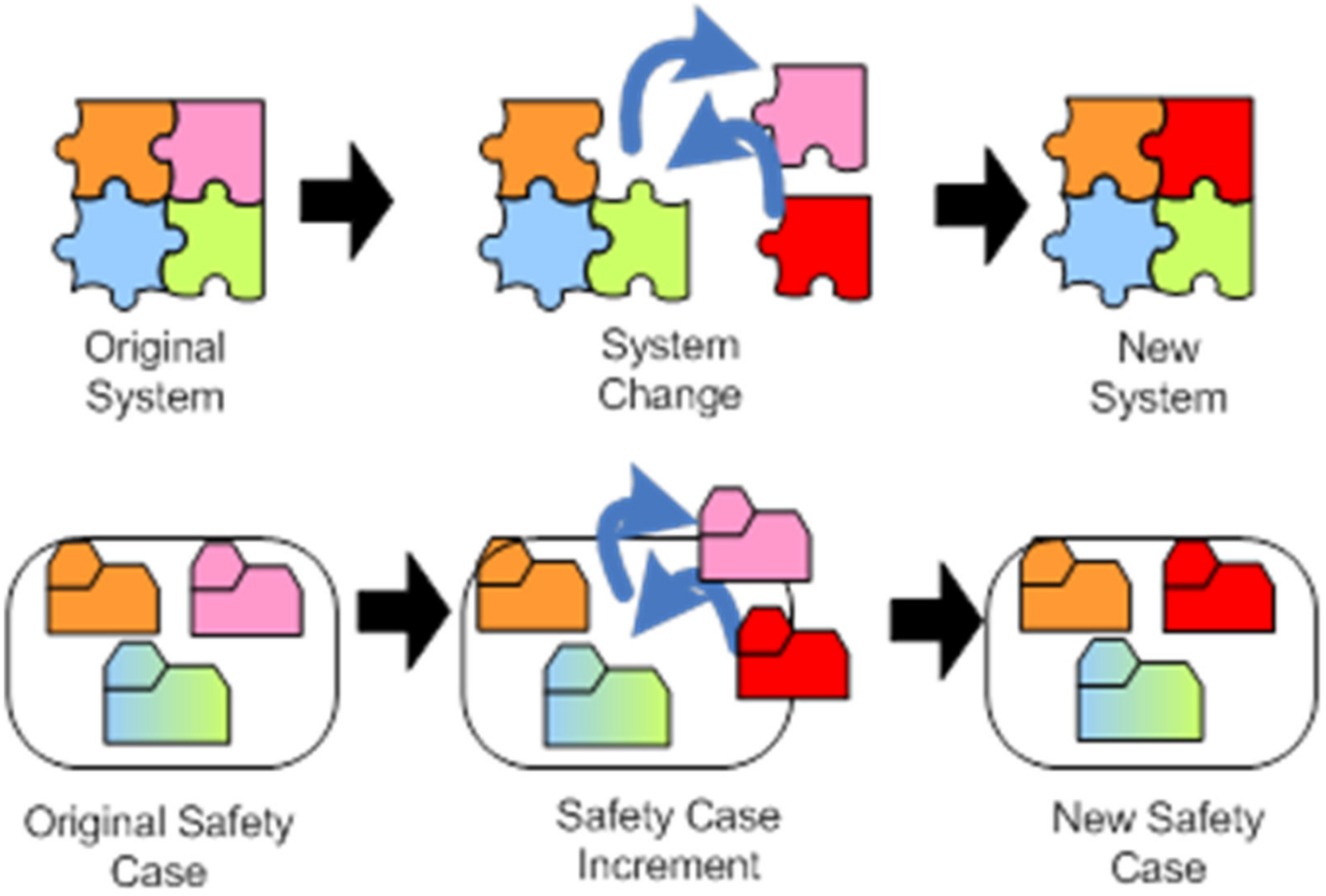
Projected for Modular Safety Cases



Agenda

- Motivation
- **Basic Concepts**
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- Maturity of MSSC
- Deciding to Use MSSC
- Where to Find Out More

