McMaster University

Inspiring Innovation and Discovery

Assurance Cases in Determining Safety of a System with Embedded Digital Devices

Alan Wassyng
McMaster Centre for Software Certification
Department of Computing and Software

NRC Workshop EDD Safety: Oct 9, 2014

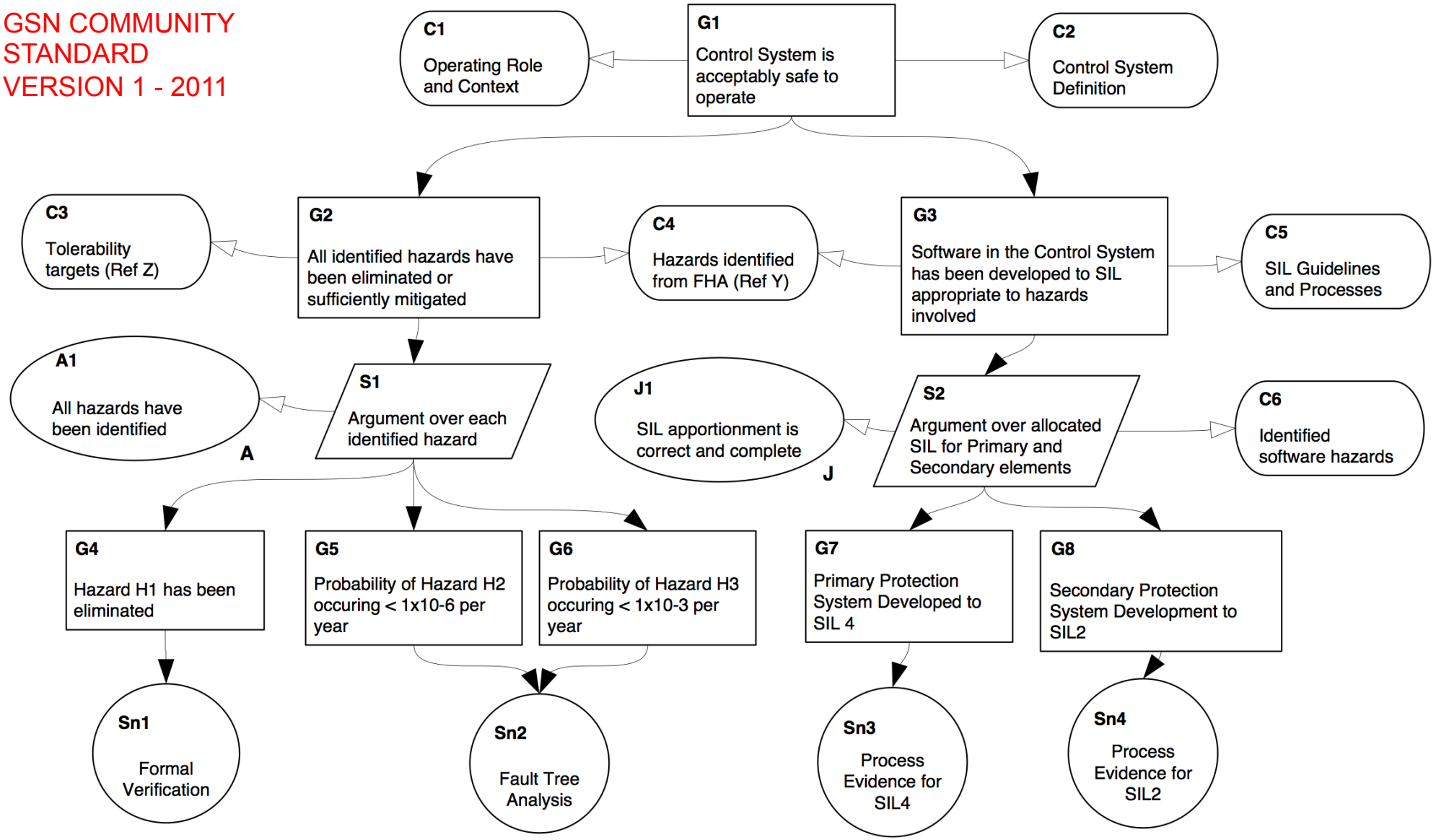
McSCert

Assurance Cases

- Original version was a Safety Case
- In use in the UK for decades, now in use throughout Europe
- Recent interest in the US & Canada
- What is an Assurance Case?
  - Structured argument that links a claim to evidence that supports the claim
- "Claimed" benefit
  - Makes the argument explicit
- Notation
  - Many – but the most popular is Goal Structuring Notation (GSN)

