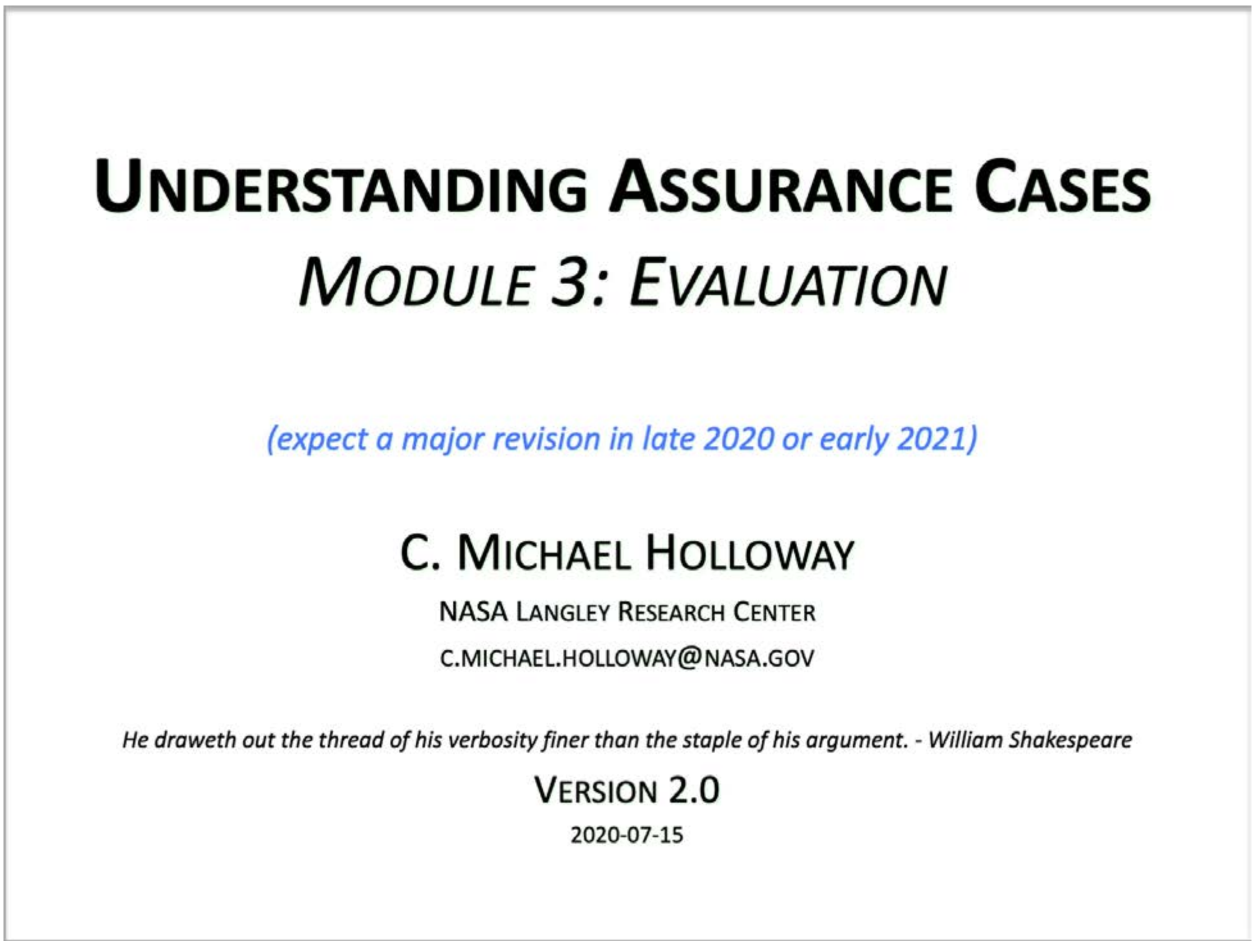
Understanding Assurance Cases:
An Educational Presentation in Five Parts

Module 3: Evaluation

C. Michael Holloway
`c.michael.holloway@nasa.gov`

Senior Research Computer Engineer
Safety-Critical Avionics Systems Branch
NASA Langley Research Center, Hampton, Virginia, U.S.A.



This material was originally created in 2015-16, as part of the Explicate '78 project. The project was supported in substantial part by the Assurance Case Applicability to Digital Systems task under the reimbursable interagency agreement with the Federal Aviation Administration for Design, Verification, and Validation of Advanced Digital Airborne Systems Technology (IAI-1073 Annex 2 for NASA; DTFACT-10-X0008, Modification 0004 for the FAA). The original presentations were delivered to a selected group of FAA civil servants and NASA Langley personnel. The audio was recorded and partial transcripts (containing only the words spoken by the presenter, Mr. Holloway) produced. The intent from the beginning was to collect the material into a form that could be made available publicly. The text adheres closely to the original transcript, except where changes have been made to the original presentation since it was first given, as part of work for for NASA IA-303333/FAA IA NO 692M15-19-T-00029 Annex 1/TO 1. The full collection consists of six documents (including this one), which are available electronically through `https://shemesh.larc.nasa.gov/arg/uac.html`.

Welcome to the third module in an educational series about Understanding Assurance Cases. [Significant changes will be made to this module by mid 2021.]

In this module, we will examine the ***Evaluation*** of assurance cases. If you have not already completed Modules 1 and 2 (Foundation and Application respectively), please stop reading this document, and complete both Foundation and Application before continuing[1].

In evaluating an assurance case one hopes the occasion will not arise to say of the writer of the case what one Shakespeare character said of another in *Love's Labour's Lost:* "He draweth out the thread of his verbosity finer than the staple of his argument."

[Shakespeare, William. *Love's Labour's Lost*, act v, scene i, lines 1750-51.]

As with all the modules, feel free to interrupt me at *any* point if you have a burning question. I reserve the right to defer the answer to later on that's appropriate, but otherwise I'll do my best to answer it.

In today's module, there will be a few times when I'll ask you to do a bit of work on your own --- nothing substantial or time-consuming, but I hope it'll help improve your understanding of the material.

[Question to participants: Does anyone have any questions or comments that you want to make now, before we proceed further?]

Let's list our learning objectives.

By the time we're finished today, I hope that you'll be able to do at least these four things.

One, identify *positive* properties that an assurance case *should* have.

Two, identify *negative* properties that an assurance case *should not* have.

Three, you should also be able to enumerate steps for evaluating an assurance case.

Four, I expect you to be able suggest potential corrections for selected deficiencies.

As I'm sure you realize, when we're done with this module, you're not going to be an expert in evaluating assurance cases (unless you're one already), but you should be fairly well acquainted with much of what's involved in evaluating them.

[Question to participants: Any questions about these learning objectives?]

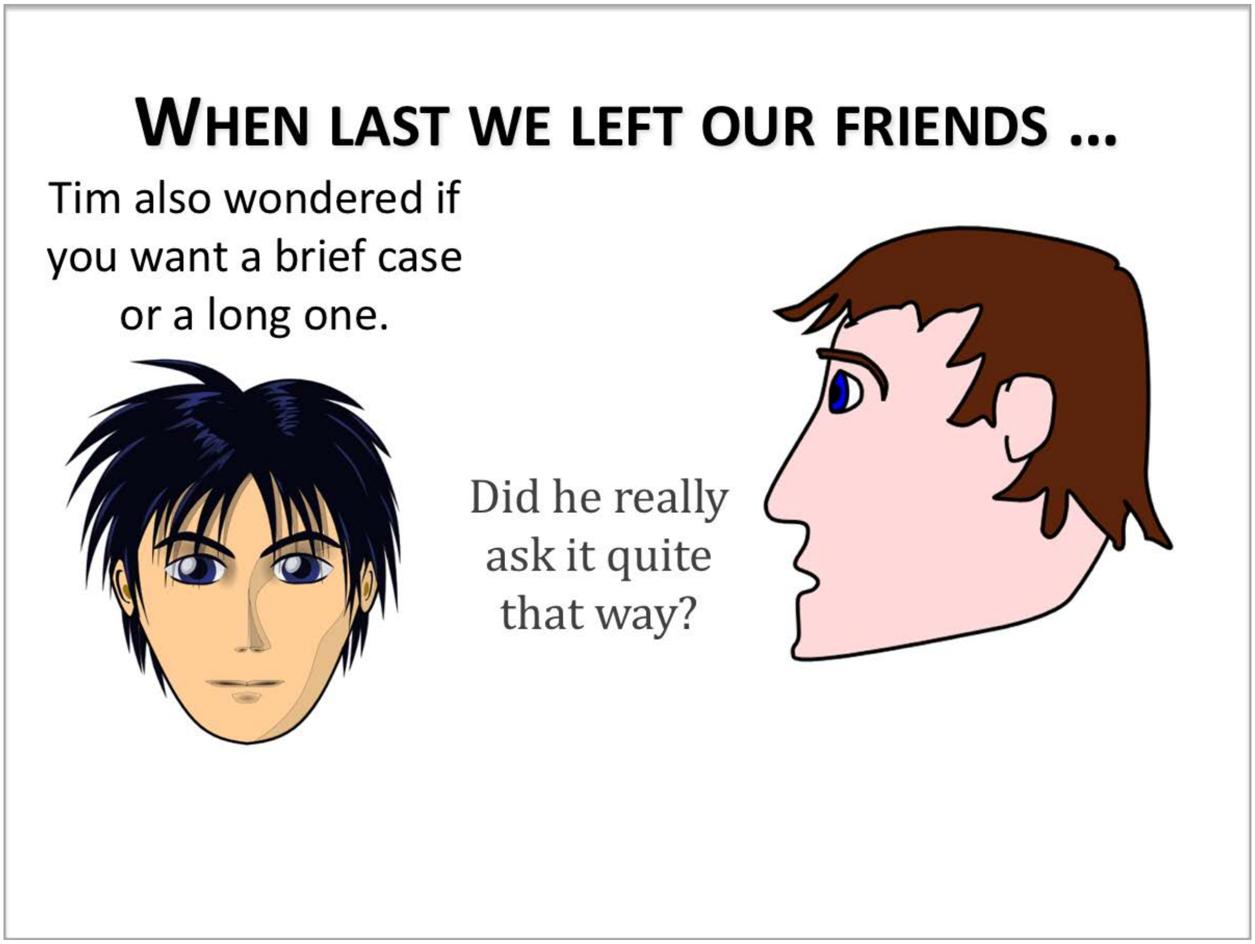
---

[1] Just in case someone does not follow the suggestion, and thus misses the preliminary information first expounded in Module 1 and repeated verbatim in Module 2, here is that information in simplified form: Within the assurance case community, intramural debates abound about a variety of topics we will discuss. Except in rare instances the existence of these debates is intentionally ignored or mentioned only briefly in this material. (See Module 1 or 2 for an explanation of why). Also, all images you see were either created by me (Michael Holloway) or are in the public domain via CC0 1.0 Universal.

When last we left our friends Jon (the teenager on the left), his dad Mike (the fellow on the right), and Tim (the unseen fellow who may or may not drive Jon to a game) Jon had told his dad that …

“Tim also wondered if you want a brief case or a long one.”

We left Jon’s dad smiling, but we know pick up the conversation a few seconds later.



Mike asks Jon, “Did he really ask it quite that way?”

“Well, no, not exactly,” says Jon, “He’s not quite as funny as I am.”

“So, what did he really want to know?” asks Mike.

“He wanted more details about what you’re expecting,” replies Jon.

“That’s simple,” says Jon’s dad.

“I want a cogent argument.”

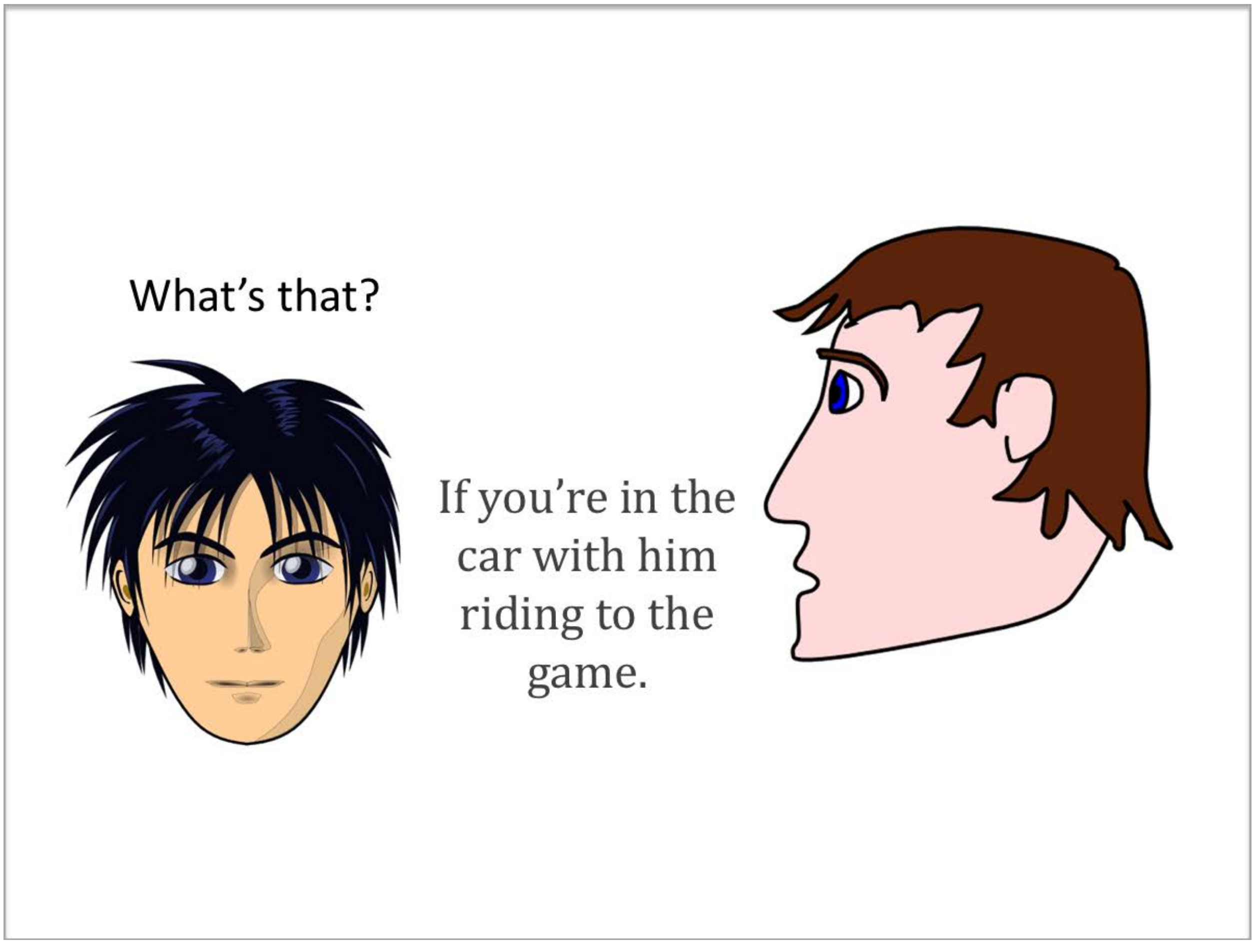
Jon is not thrilled by that answer, and exclaims,

“Simple? … How will Tim know what you’ll think is cogent?”