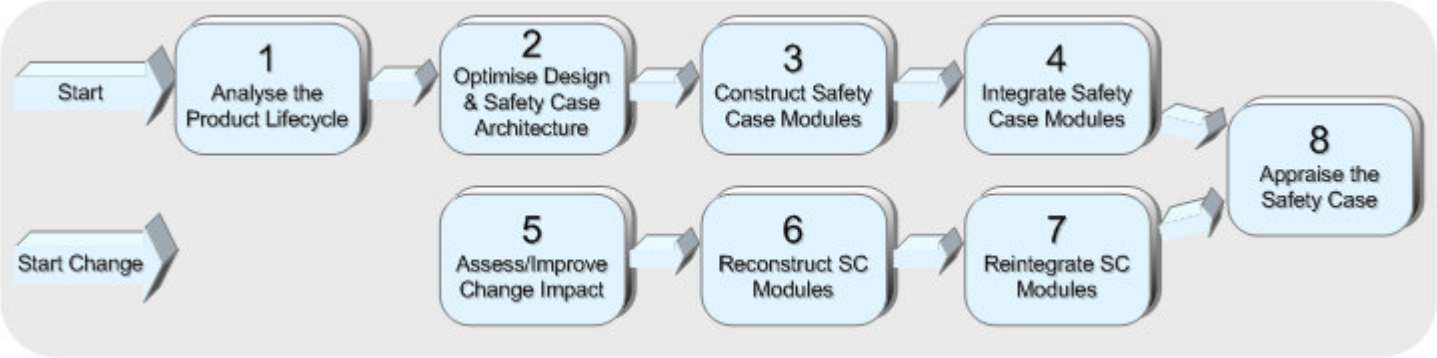
Basic Concept of Modular Safety Cases



Agenda

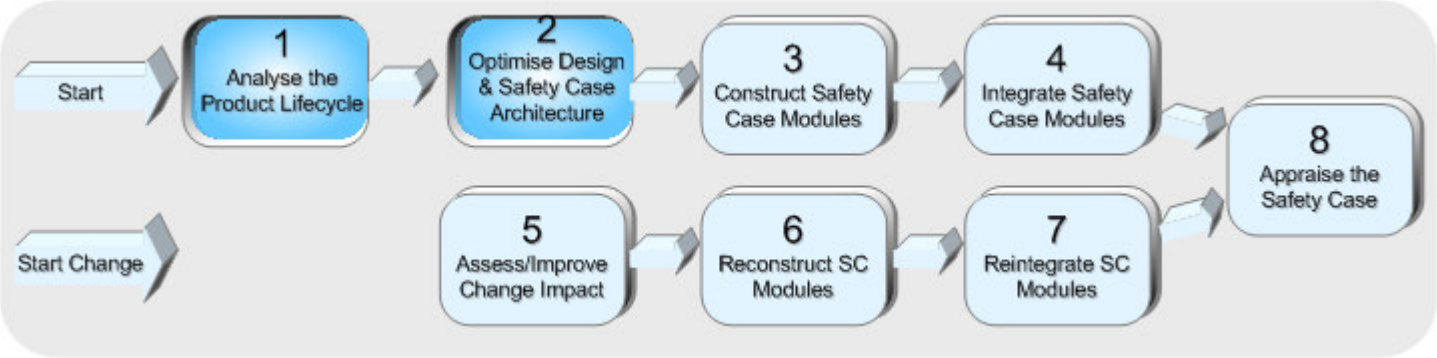
- Motivation
- Basic Concepts
- **Overview of Modular Software Safety Case Process**
- Benefits of MSSC
- Maturity of MSSC
- Deciding to Use MSSC
- Where to Find Out More

Modular Software Safety Case Process Overview



- 8 steps:
- Top row relates to initial development of modular Safety Case
- Lower row relates to changes to the modular Safety Case
- Final step is common to both

Modular Software Safety Case Process Overview



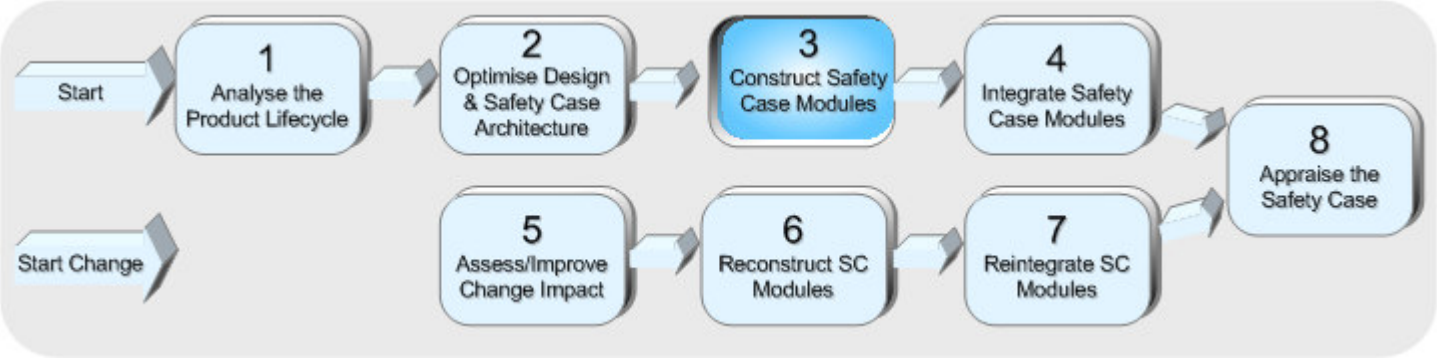
Step 1:

- Predict significant future change scenarios and their likelihood

Step 2:

- Review proposed design modularity and Safety Case modularity together
- Estimate impact of change for each scenario
- Repeat for alternative design and/or Safety Case modularity
- Optimise for change resilience

Modular Software Safety Case Process Overview

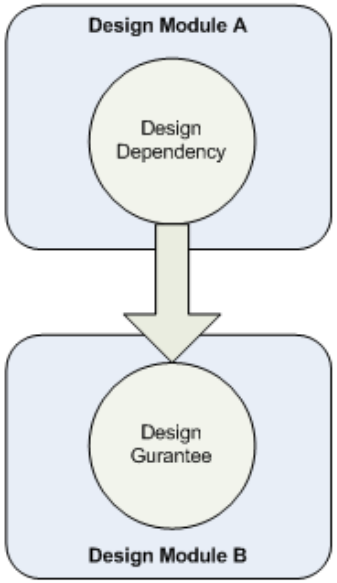


Step 3:

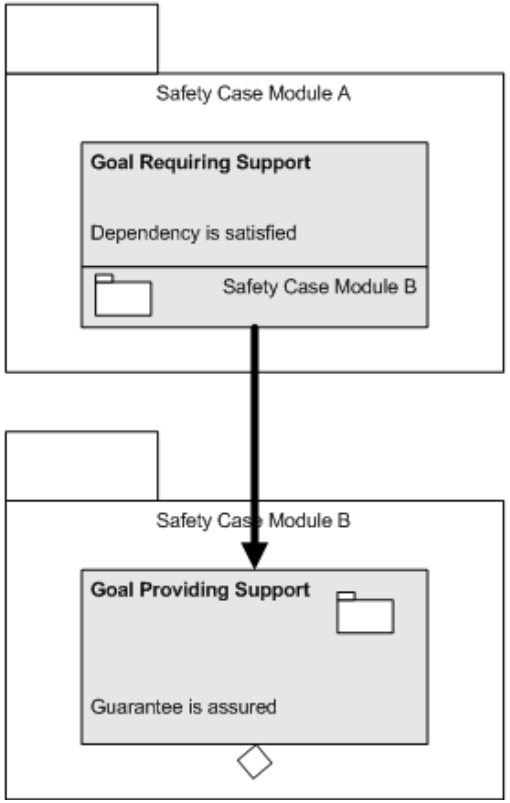
- Define safety properties offered by each module or group of modules, referred to as 'blocks'
- For each block, identify any dependencies on other blocks or the computing environment
- Generate safety argument for assurance of block safety properties, given dependencies are met
  - *Re-usable 'best practice' argument patterns are recorded in MSSC using Goal Structuring Notation (GSN) – see http://www.goalstructuringnotation.info/*