Example of an Assurance Case Segment

GSN COMMUNITY STANDARD VERSION 1 - 2011



McSCert

Problems with Assurance Cases

- Many people think you develop an assurance case simply to show to regulators
- If every argument is structured differently, how will regulators cope?
- The argument is NOT explicit! How do we create and evaluate this kind of argument?
- Most people structure the assurance case by showing hazards are mitigated
- Assurance cases eventually depend on evidence – what evidence do we know how to generate and evaluate?
- How do we evaluate the confidence we have in our assurance case?

McSCert

Assurance case developed after the fact

Assurance Case Template

- Let's start at the beginning
  - Developers of safety critical systems do not develop the system and then ask "is it safe?"
  - They plan the system to be safe
  - That is one (really good) motivation for an assurance case template
    - It can drive the development so that the product will be dependable
    - It provides assistance through experience and planning
    - It informs the developers about evidence they should target and why they need it
    - It still allows flexibility, some parts of the template are not yet complete because the work is still to be done – but the direction is there
    - (An assurance case template is not the same as an argument pattern)

McSCert

Assurance Case Template

One-off argument structure

- There is no (good) reason for each assurance case within an application domain to be structured totally differently
  - Safety integrity levels (SIL) are domain specific, and with a SIL we associate requirements on the development of the system – these should be reflected in the assurance case
  - Documented experience is invaluable for both the regulator and the developer
  - I believe there is a structure that is applicable to almost all products – especially at the top-level of the argument – so doing this within an application domain should be even easier
  - Templates are a partial counter to the view that assurance cases suffer from confirmation bias

McSCert

Assurance Case Template

Arguments are not explicit

- If you look again at the example assurance case a few slides earlier, you can see that there could be an argument provided by the structure of the graph – but it is not given
- GSN uses strategy boxes, but the strategies are quite superficial and are more accurately representative of goal decomposition - without an argument as to why the composition of the sub-goals results in the goal
- We need to pay more attention to how to structure valid arguments and how to detect specious arguments

McSCert

Assurance Case Template

Argument over all hazards mitigated

- The decomposition of the goal very often follows a strategy of "argue over all hazards are mitigated"
  - It is sometimes augmented by a process claim as in the example a few slides back
  - This may work only if you accept the view that the only important issue is one of safety/security
  - I strongly believe we need to be concerned with dependability - NAS Report on Software for Dependable Systems defines dependability as:
    - "A system is dependable when it can be depended on to produce the consequences for which it was designed, and no adverse effects, in its intended environment."

McSCert

Assurance Case Template

- I am going to leave out discussion on two of the most important topics – at least for now
  - Evidence
  - Confidence

Assurance Case Template Top Level Claim(s)

- System is acceptably dependable within its operating context and application domain
  - System requirements are adequately correct and include all safety & security constraints as well as all operator requirements including safe and secure HMI
  - System implementation complies adequately with its requirements and has not added any unmitigated hazards
  - System is robust with respect to (reasonably anticipated) change and is maintainable over its projected lifetime - changes will not degrade dependability, safety and security of the system
  - System maintains safe behaviour in the presence of hardware malfunctions

Need an argument that convinces us that if the 4 sub-claims are true, then the top-level claim must be true

Assurance Case Template Top Level Claim(s)