“Well … there’s one surefire way he’ll know …”

“What’s that?” asks Jon.

“If you’re in the car with him riding to the game.”

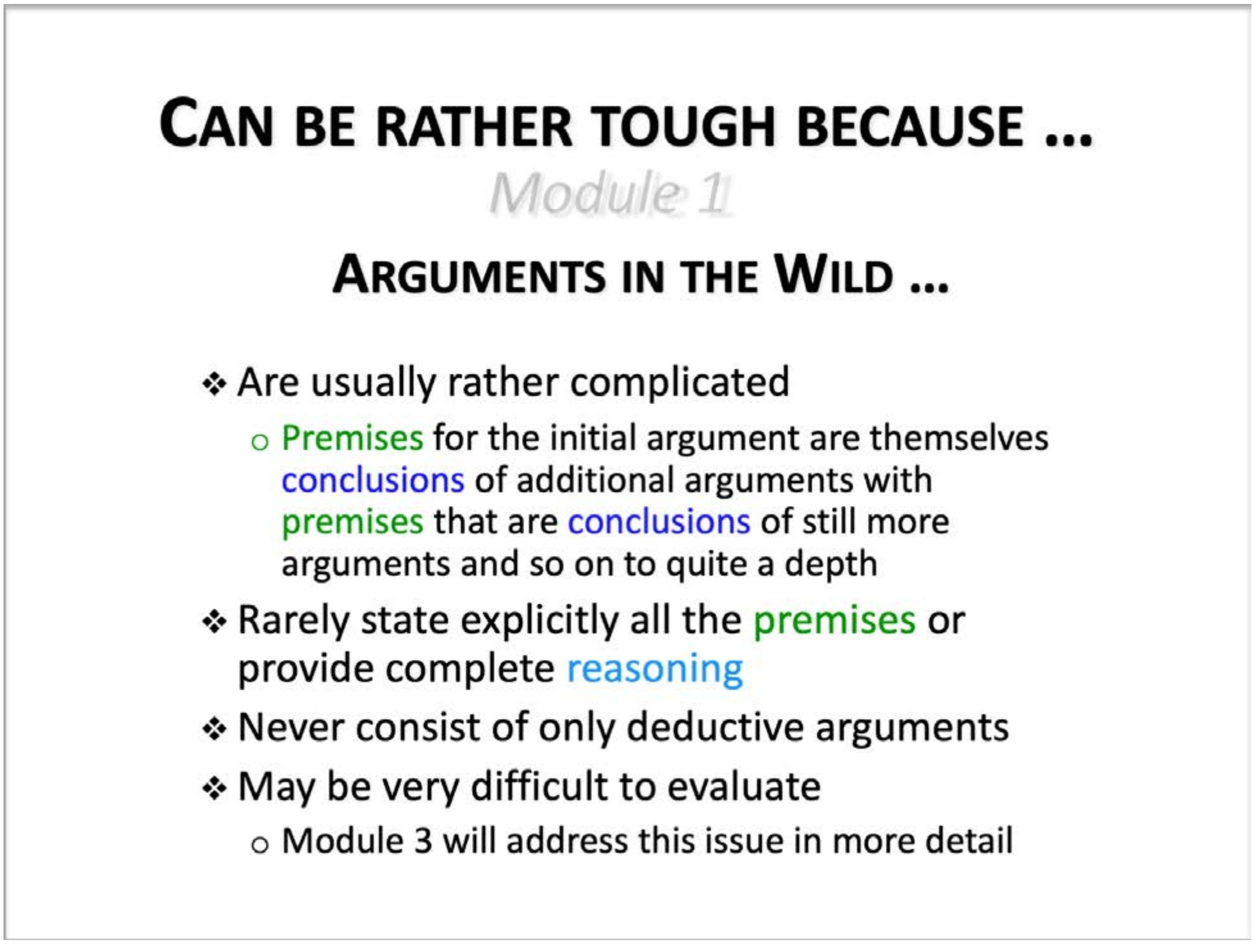


With a sigh, Jon replies, “Cute Dad … but that’s a bit late to find out, don’t ya think?”

“Yeah, sorry … there really isn’t a simple answer. Deciding if a case is good enough can be rather tough.”

Mike is spot on: evaluating an assurance case can be rather tough, whether you’re a writer evaluating your own case, an auditor evaluating someone else’s case, or just an inquisitive learner wondering about the matter.

It can be rather tough for a variety of reasons, beginning with some of the observations we made in Module 1 concerning the nature of arguments in the wild.



Recall then that we said that real arguments are usually rather complicated. We noted in particular, the premises for the initial argument are themselves usually conclusions of additional arguments with premises that are conclusions of still more arguments and so on to quite a depth.

In any real assurance case, the *premises* for the top level *conclusion* will almost certainly not be obvious truths, but rather statements that will need to be supported by argument themselves. Eventually the assurance case should stop with sub-arguments with *premises* whose truth can be agreed upon by all relevant parties; such premises are sometimes called evidence, 'though, as I've mentioned, I am not particularly fond of that term.

Second, real arguments rarely state explicitly all of the premises or provide complete reasoning. This should be less true of assurance case arguments than is generally true of generic arguments in the wild, but deciding whether it's true is one of the evaluation activities, and it is not necessarily an easy one.

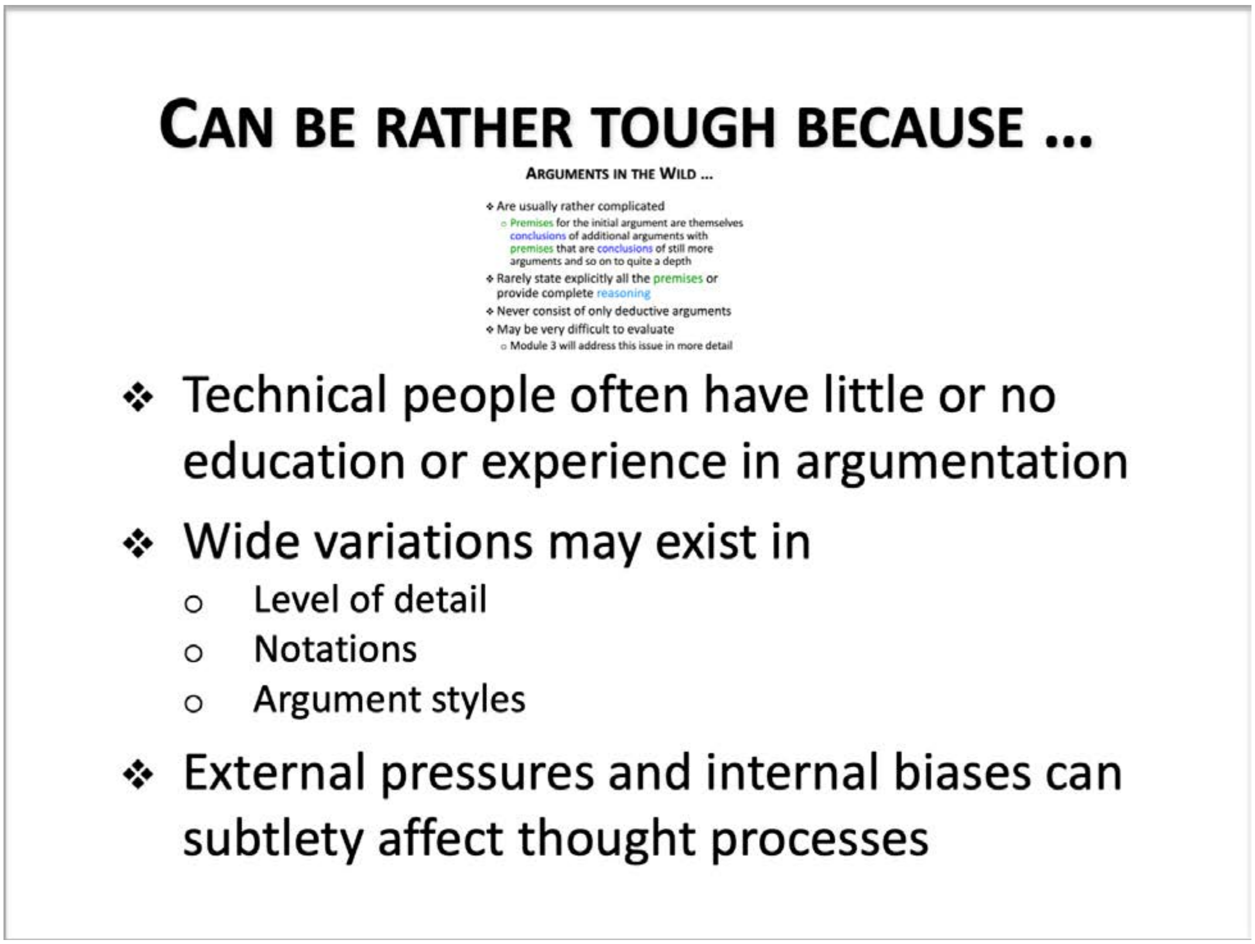
Third, real arguments, both in the generic wild, and in the assurance case context, almost never consist of only deductive arguments.

You'll recall from Module 1 (or from prior knowledge) that deductive arguments are ones in which true premises and valid reasoning *guarantee* the truth of the conclusion.

Inductive arguments, on the other hand, do not provide guarantees, only increases in confidence. An inductive argument with true premises and strong reasoning should improve our confidence in the truth of the conclusion, but ought not provide us with certainty.

We talked a bit in Module 1 about the controversy that exists within the assurance case community over whether there may be advantages to be gained from making deductive as many arguments as possible; or perhaps by using a normalized structure that isolates inductive arguments into specific parts of the overall argument. That controversy is currently an academic one, because everyone, even the most zealous formalist, recognizes that the current state of the practice involves mostly inductive arguments.

These three facts aren't the only things that can make it rather tough to evaluate an assurance case.



Other toughness inducing-aspects include the things you see here. Technical people often have little or no education or experience in argumentation. This lack of knowledge and practiced ability can lead to poorly written assurance cases, and perhaps to an inability by auditors to recognize them as such.

You may recall from Module 2 the poor quality of some safety cases was identified in several accident inquiries including Ladbroke Grove and Nimrod, and has been identified by the FDA as a problem they are experiencing as they use assurance cases in infusion pump approvals.

Evaluating assurance cases can be tough also because in practice cases may vary widely in the level of detail provided (some cases may be really just argument sketches, while others may delve deeply into the tiniest details of a system).

They may also differ widely in the notations used, ranging (as we saw in Module 1) from unstructured prose to highly structured, but not necessarily easy to understand, graphical notations.

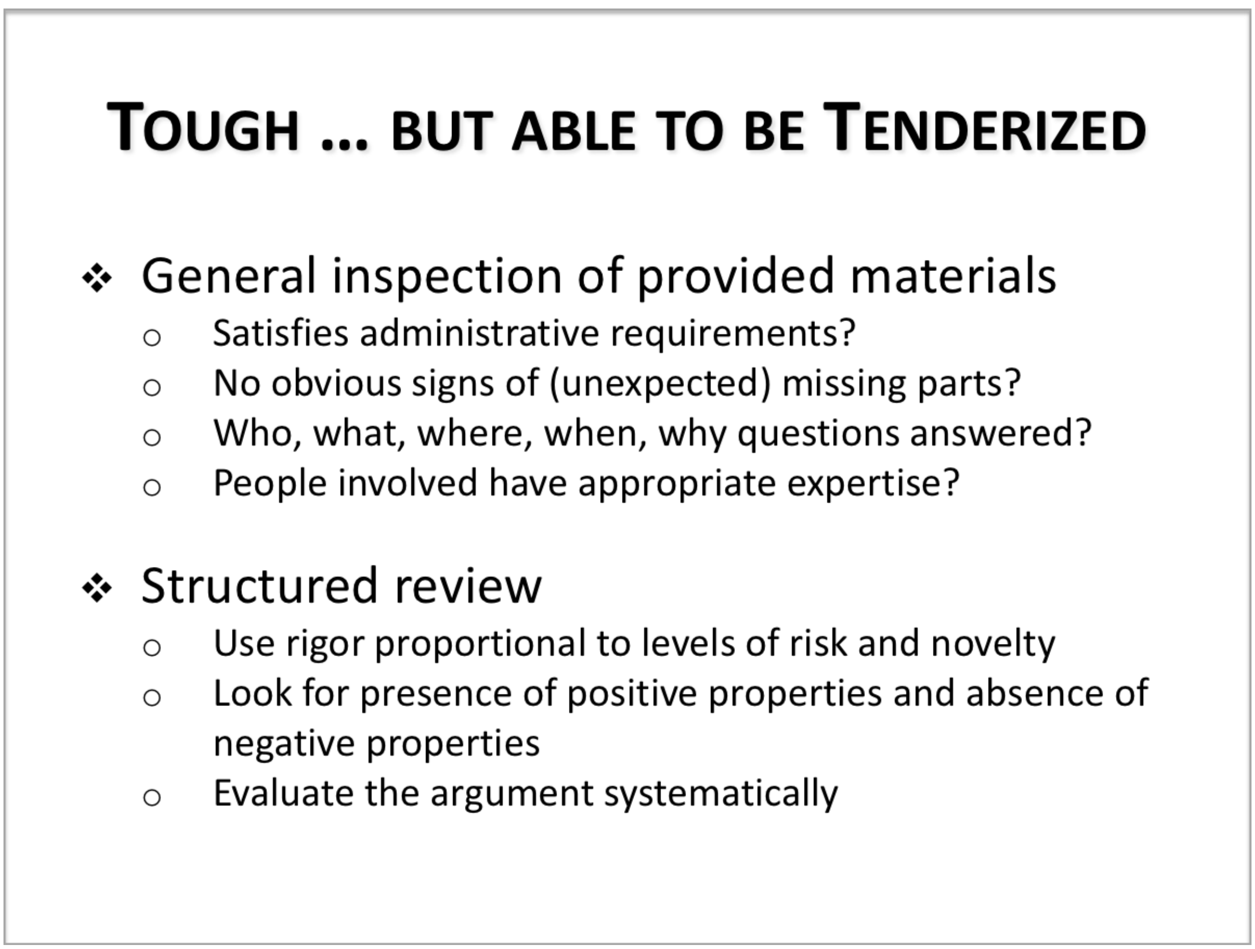
If you're asked to evaluate an assurance case in a notation you don't already know, you may find it quite hard to distinguish between problems in the assurance case itself and problems in your own understanding of the notation.

There can also be wide variations in argument styles, which can make consistent evaluation hard.

Finally, evaluating an assurance case can be made tough by external pressures and internal biases that can affect your thought processes, even if you try to block out the effects. We'll talk some more about these things a bit later on. All these things, and probably others we've not discussed, make evaluating assurance cases tough.

[Question to participants: Before I talk a bit about how this toughness may be tenderized, does anyone have a question they'd like to ask now?]

Evaluating assurances is tough, but it can be tenderized in some very helpful ways.



First, there are various steps that can be taken by way of the general inspection of provided materials.

Before starting evaluation of the assurance argument, you should look everything over to see if (first) it satisfies administrative requirements. For example, if the argument is required to be expressed in a particular notation or style, is it?

You should be sure that there are no obvious signs of (unexpected) missing parts. Does the argument have a top level conclusion, for example.

As we discussed in Module 2, answers to the who, what, where, when, why questions are important. Does the case make clear who wrote it, what its scope is, and what assurance target is applicable, for example?