Basic Principles of Modular Safety Cases

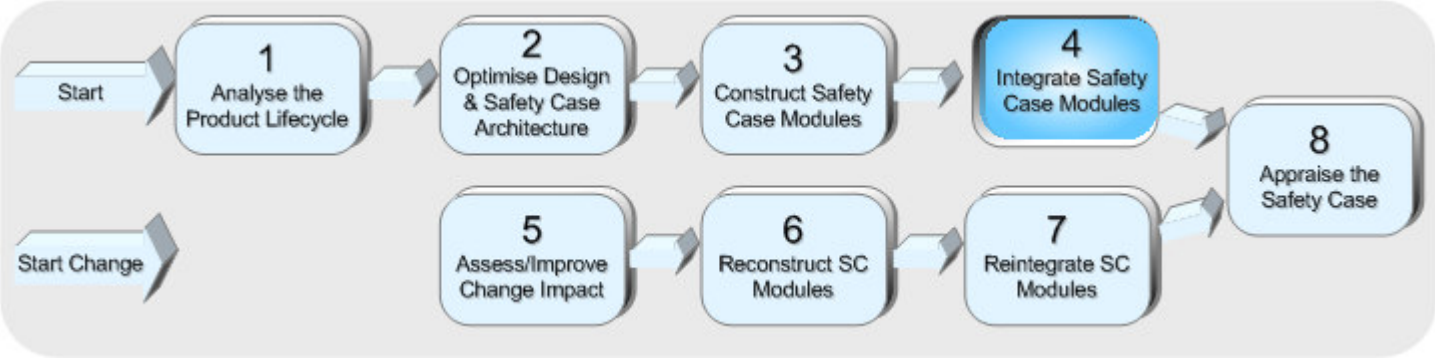
Physical Domain



Safety Case Domain



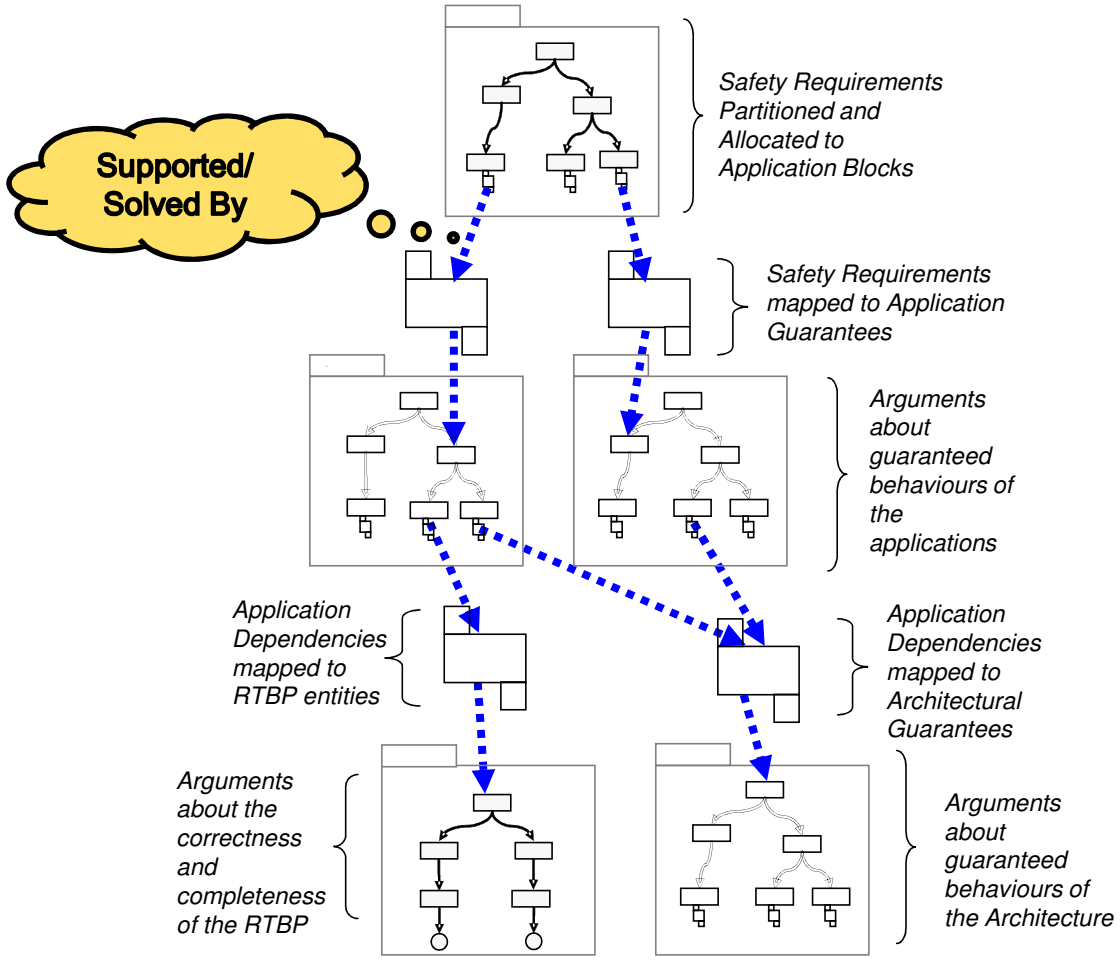
Modular Software Safety Case Process Overview



Step 4:

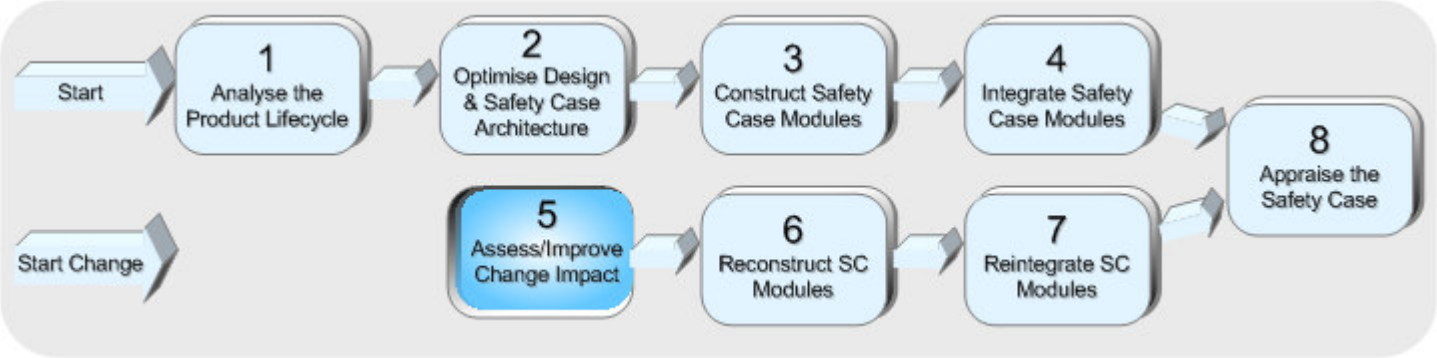
- Integrate 'Goals Requiring Support' with 'Goals Providing Support'
- May be necessary to create additional integration arguments
  - *E.g. end-to-end timing property or system-wide resource usage*