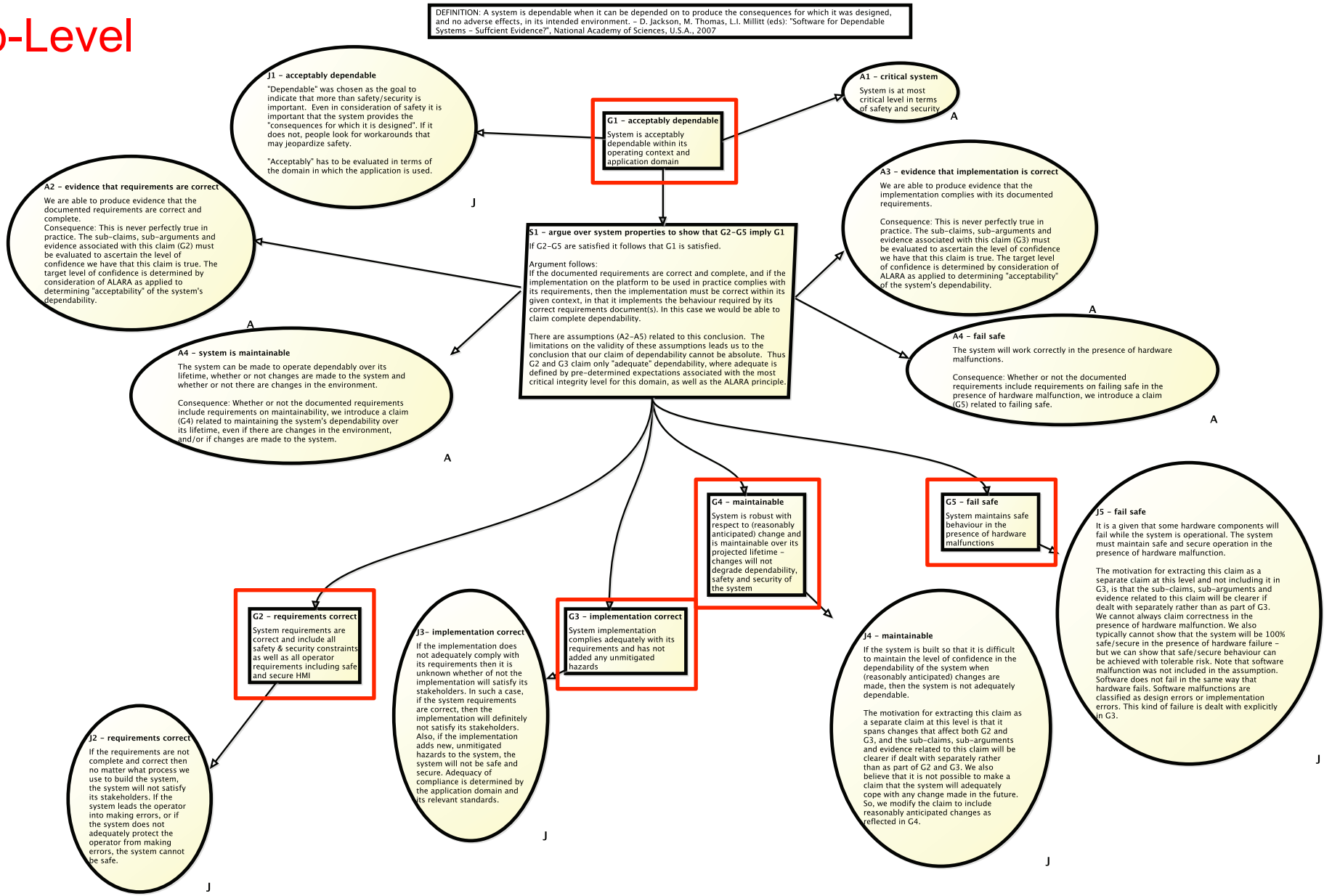
- System is acceptably dependable within its operating context and application domain
  - System requirements are adequately correct and include all safety & security constraints as well as all operator requirements including safe and secure HMI
  - System implementation complies adequately with its requirements and has not added any unmitigated hazards
  - System is robust with respect to (reasonably anticipated) change and is maintainable over its projected lifetime - changes will not degrade dependability, safety and security of the system
  - System maintains safe behaviour in the presence of hardware malfunctions