Finally, by way of general inspection, does the information you have available show that the people involved in designing the system or service, have appropriate expertise?

If an assurance case that you've been asked to evaluate does not pass even a general inspection, there is no good reason to attempt a more extensive, structured review.

We'll talk about the structured review in much more detail shortly, but here on the slide are three important aspects of it.

First, the rigor of the review should likely be tailored to the levels of risk and of novelty in the system or service for which the assurance case has been developed. Generally, the greater the risk the more rigorous the review should be, and the greater the novelty of the system, the more rigorous the review should be.

Second, you should be continually looking for presence of positive properties and absence of negative properties (both of which we'll talk about a bit more shortly).

This looking for properties will be going on while you evaluate the argument systematically. We'll go through a procedure for this systematic review shortly.

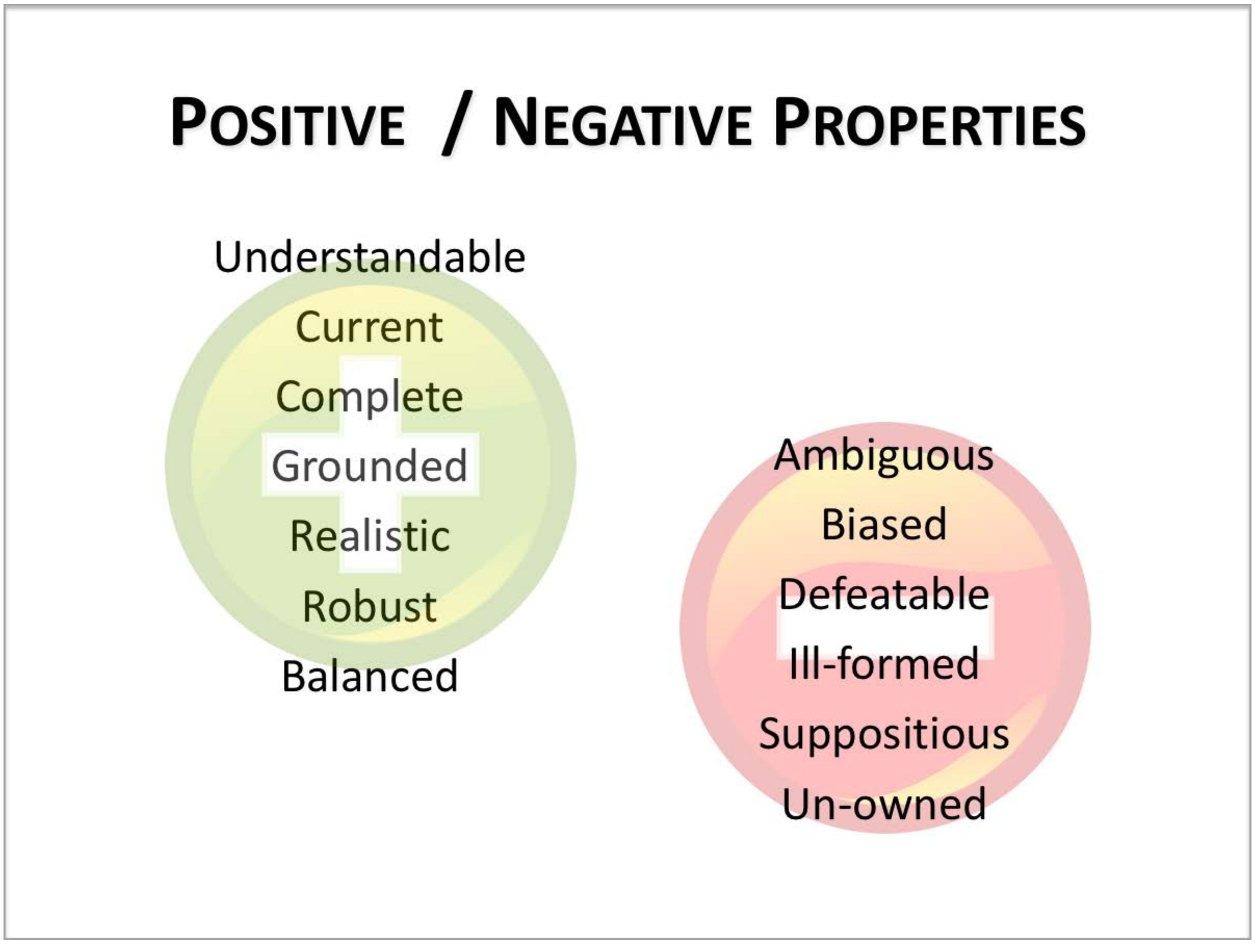
Now, I want to enumerate briefly some positive properties and some negative properties that an assurance case may possess.

The next slide lists seven positive properties and six negative properties; the meaning of some of these is probably self-evident; while the meaning of some others … not so much.

Understandable is pretty self-evident: the assurance case needs to provide enough information, in a clear way, so that everyone who will use it knows what it means, and to what it applies.

Current means that the case accurately represents the current status of the system or service in all relevant aspects.

Complete is a relative term, which depends on the life cycle stage(s) covered by the case, but relative to that stage, the assurance case should cover in an appropriate way all aspects of the system or service.



By *Grounded* I am referring back to the concept we introduced in Module 1, namely that the argument structure terminates in premises whose 'truth' can be agreed by all relevant parties.

Realistic means that the case identifies its assumptions and that these assumptions correspond well to what will happen (or is happening) in the actual world.

Robust refers to an assurance case incorporating good engineering practice and known sound safety principles.

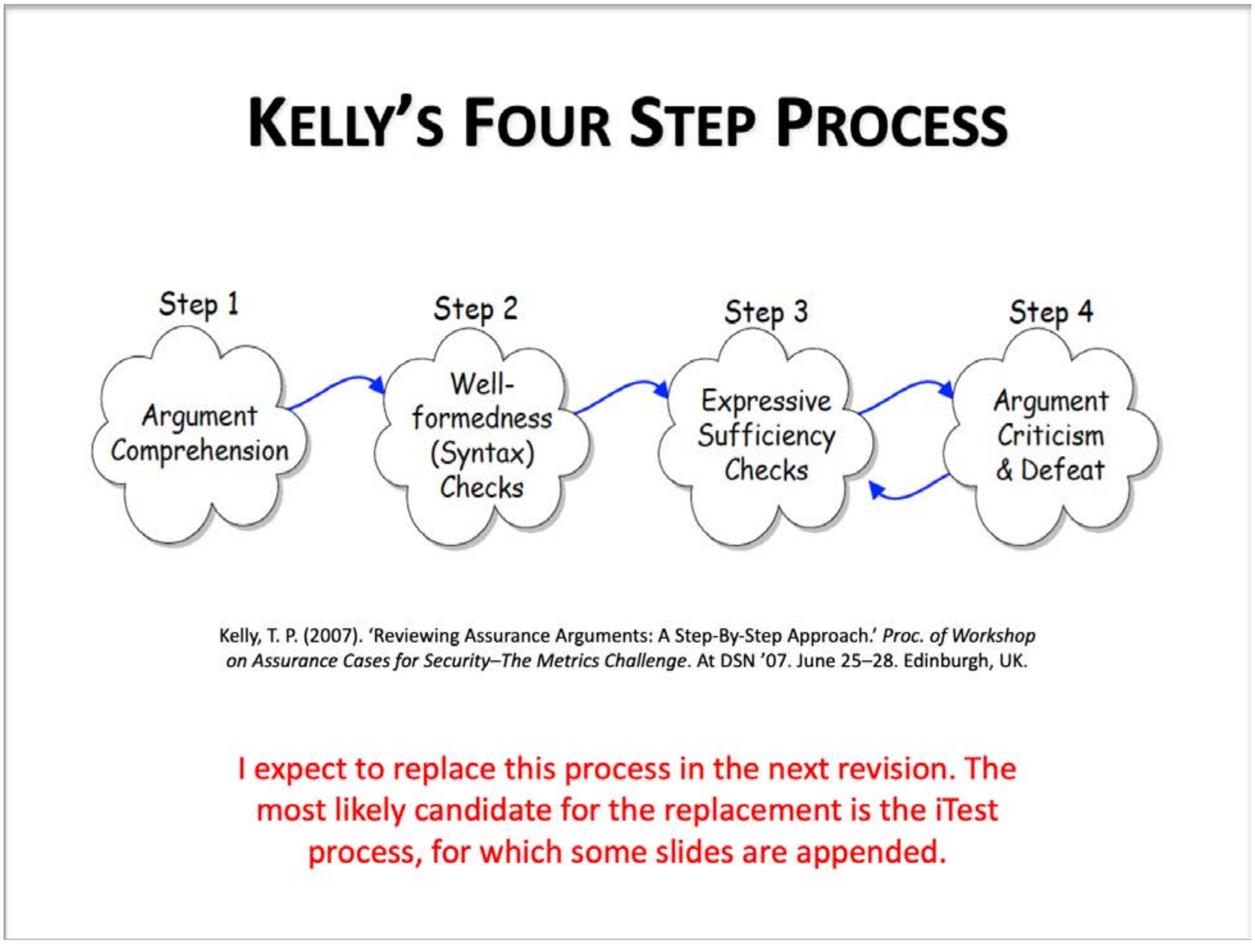
Finally, the positive property *Balanced* refers to the assurance case identifying not only the strengths of the system or service but also its known weaknesses.

Those are seven positive properties that a good assurance case should possess.

You see also 6 negative properties that a good assurance case should not possess, but which a bad one probably will: ambiguous, biased, defeatable, ill-formed, suppositious, and un-owned. I'm going to defer talking about what these mean until after we've gone through a process for systematic evaluation of the assurance case argument.

[Question to participants: Before we proceed, does anyone have a question?]

I'm going to present one particular way to undertake a systematic evaluation. There are many other ways, 'though all of them will necessarily include similar sorts of things as the process that I'll show you. [By mid 2021 this approach will be replaced.]



The four step process you see here was developed by Tim Kelly at the University of York, and published in a DSN-affiliated workshop proceedings in 2007.

[Kelly, T. P. (2007). "Reviewing Assurance Arguments: A Step-By-Step Approach." *Proc. of Workshop on Assurance Cases for Security---The Metrics Challenge*. At DSN '07, June 25-28. Edinburgh.]

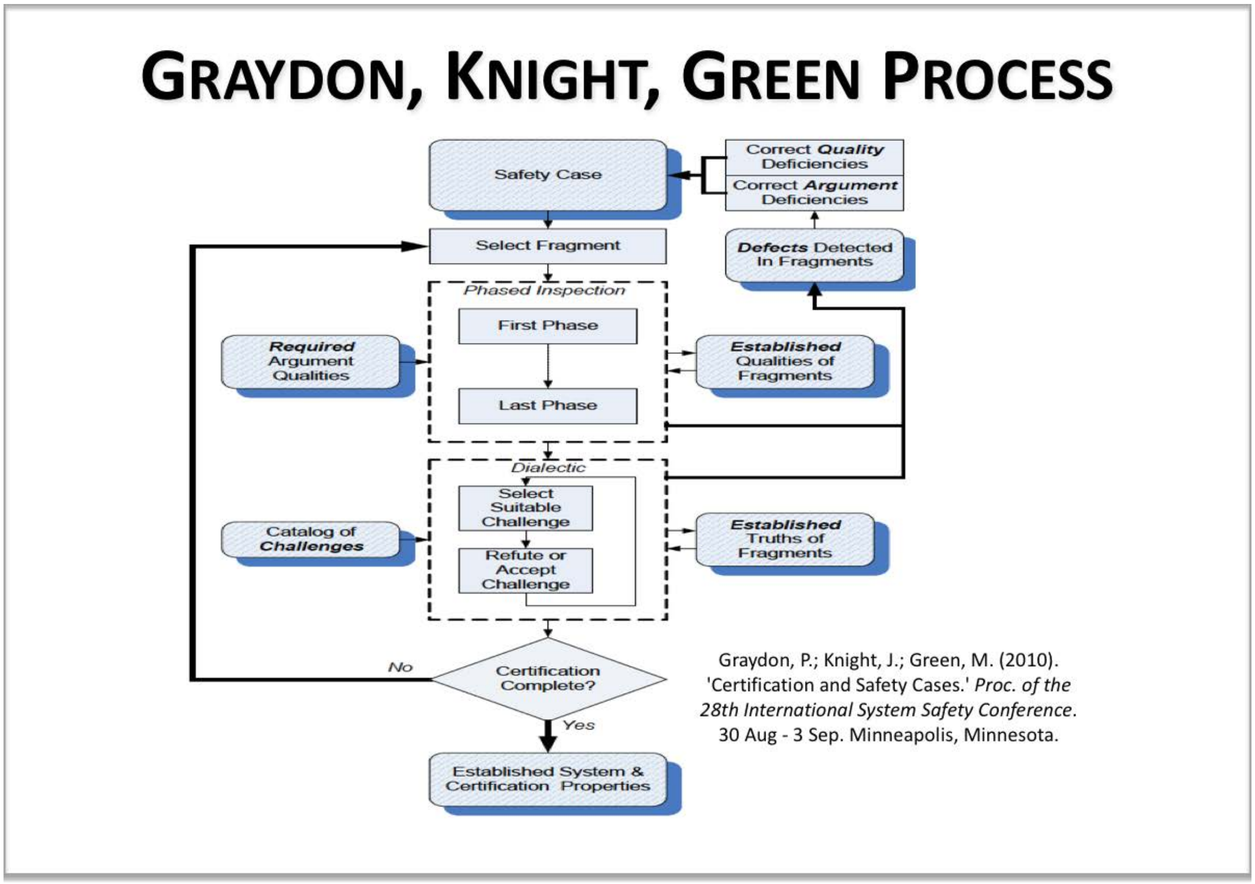
Argument evaluation starts with argument comprehension then proceeds to checking for well-formedness, followed by checking for expressive sufficiency, and concluding with argument criticism (and possible) defeat, which may lead to changes in the argument necessitating repeating step 3, followed by step 4.

We'll look at each of these steps in more detail shortly.

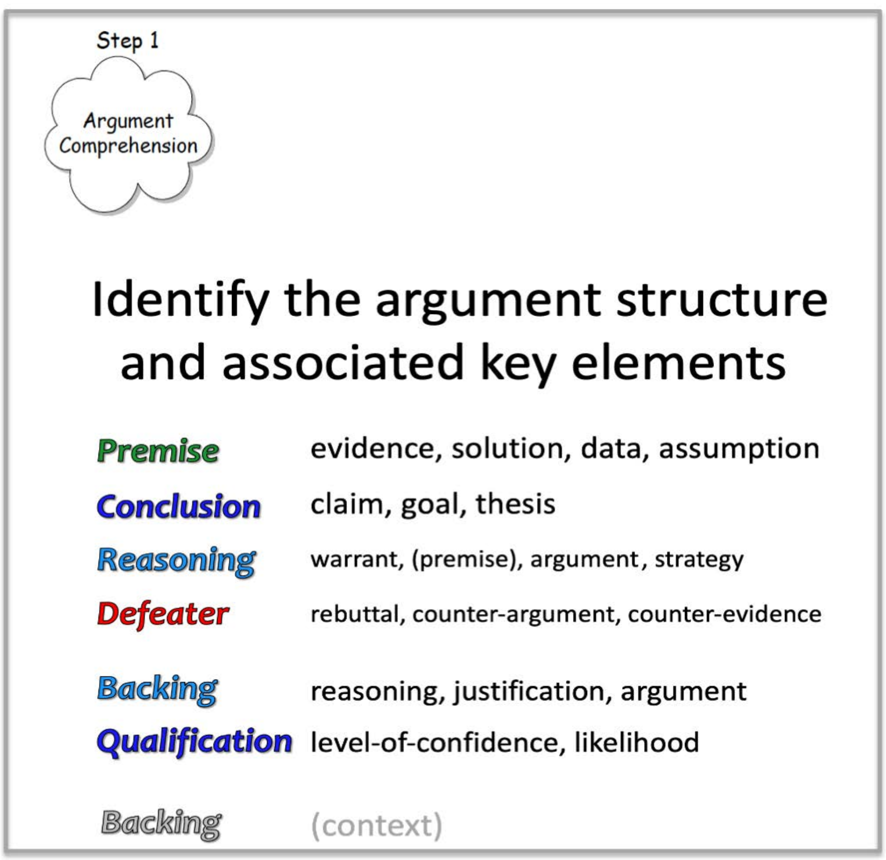
Before doing so, I want to also mention another argument evaluation process from which I've borrowed some parts.