Mapping Safety Dependencies within a Safety Case



Based on a Integrated Modular Avionics System example

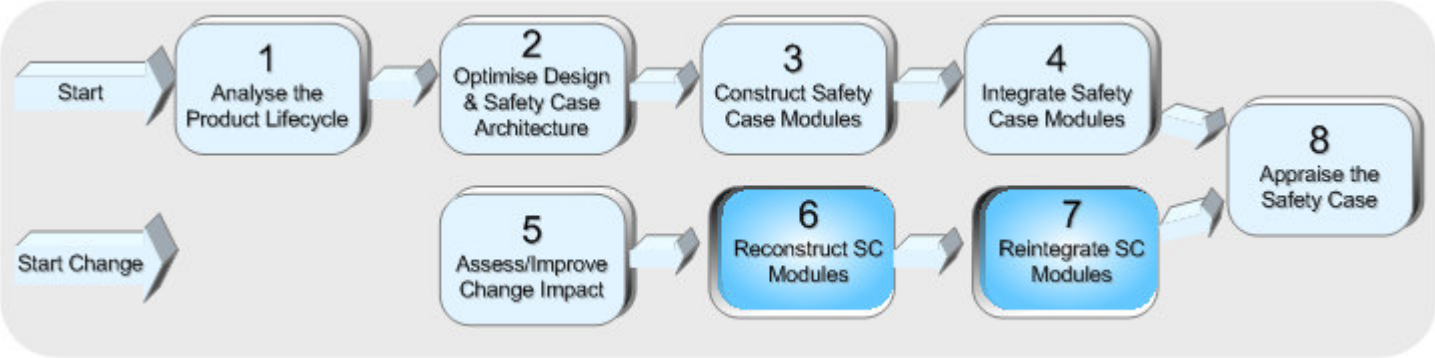
Modular Software Safety Case Process Overview



Step 5:

- Assess the impact of change by understanding the mapping of safety dependencies through the system
- Identify Safety Case modules that need to be developed or changed
- Revisit the Safety Case and design architectures to determine whether they are still optimal

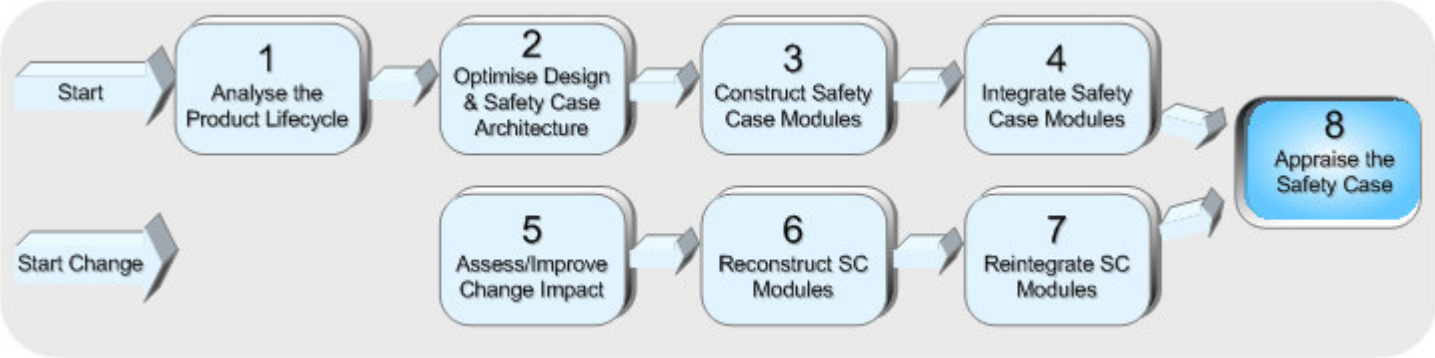
Modular Software Safety Case Process Overview



Step 6 & 7:

- As per steps 3 & 4, but only for modules that need to be developed or changed, as identified in step 5
- Generate 'Change Argument' around the suitability of the output of the change impact analysis process and the rationale for not re-visiting the unchanged Safety Case Modules

Modular Software Safety Case Process Overview



Step 8:

- Review whether the completed safety case is expected to achieve, or has achieved, the desired change containment defined at steps 1 & 2

Any questions on the process?

Agenda

- Motivation
- Basic Concepts
- Overview of Modular Software Safety Case Process
- **Benefits of MSSC**
- Maturity of MSSC
- Deciding to Use MSSC
- Where to Find Out More

Benefits of MSSC

Disciplined recording of safety-related properties and dependencies across a system supports:

- ***Module Replacement*** – if a changed module meets the same safety properties as the original module, the remainder of the Safety Case is not affected
- ***Change Containment*** – if a changed module does NOT meet the same safety properties as the original module, the 'map' of inter-dependencies help to identify the impact of change
- ***Distributed Team-Working*** – authorship of Safety Case modules can be more effectively managed on the basis of understanding the inter-dependencies
- ***Protection of Intellectual Property*** – implementation detail of designs and Safety Case modules can be protected from other authors as only the interfaces need to be made 'public'. Helps with IP and/or Export Control

Agenda

- Motivation
- Basic Concepts
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- **Maturity of MSSC**
- Deciding to Use MSSC
- Where to Find Out More

Maturity of MSSC

- Concepts have been demonstrated to TRL7
  - Included a parallel certification activity of a complex avionic system on a fixed-wing aircraft project
  - Refinements have been made through additional trials on rotary-wing systems
- Not currently in use on an in-service aircraft, but will be used on AgustaWestlands' Wildcat Helicopter
- Also being used on research programmes within IAWG companies

Agenda

- Motivation
- Basic Concepts
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- Maturity of MSSC
- **Deciding to Use MSSC**
- Where to Find Out More