If you examine the claim and sub-claims you see that we cannot make these claims absolute. One of the most important aspects of our case is to build ***confidence*** in the validity of the case. Confidence is also tied up with ***evidence*** – different types of evidence engender different levels of confidence

Example of an Assurance Case Template

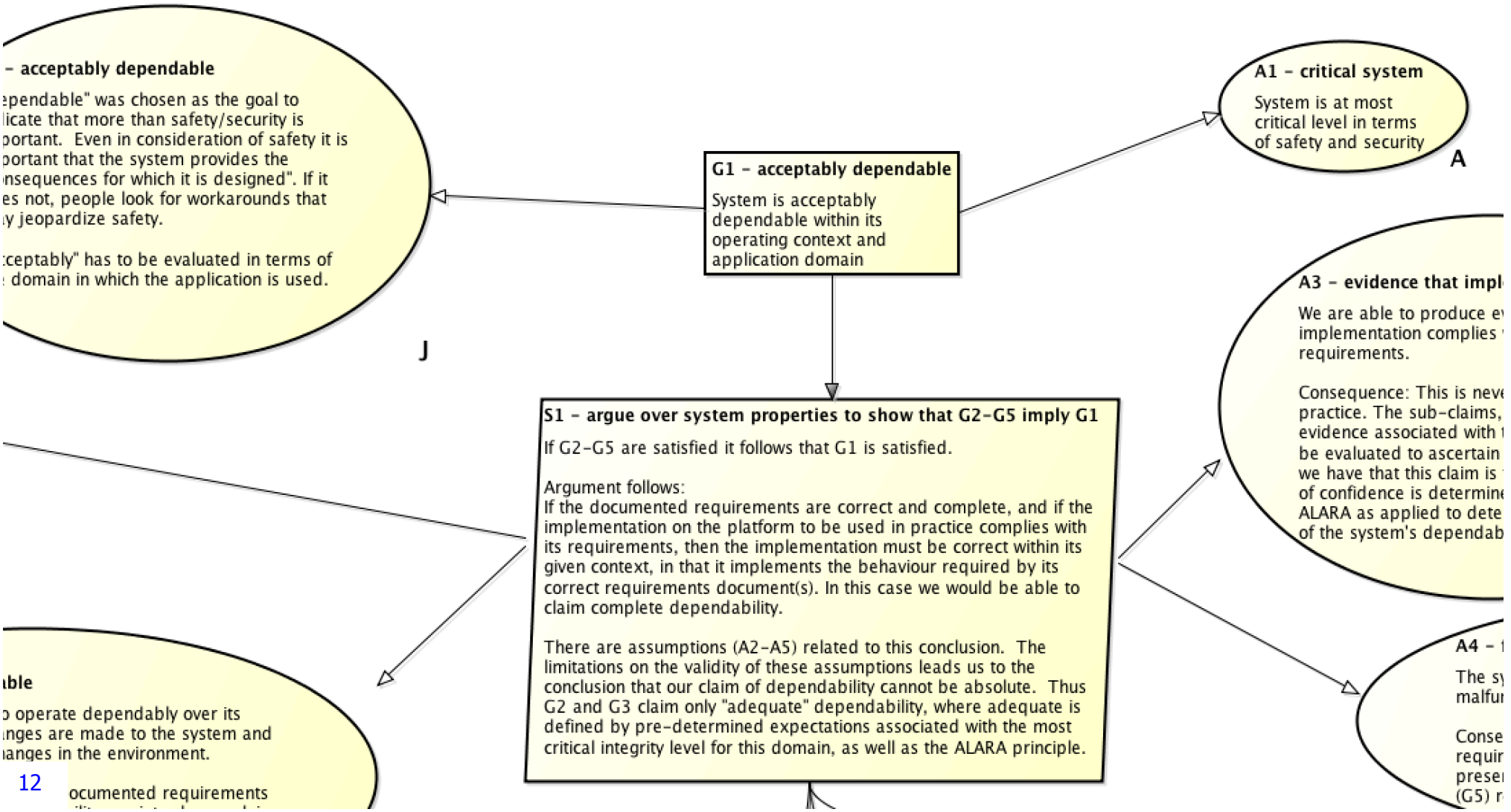
McSCert

Top-Level

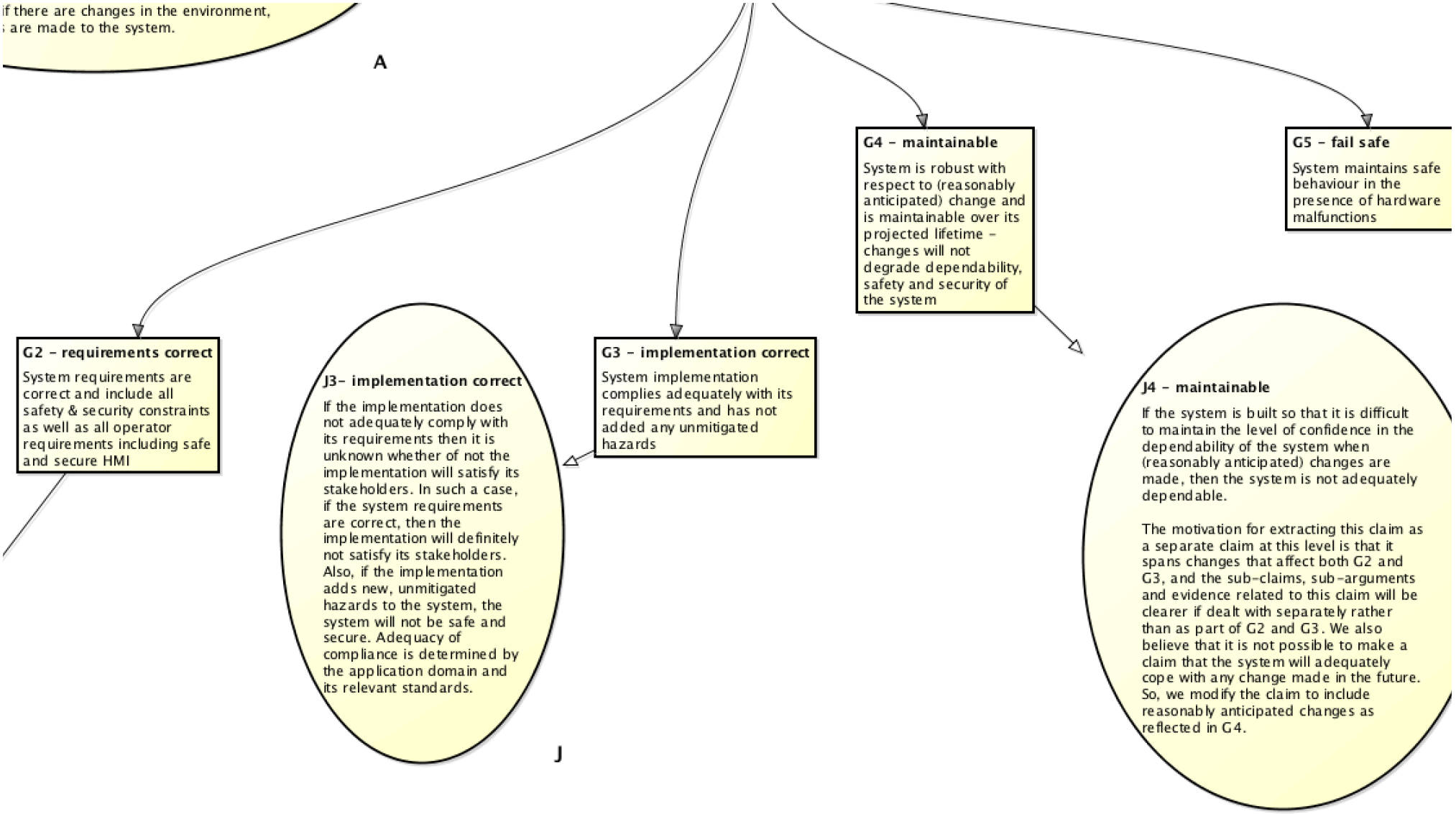


Top Claim

DEFINITION: A system is dependable when it can be depended on to produce the consequences for which it was designed, and no adverse effects, in its intended environment. – D. Jackson, M. Thomas, L.I. Millitt (eds): "Software for Dependable Systems – Suffcient Evidence?", National Academy of Sciences, U.S.A., 2007



Top Claim Decomposition



McSCert

Completing the Template

- The previous slides show just the top-level of the assurance case
  - It does demonstrate why a template can be constructed and why it is useful
- We then have to continue goal decomposition, recording the arguments as we proceed
- We do this until we get to where we can introduce evidence that supports the argument
  - Very different doing this after the system is built compared with before or while it is being built
  - Much greater risk of confirmation bias when it is done afterwards

McSCert

Example

- Expand on G3 - System implementation complies adequately with its requirements and has not added any unmitigated hazards
  - There is a design that documents behaviour of hardware and software components in enough detail to determine that the design includes all the behaviour in the requirements, it is adequately equivalent in behaviour to that in the requirements, and does not introduce any unmitigated hazards