Patrick Graydon, John Knight, and Mitchell Green, who were all at the University of Virginia at the time, published this process at the International System Safety Conference in 2010.

I'm not going to go into any detail, but just want to note that I'll be incorporating some of the ideas from the GKG approach into my elaboration of the Four Step Process.



So let's look at each of the four steps in Kelly's process in turn. Step 1 is understanding the argument.

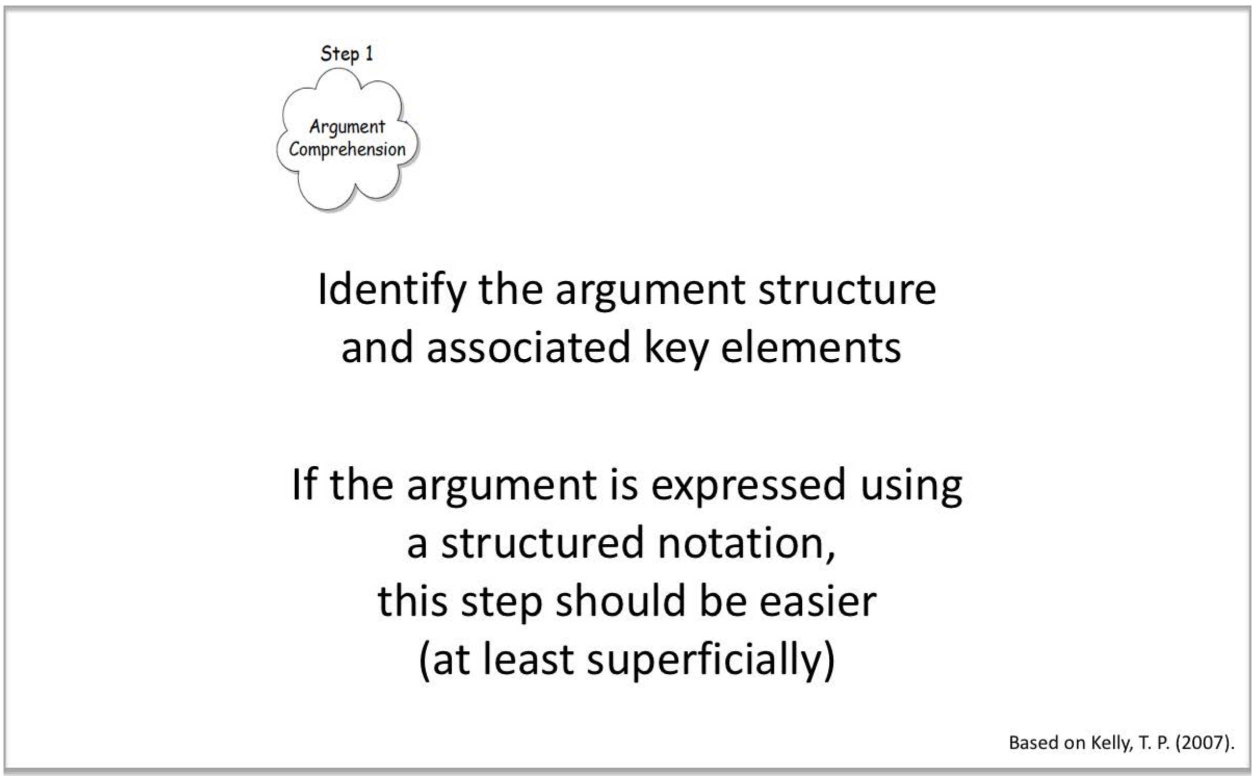


Or to be more precise, identifying the argument structure and associated key elements.

As I hope you recall from Module 1, those key elements include *premise*, *conclusion*, *reasoning*, (all three of which are necessarily present) and the other elements (which may or may not be present) *qualification*, *defeater, backing,* and *binding*.

As we did in Module 1, other common names for these concepts are listed, too.

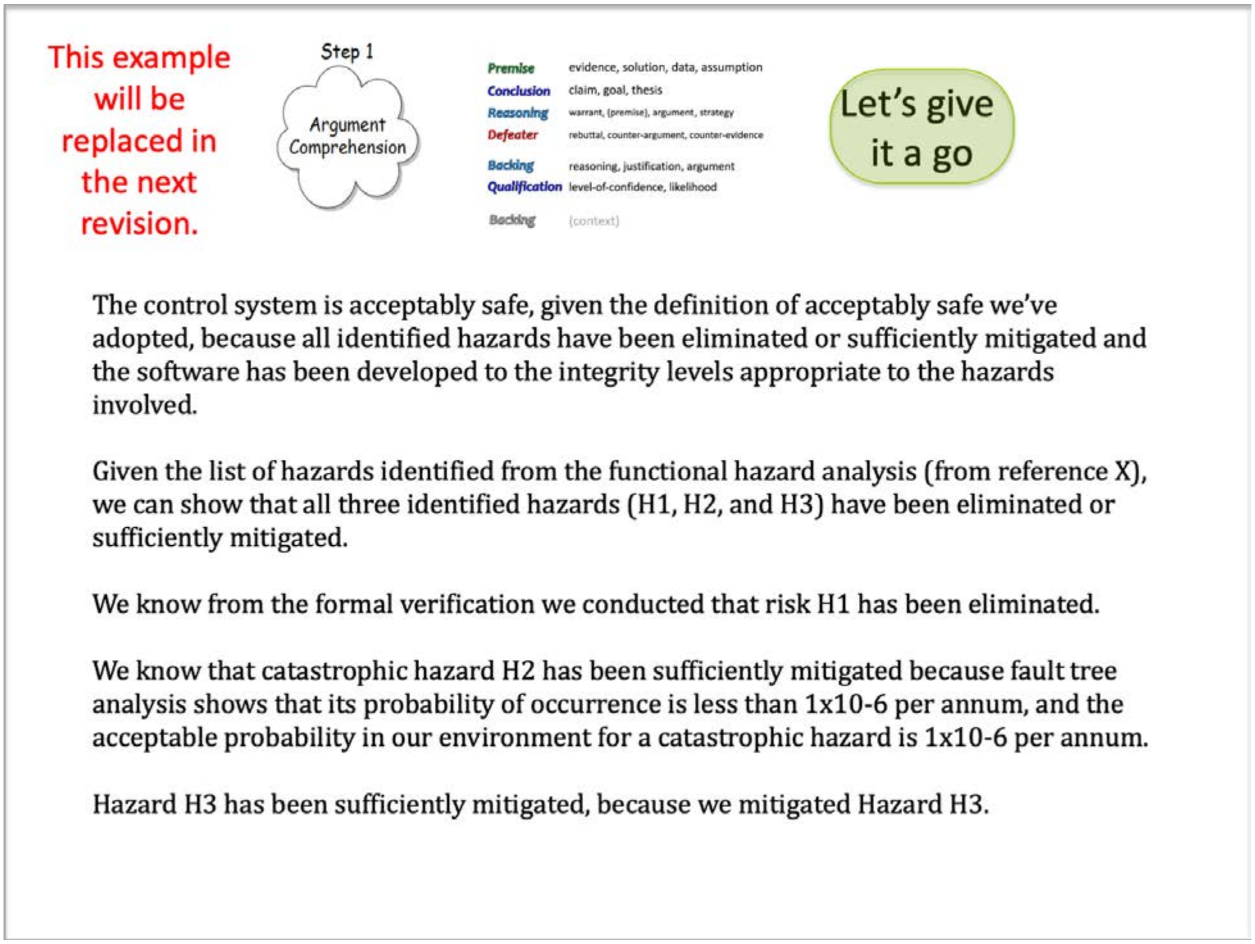
Kelly notes in his paper that if the argument is expressed using a structured notation, this step should be easier. I've added the qualification "at least superficially", because using a structured notation doesn't really guarantee that a comprehensible argument will be created. Certainly, if the argument is expressed in an entirely unstructured way using regular prose, re-representing it in some structured way, (which doesn't have to be graphical) can be a wise thing to do at this stage.



Now I'm going to show you a short assurance case, written in natural language. The example is based on the example I used in the 2008 notations paper mentioned in Module 1 [Holloway, C. M. 2008. "Safety Case Notations: Alternatives for the Non-Graphically Inclined?" IET 3nd International Conference on System Safety. 21-23 October 2008, Birmingham, UK. Available at `http://hdl.handle.net/2060/20080042416`.]

What you'll see is not identical to the case presented in the paper, but it is very similar.

Here it is. (In the planned revision, this example will be replaced.)



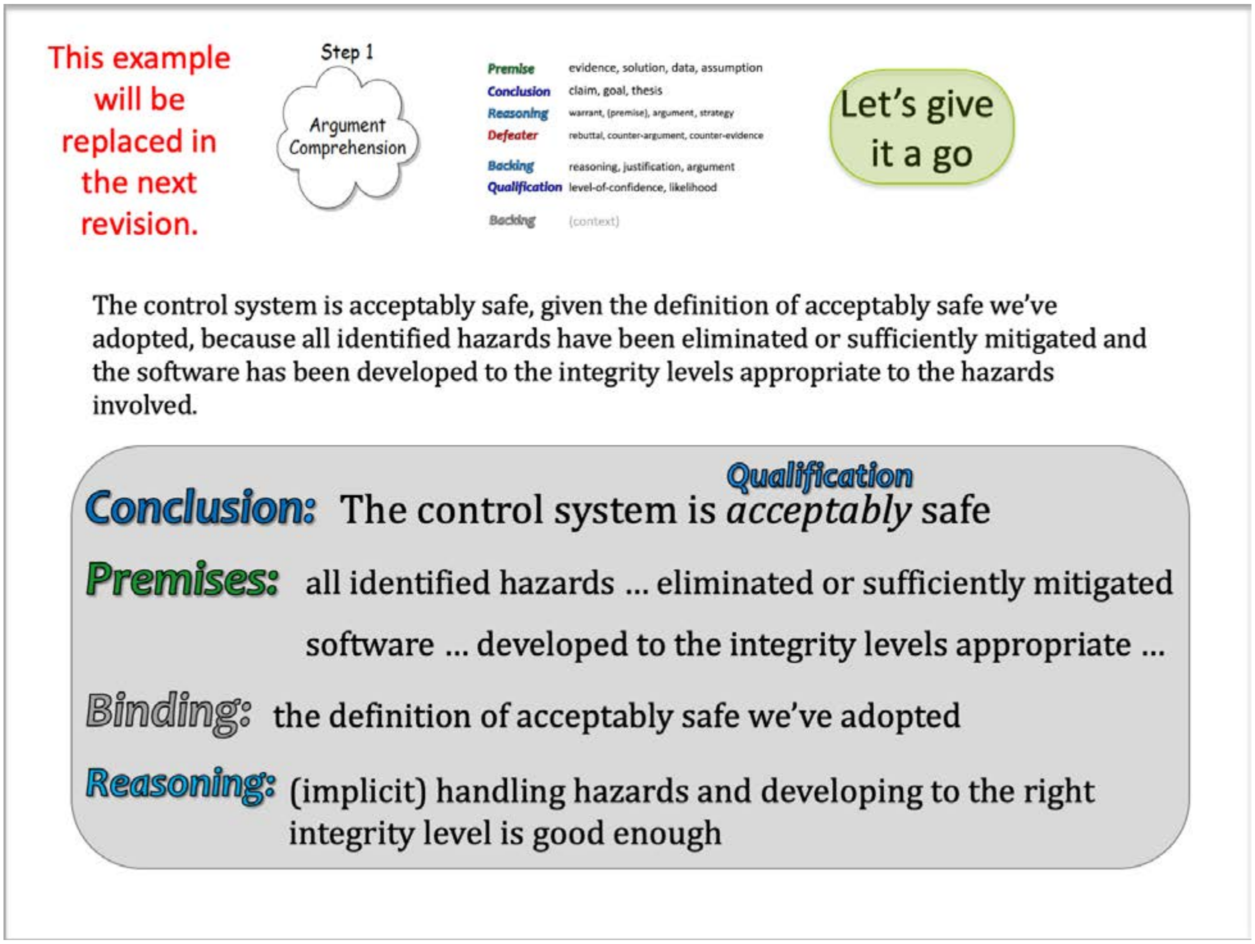
What I want you to do is to identify the top level conclusion and the premises and reasoning upon which it rests, along with any qualifications or bindings that are associated with them.

Don't try to do anything more than that. Remember that these top-level premises may well serve as conclusions for lower-level arguments. Do not worry about the lower-level arguments.

As two very big hints … you don't have to read very much of the text, and it is entirely possible that an important element may be implicit.

Please do not turn the page until you have attempted the exercise.

Here’s my answer.



The conclusion is “The control system is … safe” with the qualification of “acceptably”.