Deciding to Use MSSC

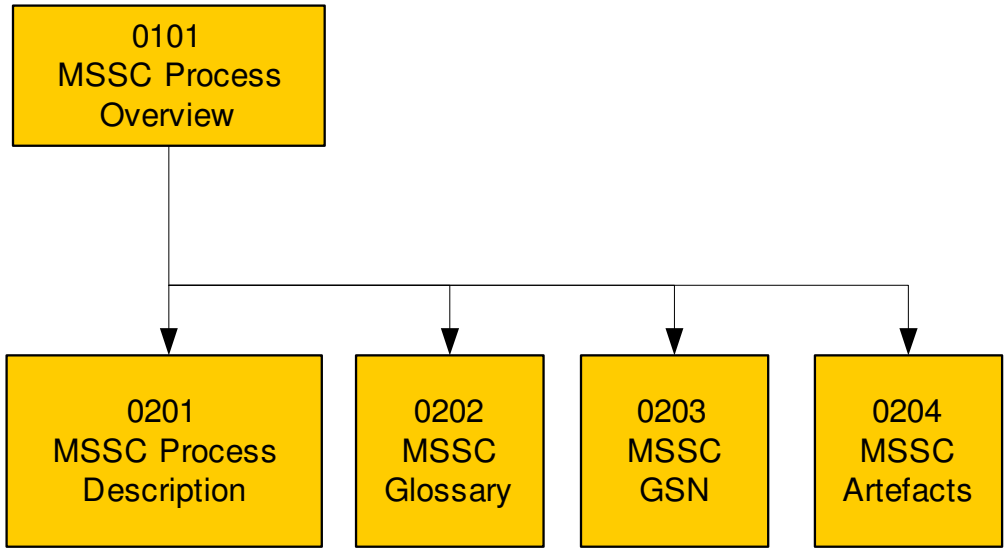
- Criteria for '**receptivity**' to using MSSC have been established:
  - *Maximising Benefits:*
    - System Size and Complexity
    - Anticipated Change
  - *Predicting Effectiveness:*
    - Design Modularity
    - Reusability
    - Use of COTS/3rd party/Legacy Software
- Spreadsheet-based tool support available to assess receptivity
  - See *Capability Agility website*
- Tool Support – no specific notation or tools are mandated for MSSC
  - Tools that support GSN are listed on *www.goalstructuringnotation.info*
- Training – contact IAWG companies for MSSC-specific training

Agenda

- Motivation
- Basic Concepts
- Overview of Modular Software Safety Case Process
- Benefits of MSSC
- Maturity of MSSC
- Deciding to Use MSSC
- **Where to Find Out More**

Where to Find Out More

- the Capability Agility website has material about MSSC and supporting guidance and processes



www.capability-agility.co.uk

david.short@baesystems.com *or*

charlie.hewitt@baesystems.com

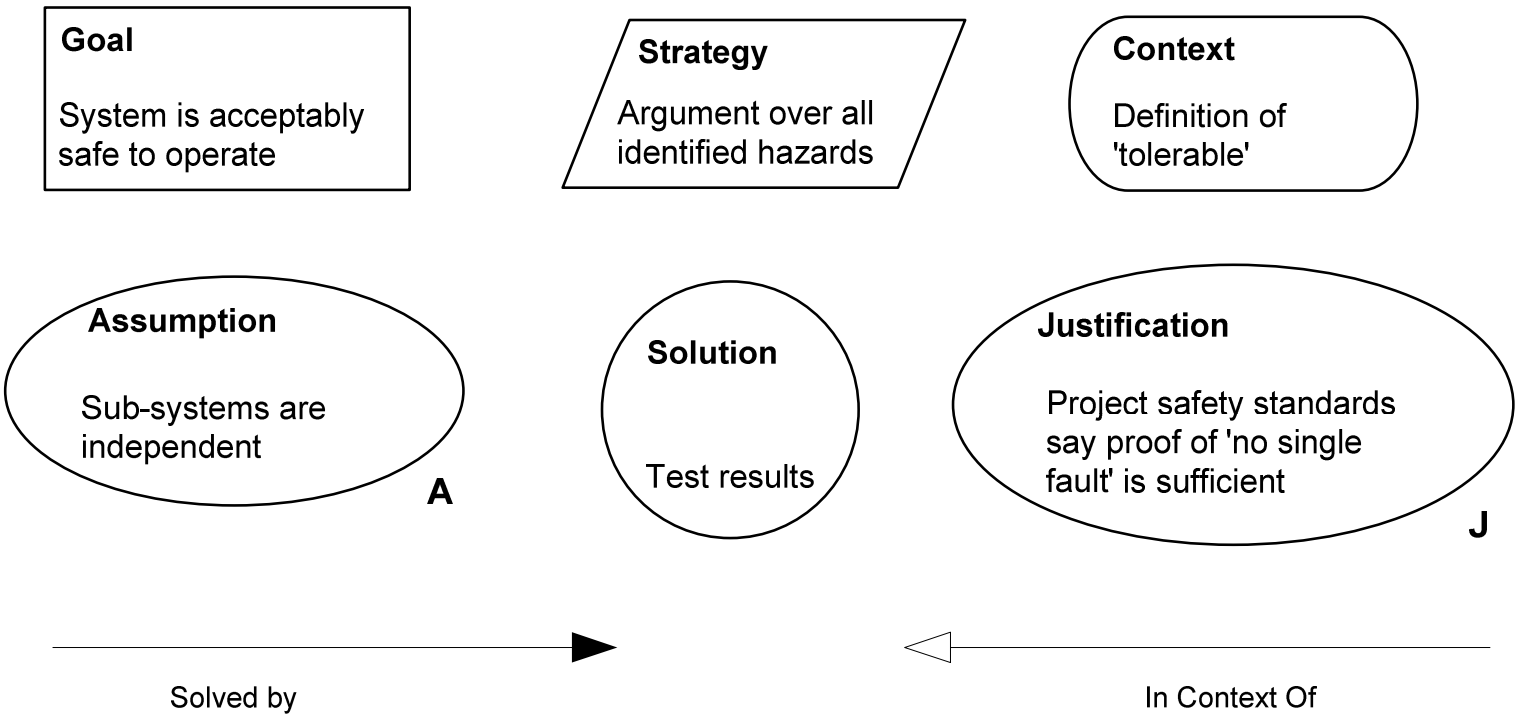
Any Questions?