  - The implementation of both hardware and software includes all the behaviour in the design, it is adequately equivalent in behaviour to that in the design, and does not introduce any unmitigated hazards

NOTE: We have now influenced our process by developing this template – we have hardware design and software design phases, and some requirements on those phases

Now we need justification as to why we decomposed the claim into these two sub-claims, and why if the sub-claims are true, the higher claim is true. We also need to document relevant assumptions etc. At this stage we can start seeing some possibilities for evidence – verifications/ reviews/testing

Evidence & Confidence

- Eventually we will get down to a stage where we can introduce relevant evidence to support a sub-claim
  - For example, to show compliance of the implementation with its design we could produce test reports
  - The confidence we have in the validity of the test report may be influenced by the test coverage used – (I am assuming we do not stop testing until all tests are successful)
  - The confidence we have in the validity of the sub-claim may be higher if we have additional evidence – mathematical verification, reports from source code static analysis, etc

The template may actually have placeholders for different types of evidence and may also include different strategies for claim/goal decomposition. The template is also likely to get to stages where the developers know how to include strategies not foreseen when the template was created

Benefits of an Assurance Case Template

- They possibly can do what standards do now, but do it better in that expectations and guidance could be much clearer
- Much easier for regulators/certifiers to develop expertise in evaluating assurance cases
- They reflect reality and good practice – design safety/ security in right from the start
- They will influence both the process used and evaluation of the product – they should also influence evaluation of the people involved
- They can help (partially) avoid confirmation bias
- With good tools and better knowledge of how to "test" arguments, they could provide more guidance than current process standards, and yet be flexible as methods/technology advance

McSCert

Assurance Cases for Software Qualification

- We have reverse engineered CSA N290.14 – a Canadian Nuclear Standard for software qualification of COTS devices/software into an assurance case
- What is the underlying major problem in evaluating safety of a system with EDDs?
  - The fact that safety is a global property of the system!
  - John Rushby: “although it is generally infeasible at present to guarantee critical [safety] properties by compositional (or ‘modular’) methods, it is a good research topic to investigate why this is so, and how we might extend the boundaries of what is feasible in this area”

McSCert

Assurance Cases for Software Qualification

- So, what do we need if we want to evaluate 'plug and play' EDDs in a system?
  - One way out is to have an assurance case for the system
  - With 'good' structure in the assurance case it may be possible to replace one EDD by another, because:
    - We can examine relevant assumptions
    - We can examine the argument(s) related to that EDD and see if the argument would need to be changed for the new EDD
    - We can examine the evidence related to that EDD and see if we can provide equivalent evidence – if not, how similar?

Thank You!

wassyng@mcmaster.ca