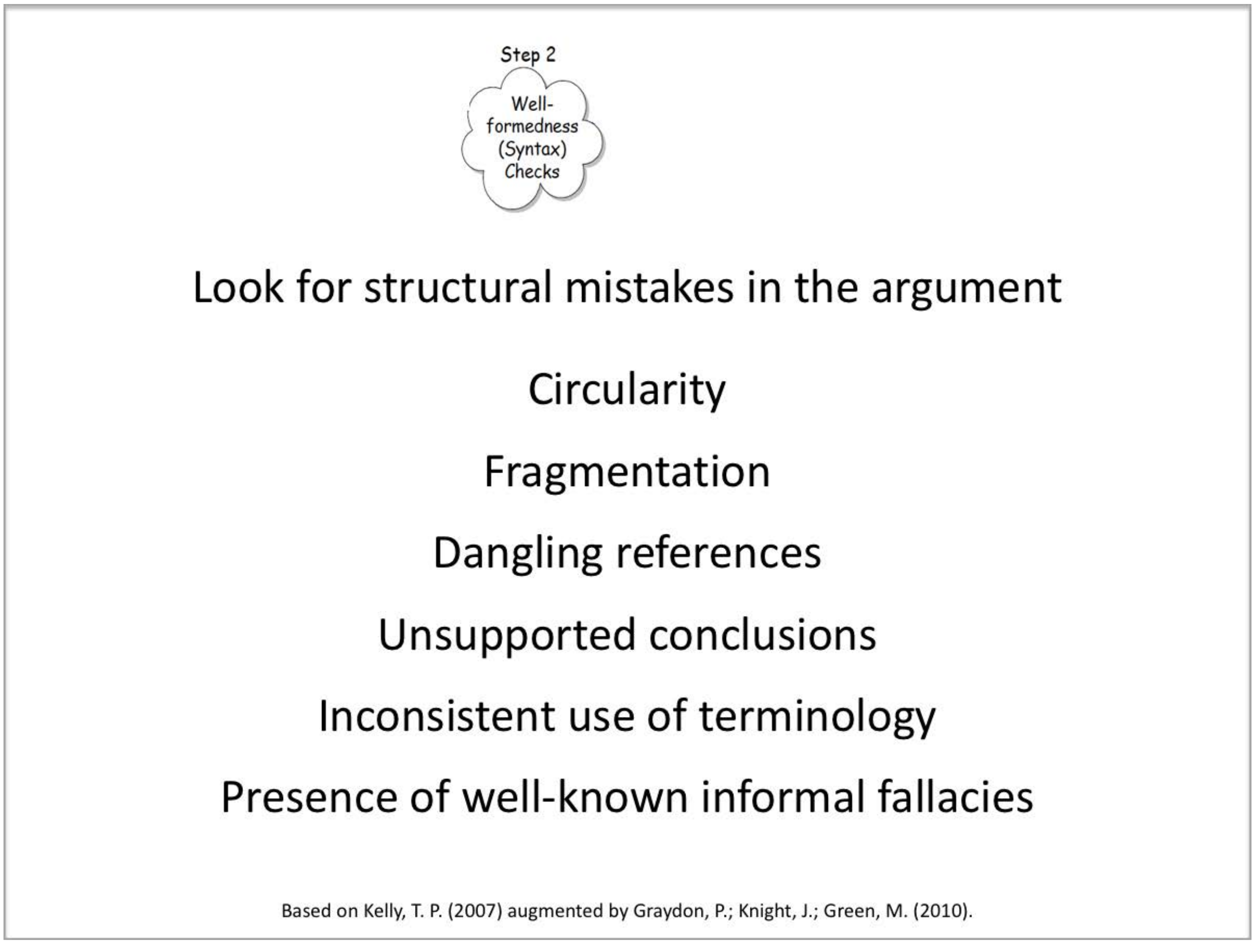
The two premises are “All identified hazards have been eliminated or sufficiently mitigated” and “The software has been developed to the integrity levels appropriate to the hazards involved.”

A definition of “acceptably safe” needs to identified in a binding, and the reasoning seems to be implicit, something along the lines of “handling hazards and developing to the right integrity level is good enough.”

That’s the argument comprehension step.

[Question to participants: Any questions?]

Step two is called “Well-formedness (Syntax) Checks.” It involves looking for structural mistakes in the argument.



Among the structural mistakes that might exist are six that you see listed here.

Circularity refers to an argument that has as a premise a statement that is equivalent to its conclusion. This could happen directly, ‘though it is more likely to happen indirectly, where the conclusion at (for example level `n`) in an argument reappears in some form as a premise in (for example) level `n+3`.

Fragmentation refers to arguments that are disconnected from the main argument.

A *dangling reference* is a reference in the argument to something that doesn’t exist (or at least isn’t present within the assurance case materials available to the evaluator).

Unsupported conclusions are conclusions for which no argument is given. In effect they are treated as premises (something about which everyone can agree), but their truth remains in doubt.

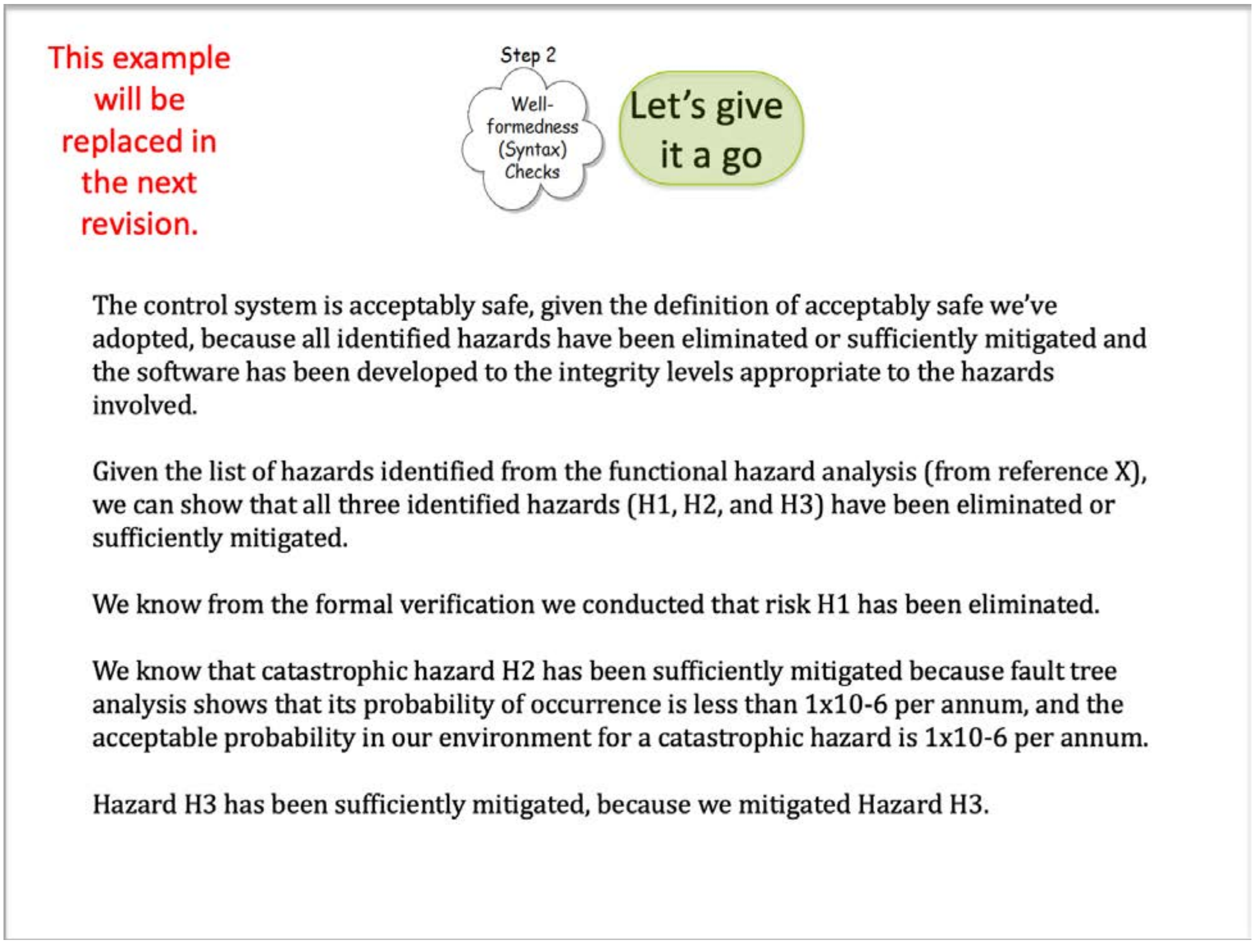
Inconsistent use of terminology is self-evident, I think.

Finally, the *presence of well-known informal fallacies* refers to using arguments that are known to be inadequate.

One example is known as the fallacious composition, which occurs when an argument claims that, because a property holds over the parts of a system or service, it therefore holds for the larger entity, without considering possible interactions between parts or external influences.

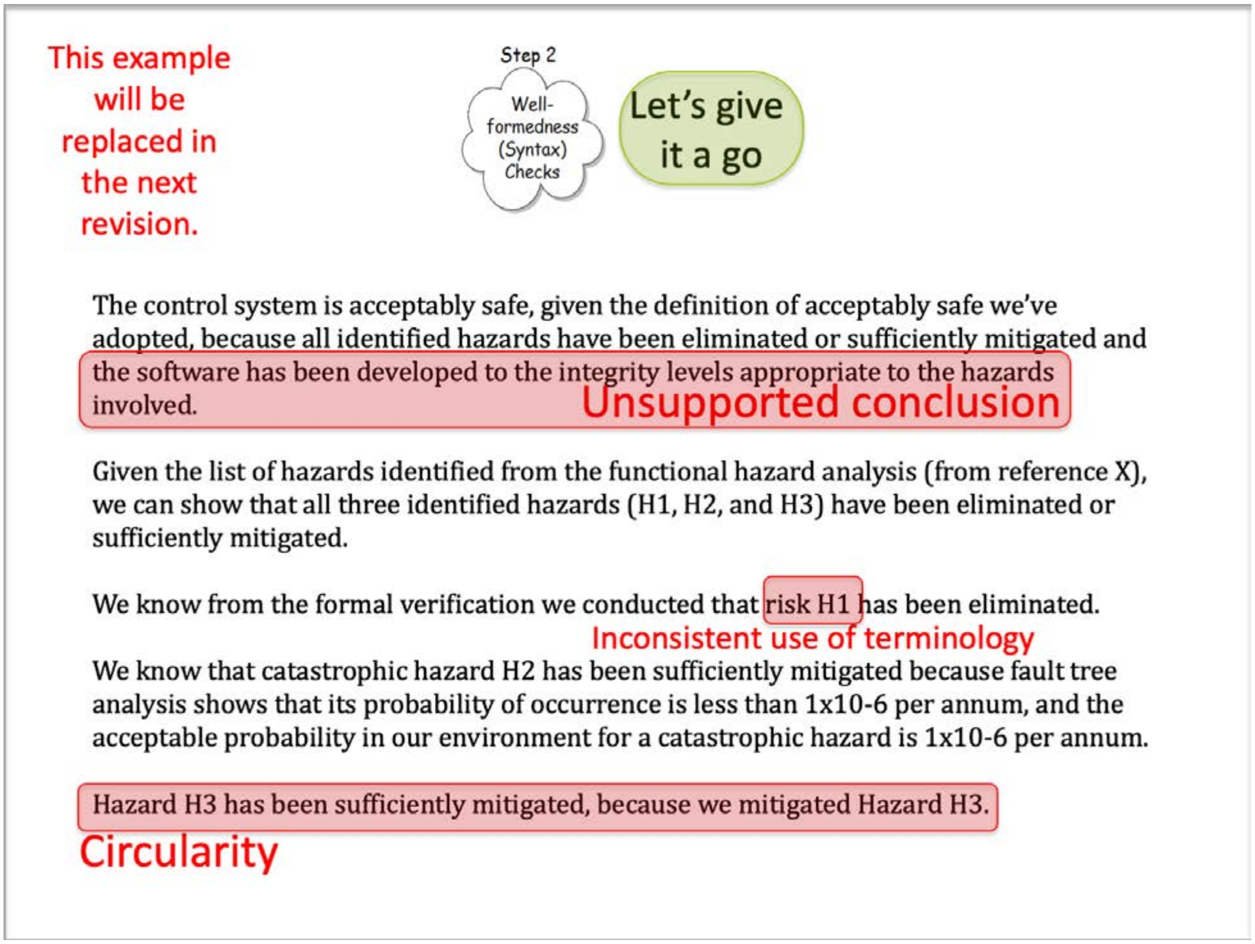
A prototypical example of fallacious composition within a safety case is an argument that claims that a whole system is safe solely because its subsystems A, B, and C are safe, while failing to consider the effect on safety of interactions among the subsystems.
Any questions before you get to try to identify some structural mistakes?

See if you can find some structural mistakes in the argument we just looked at for step 1. There are at least three.



Please do not turn the page until you have attempted the exercise.

Here's my answer.



One structural mistake is that the top-level premise concerning software being developed to appropriate integrity levels is not supported by any argument, hence it is an unsupported (lower-level) conclusion.

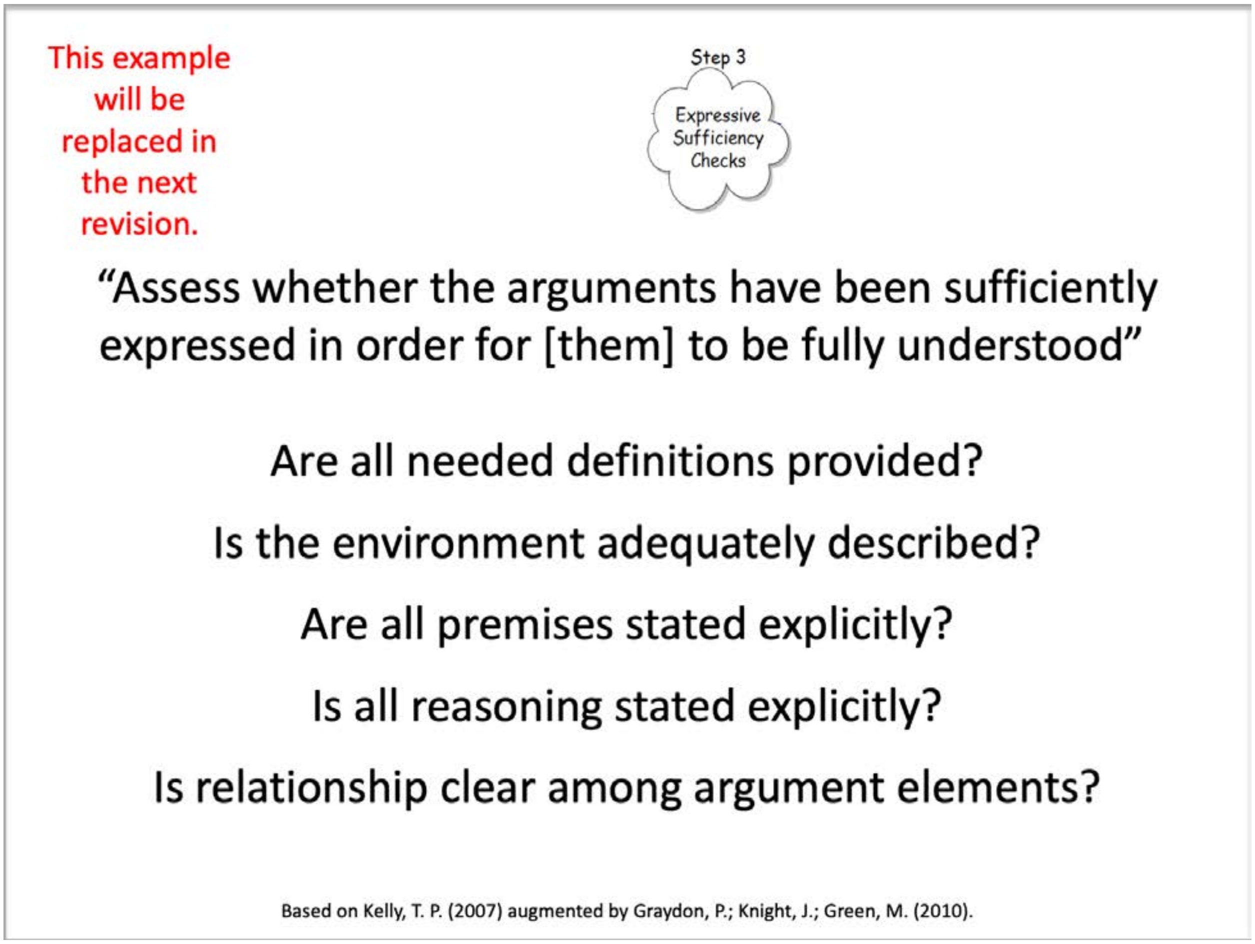
Another structural mistake is the use of the word 'risk' in relation to H1 in the third paragraph, which is inconsistent terminology as 'hazard' is used elsewhere.