BACK-UP MATERIAL

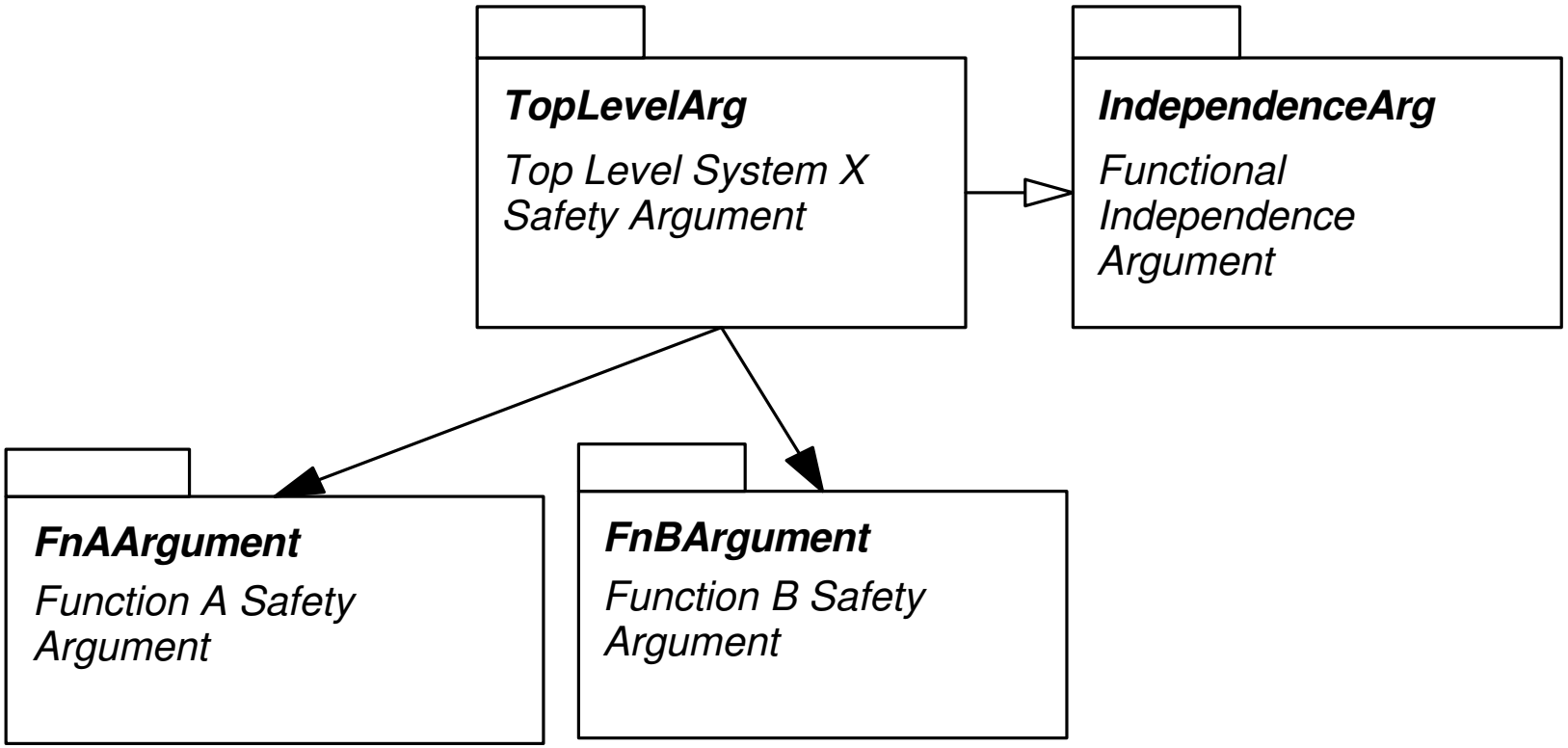
Technical Detail of Process

BACK-UP MATERIAL

Goal Structuring Notation - Symbols

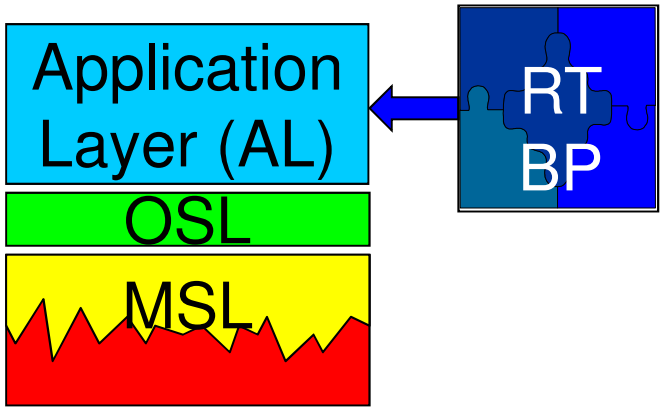


Modular GSN – Module View/Safety Case Architecture

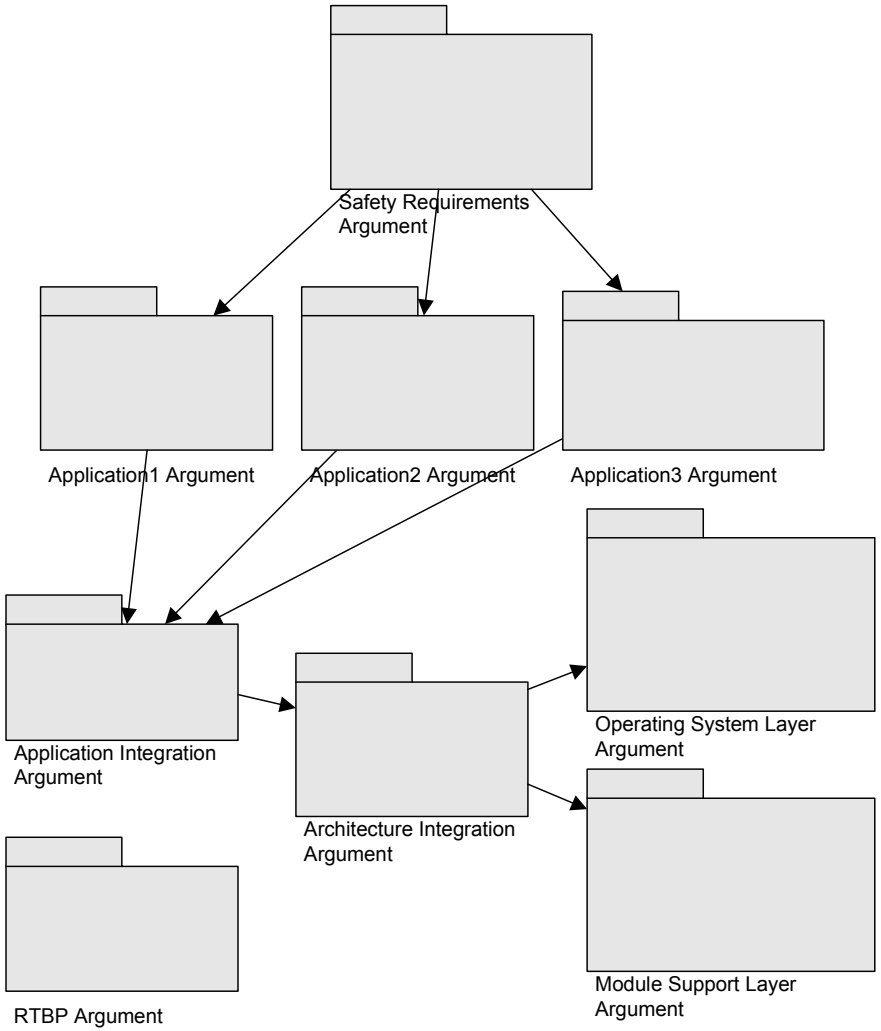


ASAAC IMA Architecture Example

Design Architecture

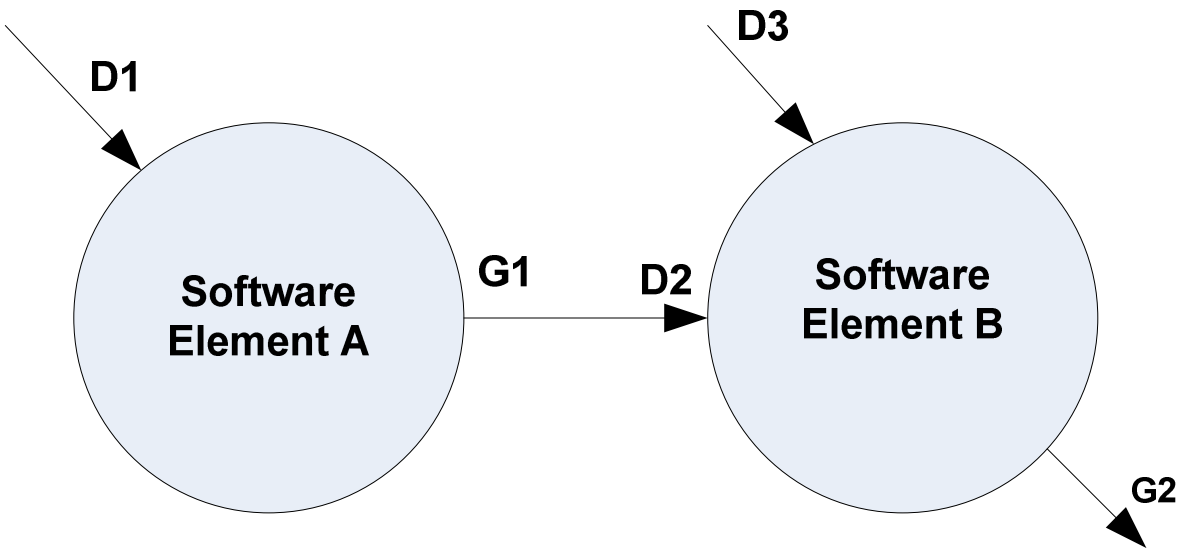


Safety Case Architecture