Finally, the argument concerning hazard H3 is blatantly circular.

That's step 2.

[Note to participants: Any questions?]

Step 3 involves assessing whether the arguments have been sufficiently expressed in order for them to be fully understood.



Specifically, answering questions such as these listed on the slide.

Are all needed definitions provided? Word or phrases without definitions may be understood differently by different people. This problem is especially acute for some technical words, in which different domains have very different definitions. 'Verification' and 'validation' are perhaps the prototypical examples of such words. The agreed definitions of the two words within the computer science / systems engineering communities are almost exactly opposite from the agreed definitions within the controls theory community.

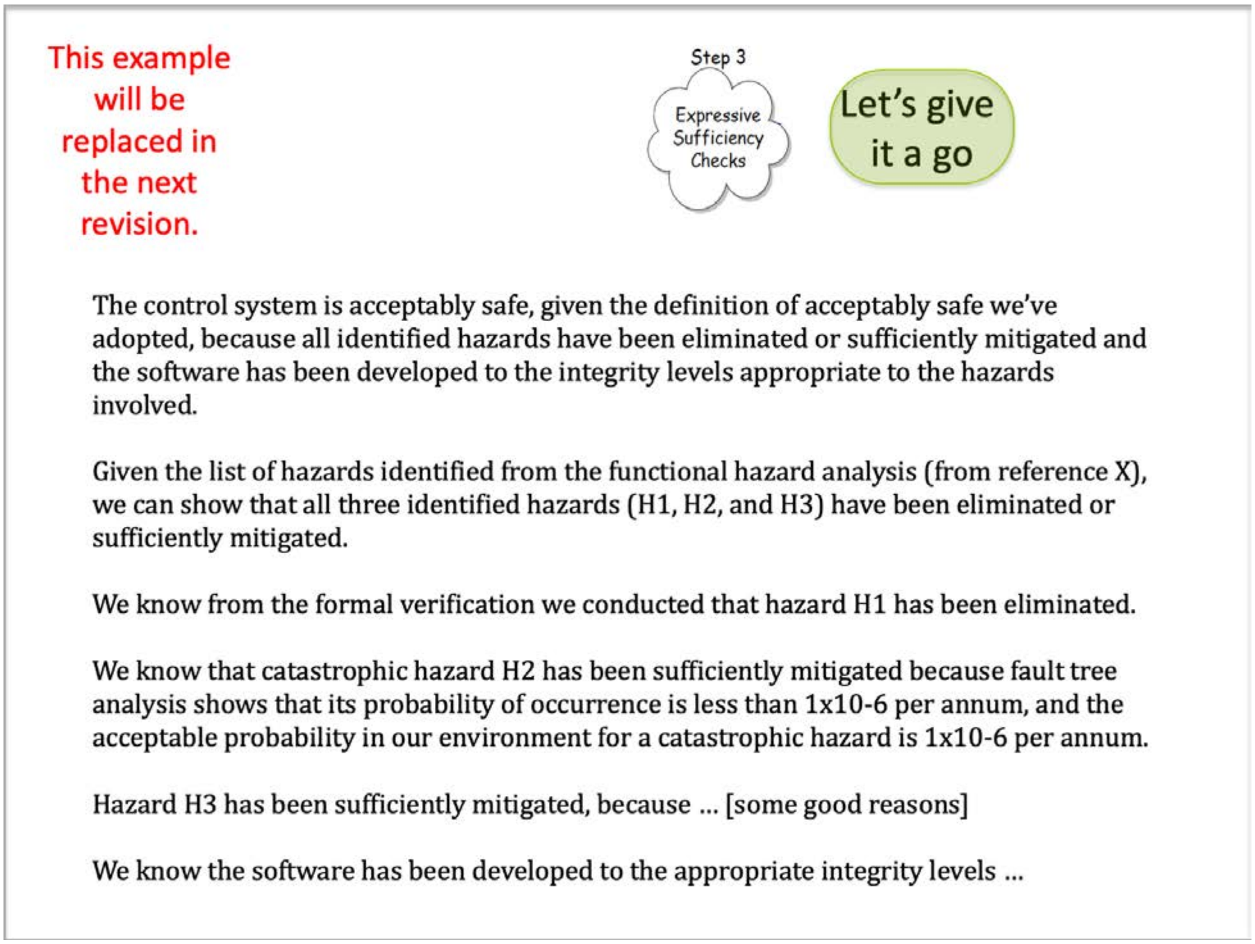
Is the environment adequately described? Failure to describe the environment in which a system or service is expected to operate can easily result in an assurance case that makes invalid assumptions about the operating environment.

Are all premises stated explicitly? As we noted in Module 1, implicit premises are a common occurrence in informal arguments.

Is all reasoning stated explicitly? As we also noted in Module 1, implicit reasoning is even more common than implicit premises. The implicitness of reasoning is especially acute in guidance documents developed without careful regard to assurance arguments. [See, for example, Holloway, C. Michael, & Graydon, P. J. *Explicate '78: Assurance Case Applicability to Digital Systems*. DOT/FAA/TC-17/67. January 2018. Available at https://www.faa.gov/aircraft/air_cert/design_approvals/air_software/media/TC-17-67.pdf].

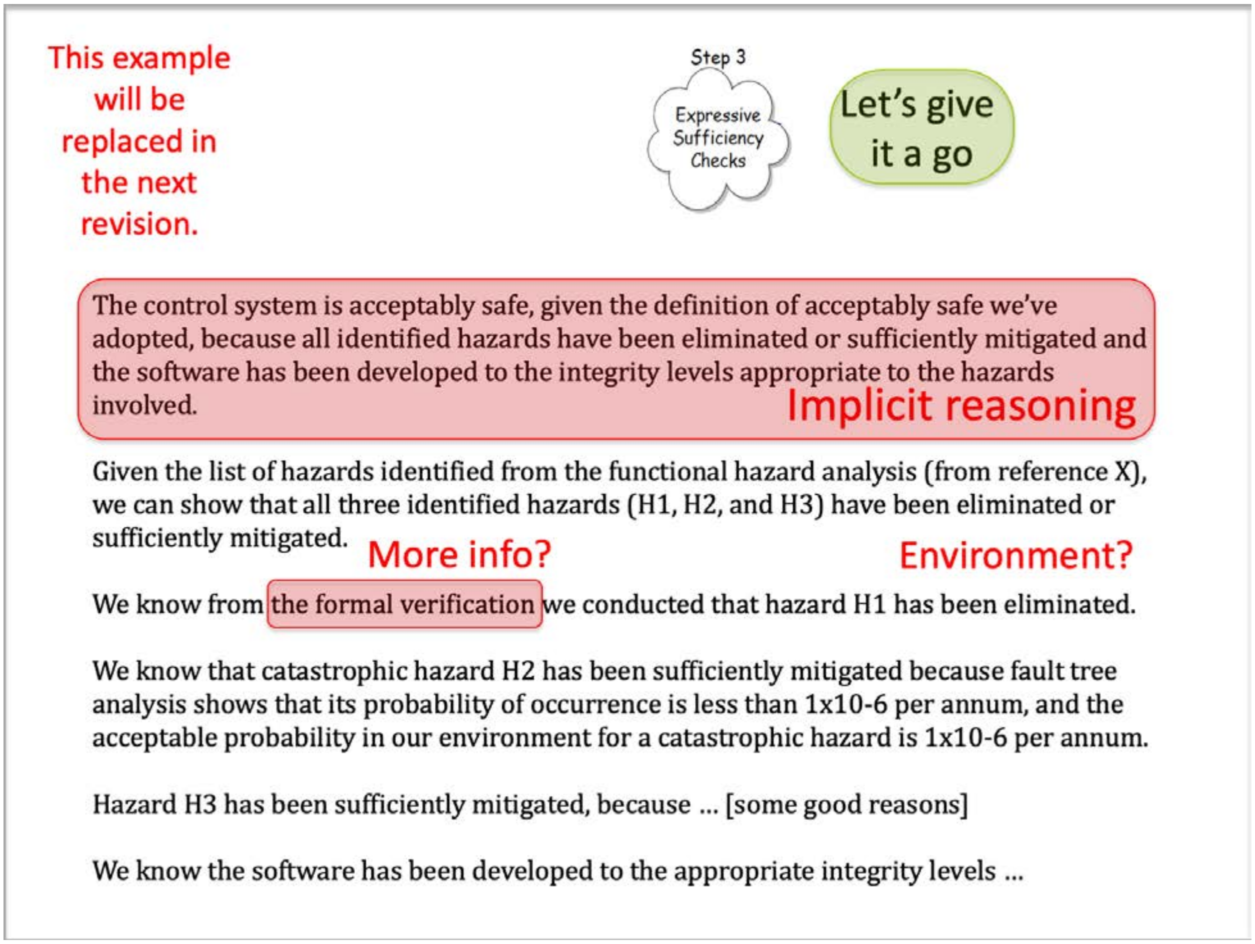
Finally (well, not really finally as there are other questions one may ask, too, but finally for this list), are relationships clear among argument elements? Can you tell which arguments are sub-arguments of which other ones? Can you work out which conclusions serve as premises for other arguments? And so on.

As you are probably expecting, we'll now see if you can find some sufficiency issues in our example. I've modified it to fix the structural problems we identified in step 2.



Please do not turn the page until you have attempted the exercise.

Here’s my answer.



As we already noted, the top-level argument has implicit reasoning.

We might also note that nothing at all is said concerning the environment in which the control system is supposed to operate.

And finally, most likely we need more information about the specifics of the formal verification performed relative to H1 before we can know whether relying on the verification provides sufficient grounds for believing the hazard has been eliminated.

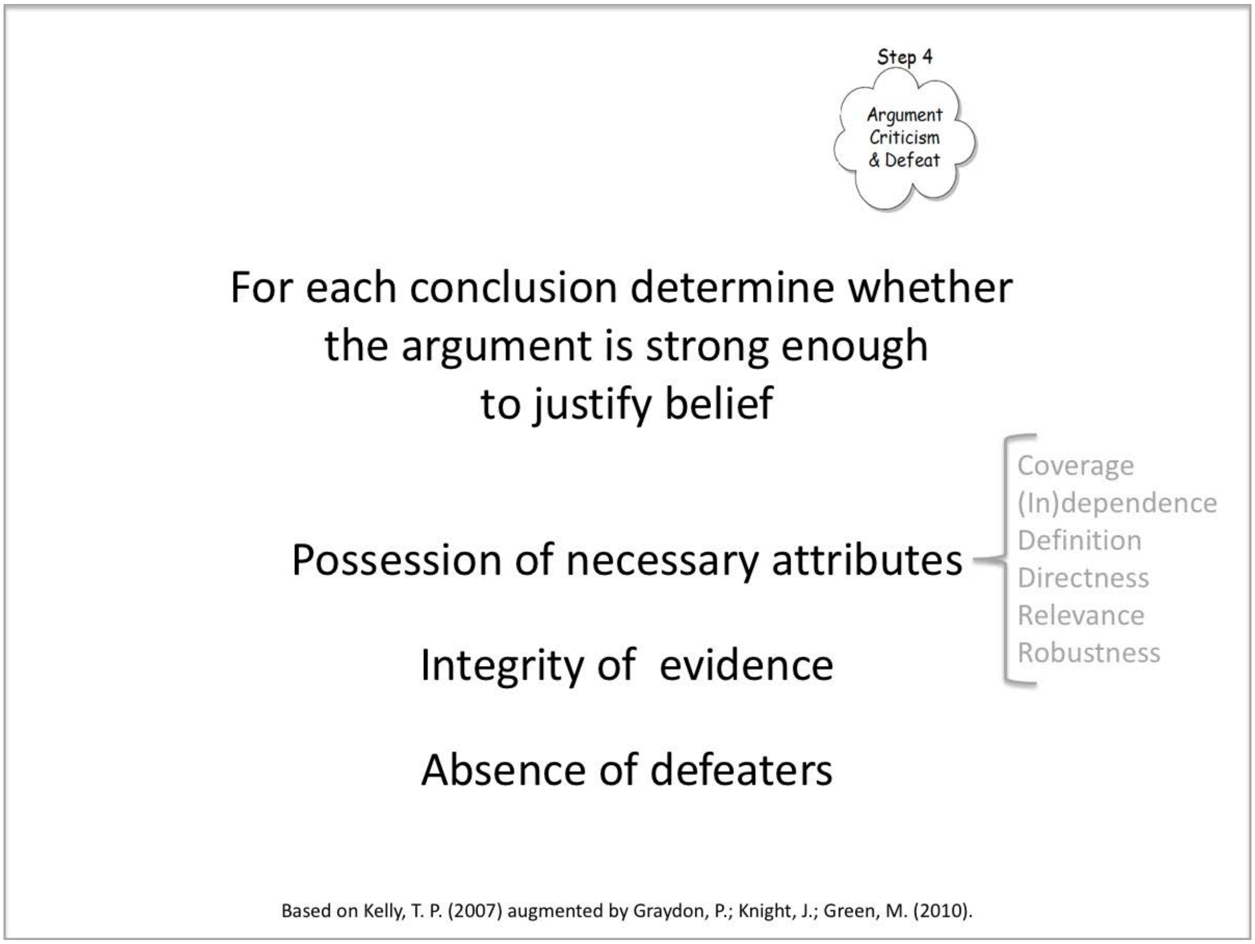
That’s the third step.

[Note to participants: What questions do you have?]

The final step is argument criticism, in which for each conclusion, we seek to determine whether the argument for it is strong enough to justify belief.

This is the most time-consuming and subjective step in the process.

In Kelly’s process, there are 3 aspects to this quest: (1) Possession of necessary attributes; (2) Integrity of evidence; (3) and Absence of defeaters.



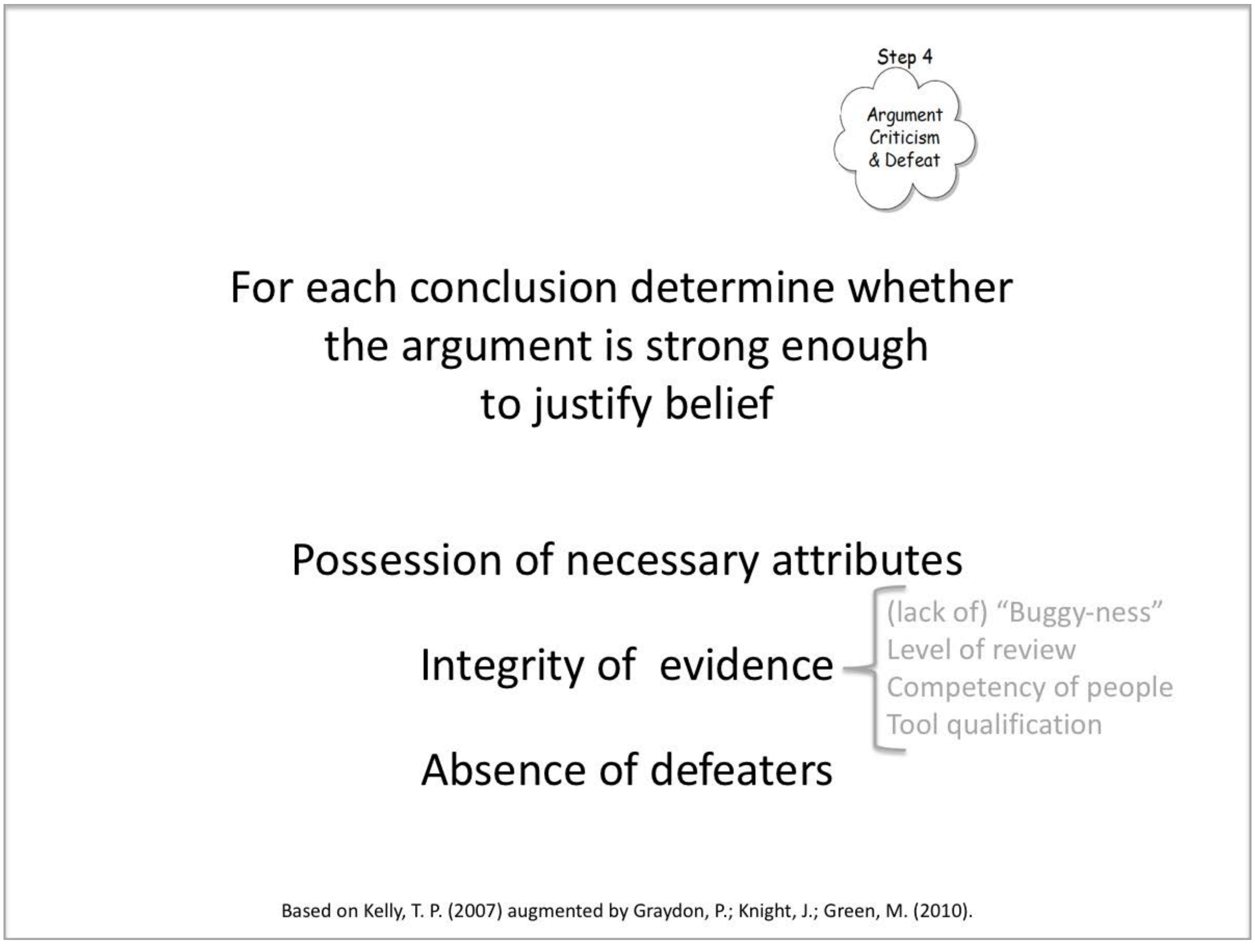
We're going to concentrate on the 3rd one of these, but I'll mention just a bit more about the first two.

'Attributes' here refers to attributes of the individual argument being considered, not of the overall case itself. In his paper, Tim lists the six attributes you see here (coverage, dependence/independence, definition, directness, relevance, and robustness), while noting that the list is not complete, and the attributes are not necessarily disjoint.

As one example, suppose an argument is claimed to support the conclusion, "All identified hazards have been addressed", but which is based on premises concerning only three hazards, when the hazard analysis shows seven hazards were identified. Such an argument would not have adequate *coverage*.

Concerning *integrity of evidence* Tim Kelly specifically mentions four considerations: (lack of) "buggy-ness", level of review, competency of people, and tool qualification.

The basic idea is that in evaluating an argument we need to have confidence that the grounded premises (as I've mentioned before, I consider this phrase a much better phrase than evidence) at the base of our argument really say what we think they say.



[Question to participants: What questions do you have about possession of necessary attributes and integrity of evidence?]

I want to talk now in a bit more detail about the last element, absence of defeaters.

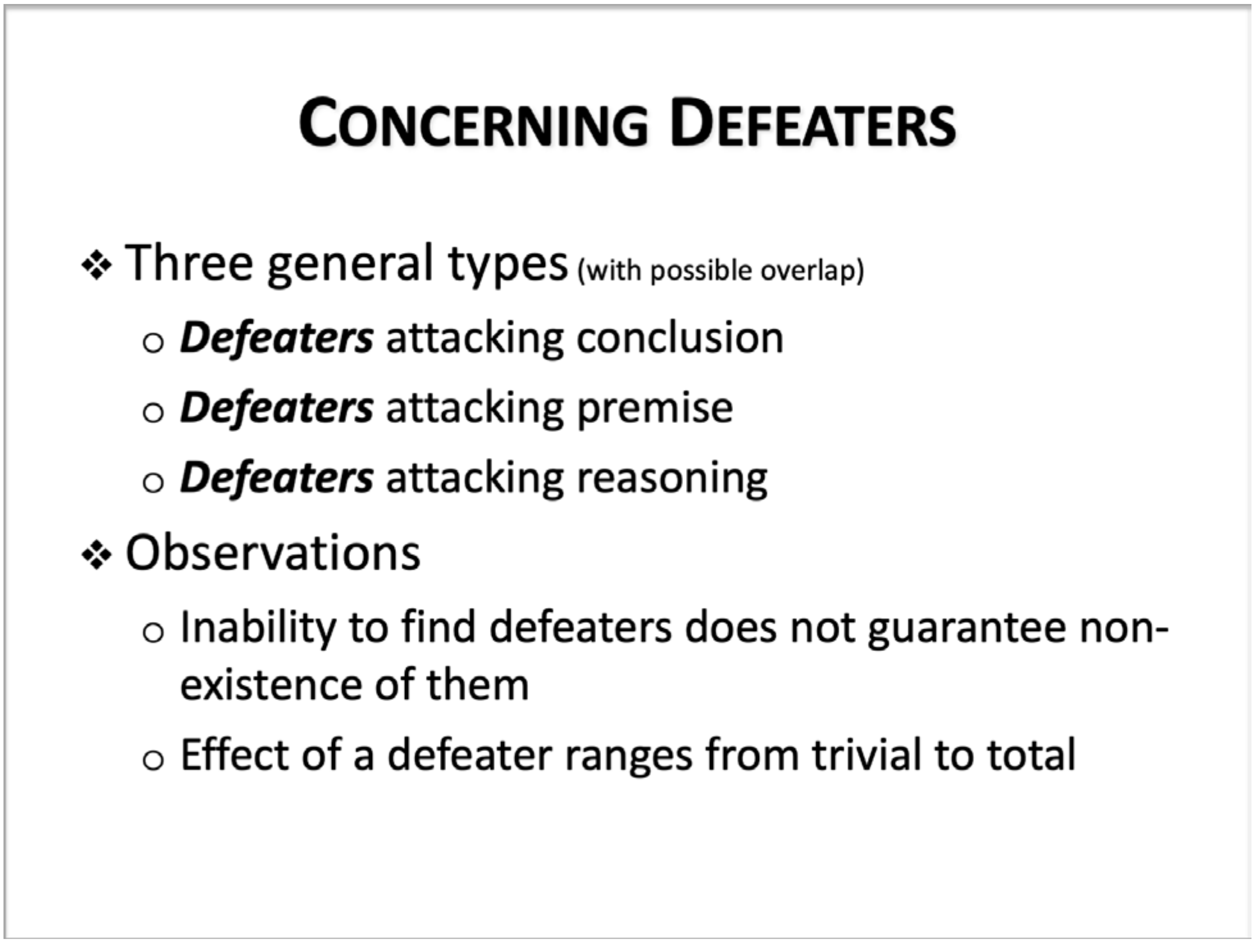
Although you may not have heard the term *defeater* before in quite this context, your intuitive notion of what it means is likely fairly accurate. Rather than giving an abstract definition[2], I'll talk about three basic types and then lead us through some examples.

There are three general types: *defeaters* that attack a conclusion; *defeaters* attacking a premise; and *defeaters* attacking reasoning.

Some treatments of defeaters give different names to the different types (rebutting, undermining, and undercutting); and some only distinguish between two types, grouping premise and reasoning defeaters together; but we're not going to use those names as they can be a bit confusing.

---

[2] The reader interested in the philosophical foundations of the concept of defeaters is encouraged to visit `https://plato.stanford.edu/entries/reasoning-defeasible/`. I have avoided using the phrase 'defeasible reasoning' in these educational materials based on prior experiences in which the use of the phrase caused more confusion than enlightenment.



The three-fold categorization is not absolute, as there can be defeaters that attack more than one element of an argument.

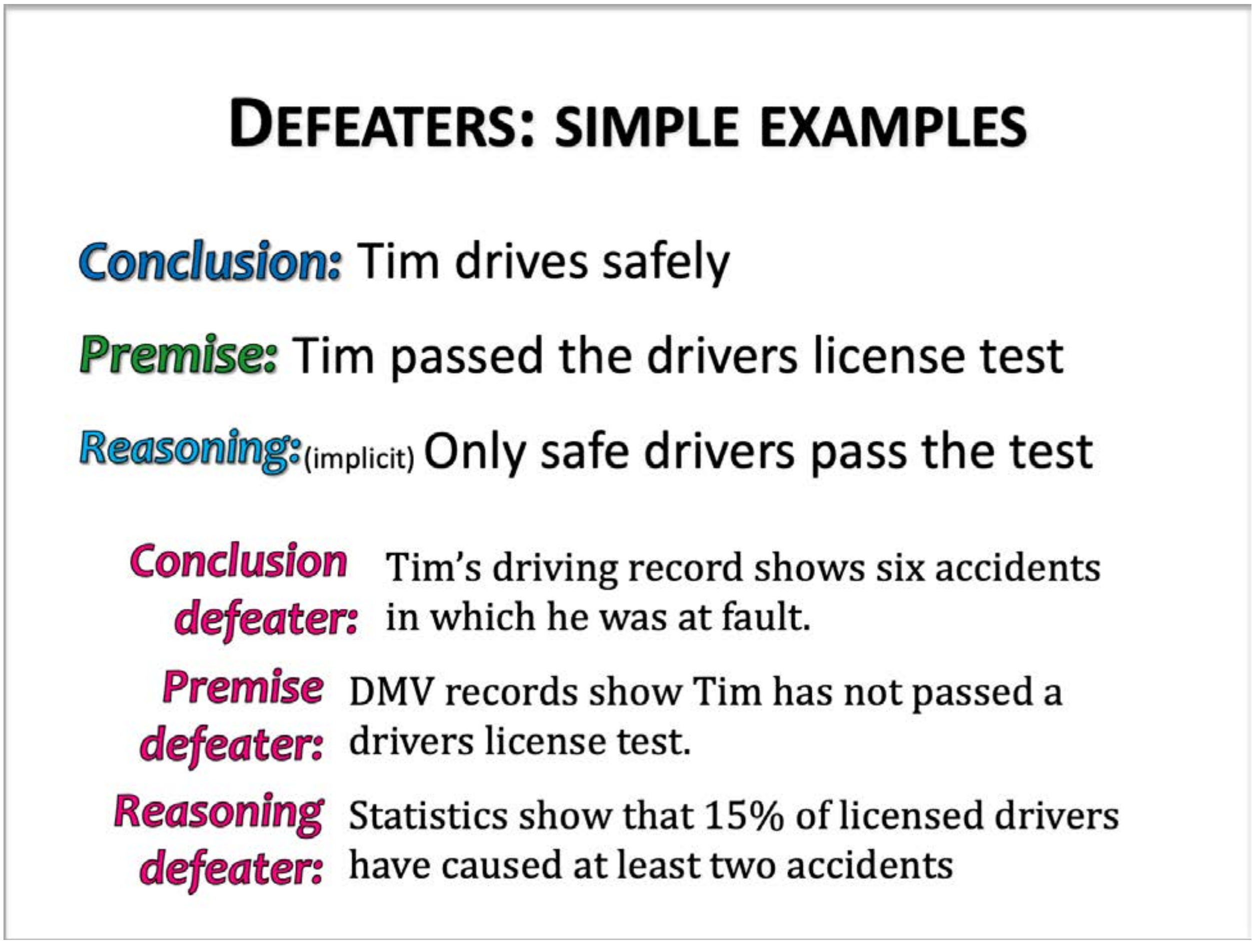
Before giving an example, I want to make two important observations concerning defeaters.

First, defeaters are partially analogous to software bugs in that the inability to find defeaters does not guarantee there aren't any.

Second the effect of a defeater on an argument ranges from trivial to total. Only defeaters that effectively attack the conclusion mean that the conclusion is necessarily not true.

Premise and reasoning defeaters show there's something wrong with the argument, but this may not necessarily mean that its conclusion is false. As an example, we will harken back to Module 1. Recall one of the simple examples we discussed was the following: Given (premise) "Annette was born in Lynchburg, Virginia" you should believe (conclusion) "Annette is a US citizen" because (reasoning) "People born in Virginia are US citizens." Supposed you discover that Annette was not born in Lynchburg, Virginia. You have defeated the premise. But the conclusion could still be true. As long as she was born in some other location in Virginia, the reasoning does not even need to change.

Also harkening back to Module 1 you may recall this simple argument.



For our purposes now, let's suppose that we have an agreed definition of what it means to 'drive safely' and that this definition involves, at least in part, the absence of 'at fault' accidents.

Let's start with the implicit reasoning, and consider what a defeater of this reasoning might look like. Well, suppose we have access to accident statistics that show 15% of licensed drivers have caused at least two accidents. Such statistics would certainly undercut our belief that *only* safe drivers pass the test.

This, in itself, doesn't mean that Tim doesn't drive safely, but it does mean that the argument provided should not give us confidence in his driving ability.

Consider the premise: "Tim passed the drivers license test". What's a defeater for this premise?

Well, the premise would be thoroughly undermined If DMV records show Tim has not passed the test.

Again, the argument now provides no confidence in the conclusion, but that alone doesn't mean the conclusion is false.