More recently, IAWG looked/looking at other types of ‘layered’ architectures

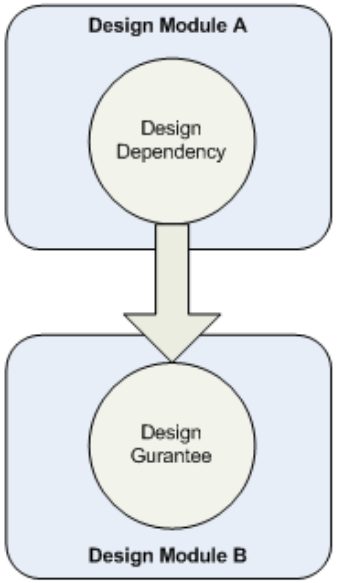
Dependency-Guarantee Relationships & Contracts



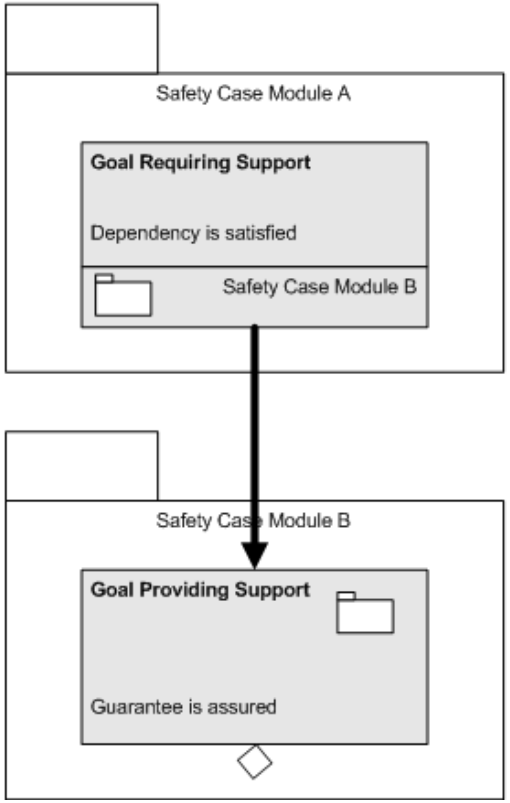
- Identify Dependency-Guarantee Relationships for each software elements of the design
- Identify Dependency-Guarantee Contracts between software elements, where appropriate

Basic Principles of Modular Safety Cases

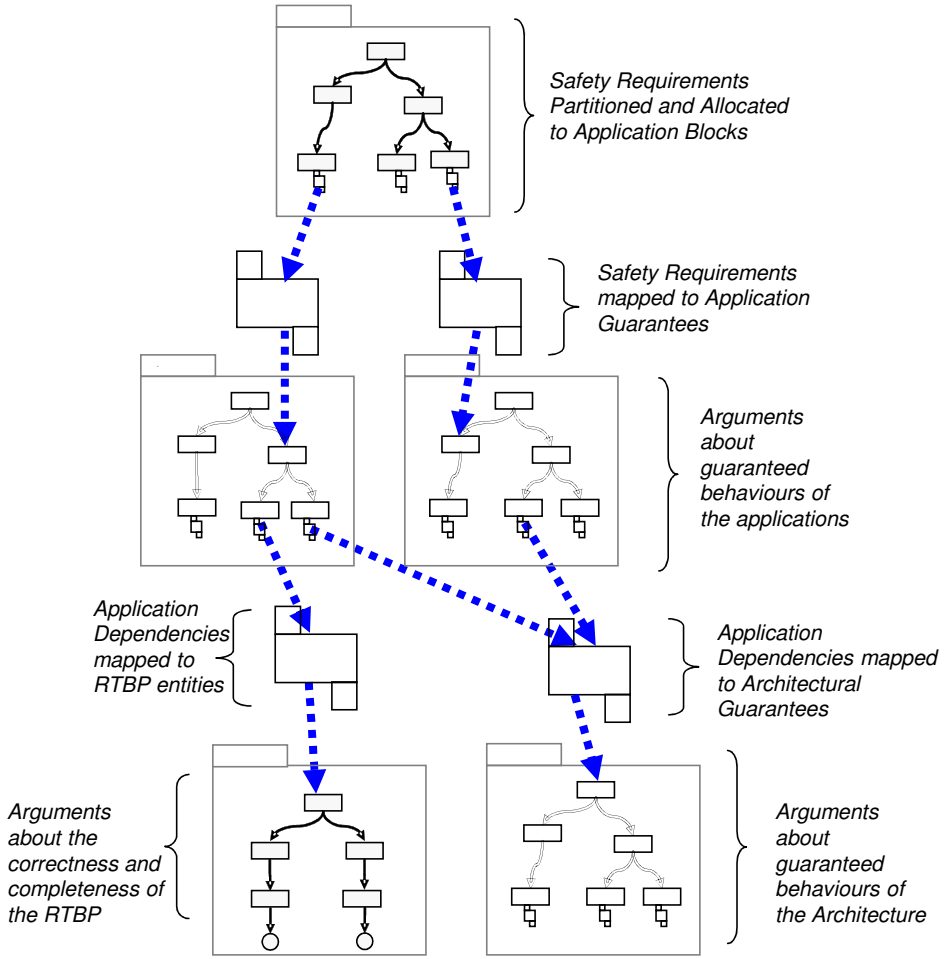
Physical Domain



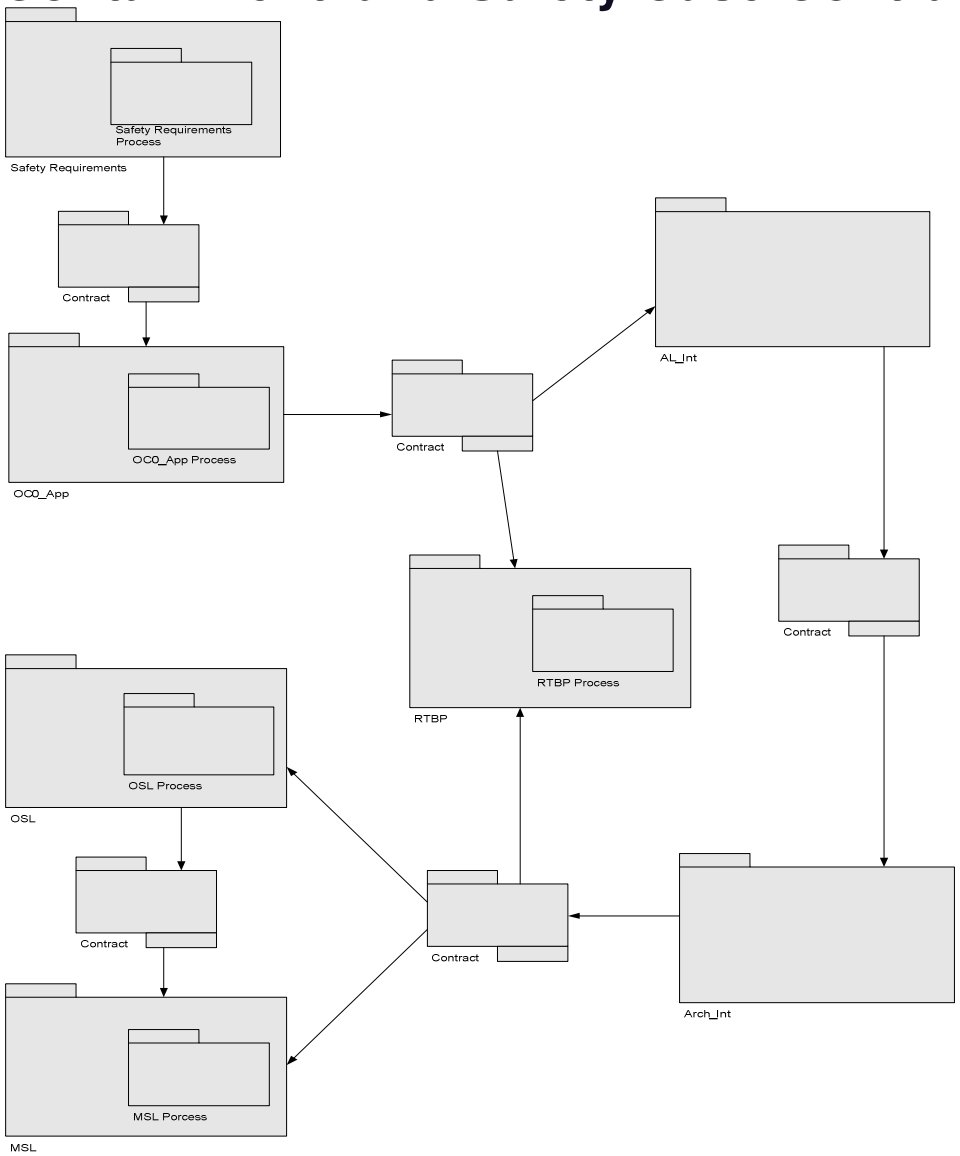
Safety Case Domain



‘Daisy-Chain of DGRs’



Safety Case Architecture Integration using 'Containment' and Safety Case Contracts



Lessons Learned, so far…….

- **Levels of abstraction**
  - Easy to get ‘lost’ in the detailed argument within a module and lose sight of the ‘bigger picture’ – need a ‘safety case architect’?
- **Some design boundaries don’t make good modular safety case boundaries!**
  - High coupling causes high set-up costs for defining DGRs and impact of change is likely to be high, e.g. porting legacy applications
- **Evidence needs to be modular**
  - Unintended evidence ‘coupling’ can defeat modularity in the argument structure
- **Context compatibility**
- **Logistical Challenges**
  - **E.g. Safety Case Report Document Structure -** Unintended coupling of safety case modules can exist by inappropriate documentation/document numbering/referencing