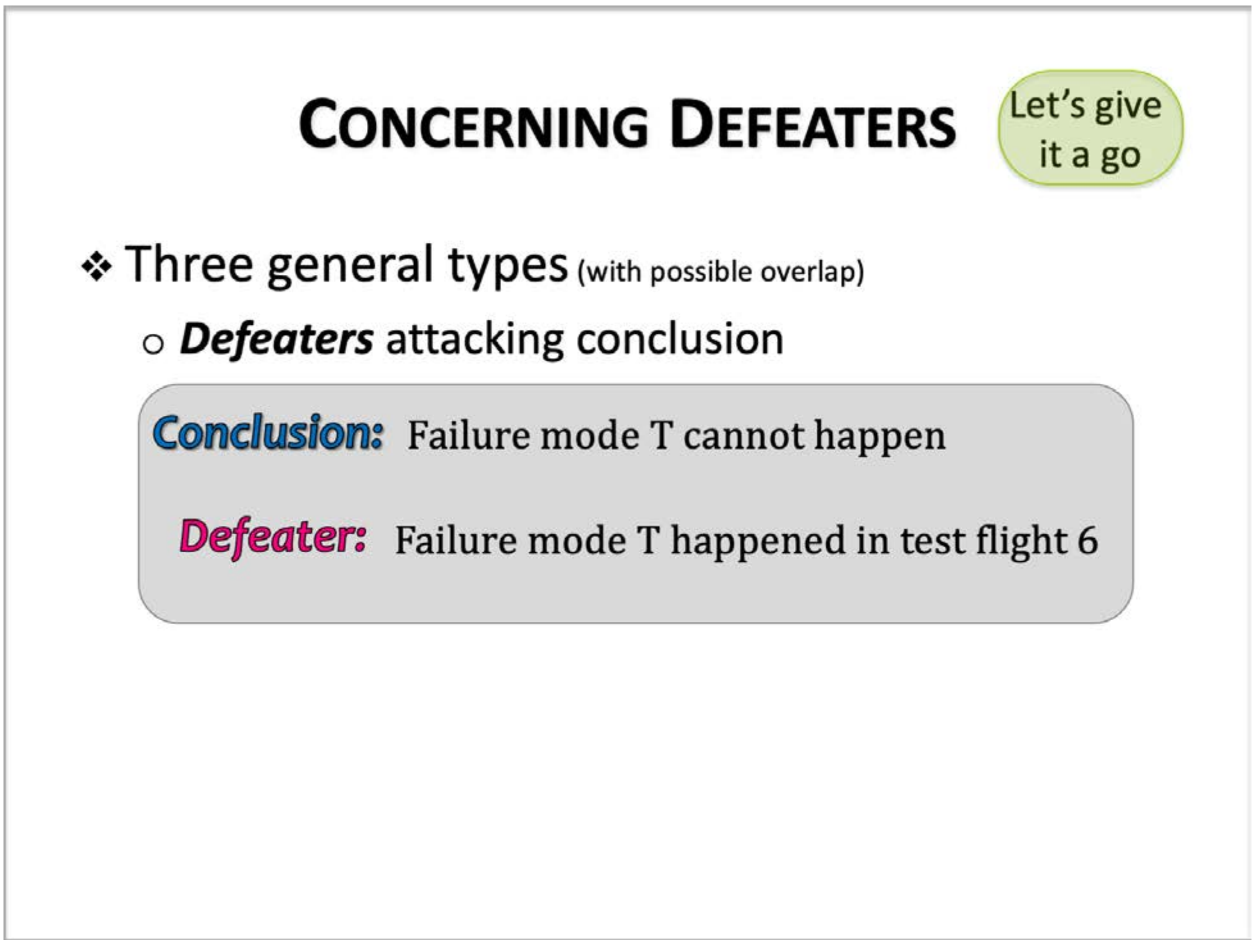
A defeater of the conclusion, on the other hand, does mean that the conclusion is false.

Here’s a possible example of such a defeater: “Tim’s driving record shows six accidents in which he was at fault.” Given such a record, I don’t know of anyone who would conclude that Tim drives safely.

Now that you’ve seen these simple examples, you’re ready to try some examples that are a bit more technically oriented. Let’s start with a conclusion defeater.

Suppose we have the conclusion “Failure Mode T cannot happen”.

What’s a defeater for it?

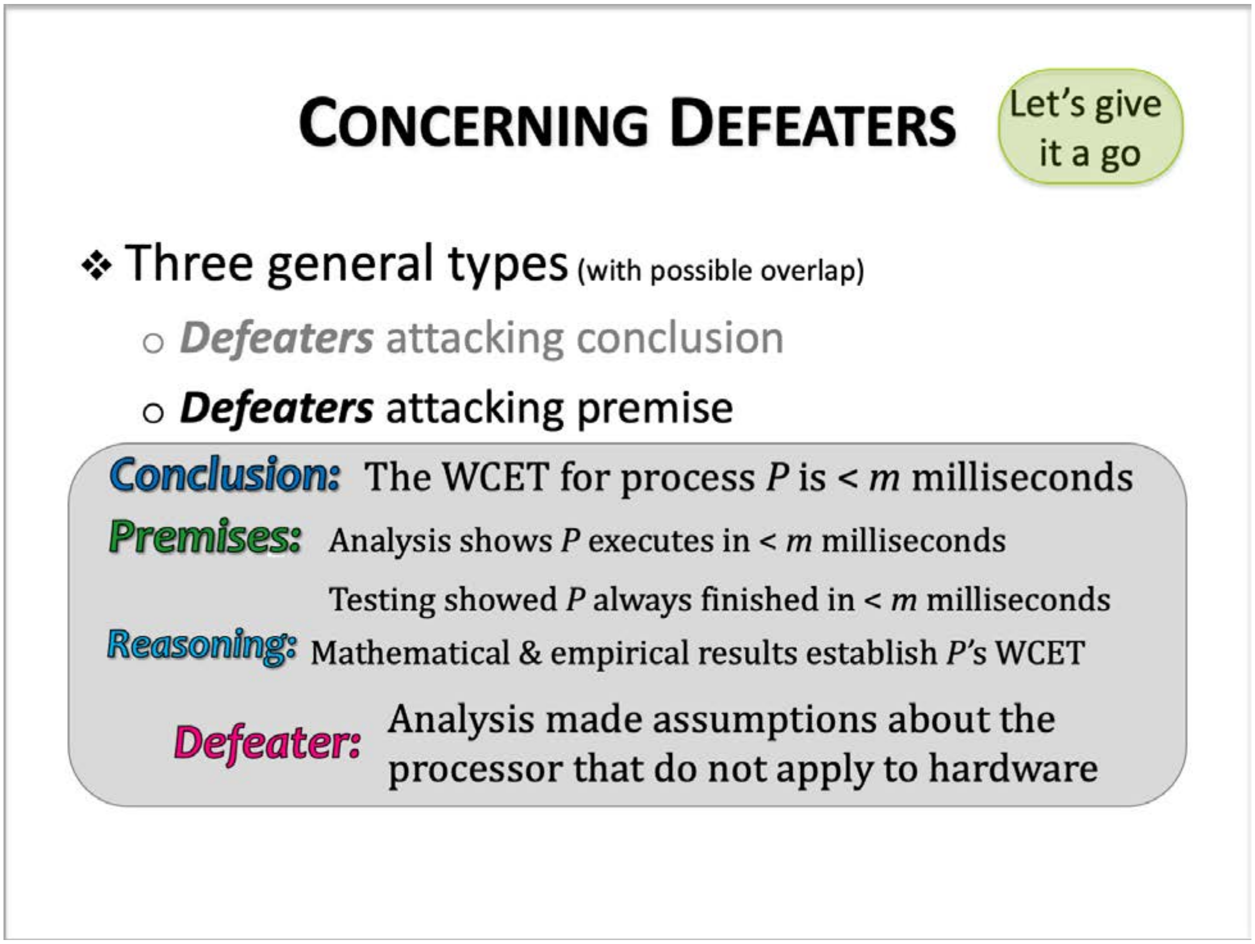


Here’s one example: “Failure mode T happened in test flight 6.” If it actually happened, the claim that it cannot happen cannot be true.

Let’s take a shot at finding a defeater of a premise.

Suppose we have conclusion “The WCET for process *P* is < *m* milliseconds”, with the two premises “The WCET for process *P* is < *m* milliseconds” and “Testing showed *P* always finished in < *m* milliseconds”, and the reasoning “Mathematical & empirical results establish *P*’s WCET” What’s an example of a defeater that attacks the truth of one or more of the premises?

Please do not turn the page until you have attempted the exercise.

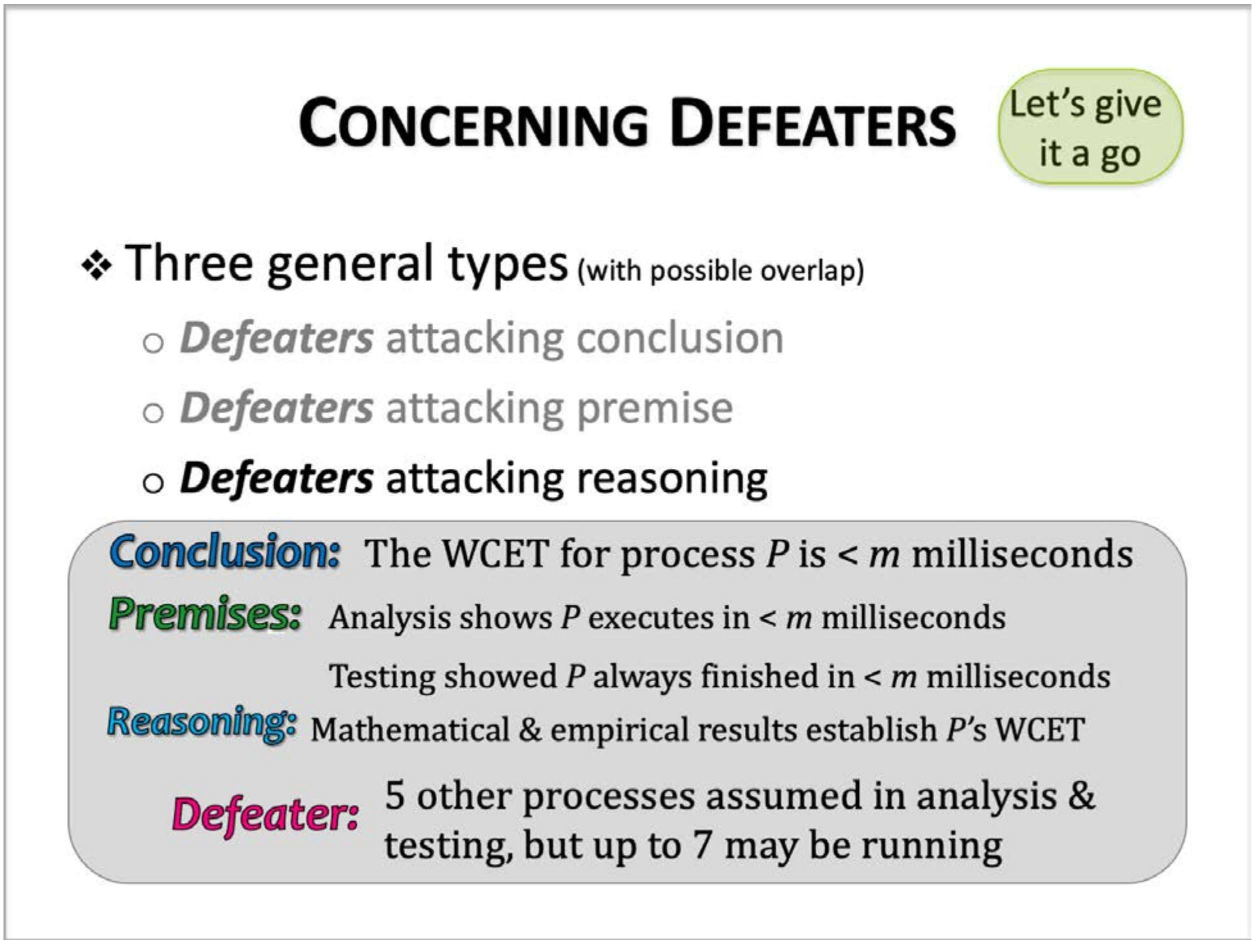


Here's one: "The Analysis made assumptions about the processor that do not apply to the actual hardware."

Finally, let's consider a defeater attacking reasoning using the same argument we just used, and assuming the premise defeater has been shown to not apply.

What's a possible defeater of the reasoning?

Please do not turn the page until you have attempted the exercise.



Here's one: "5 other processes assumed in analysis & testing, but up to 7 may be running"

If that assertion is true, the reasoning is weakened.

There's a lot more we could say here about defeaters, but this is a good time to pause for questions.

[Question to participants: Who has a question?]

Here are four questions for you to consider at your leisure:

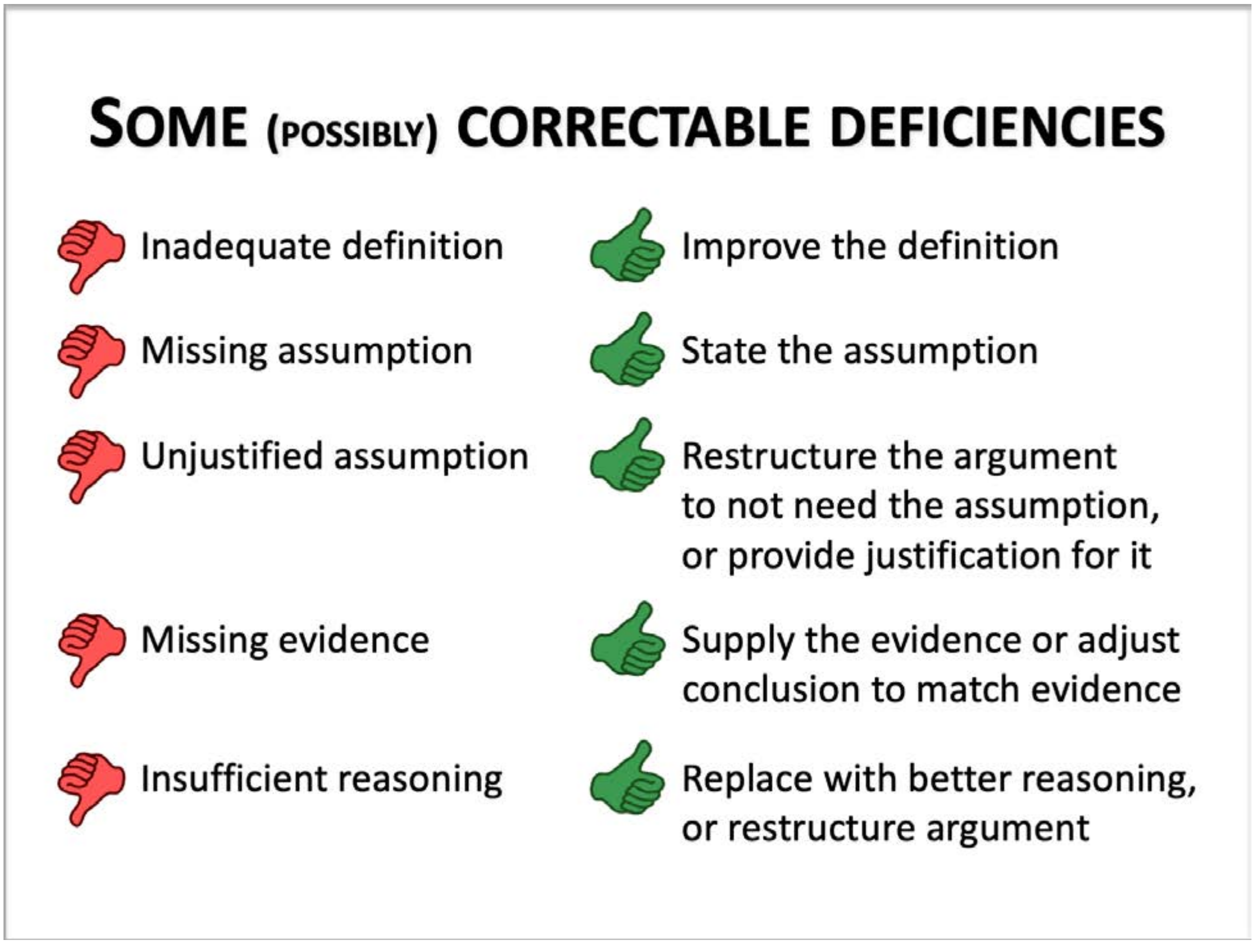
Does the number of possible defeaters of an argument say anything important about the cogency of the argument?

Does the number of resolved defeaters say anything important about the cogency of an argument? (A resolved defeater is a possible defeater that has been shown to not apply to the argument.)

Does the number of unresolved defeaters say anything important about the cogency of an argument?

How about the ratio of resolved to unresolved defeaters?

This slide lists some deficiencies that may be encountered in an argument, but for which there may be fairly simple corrections possible.



For an inadequate definition, improving the definition *may* be an easy thing to do.

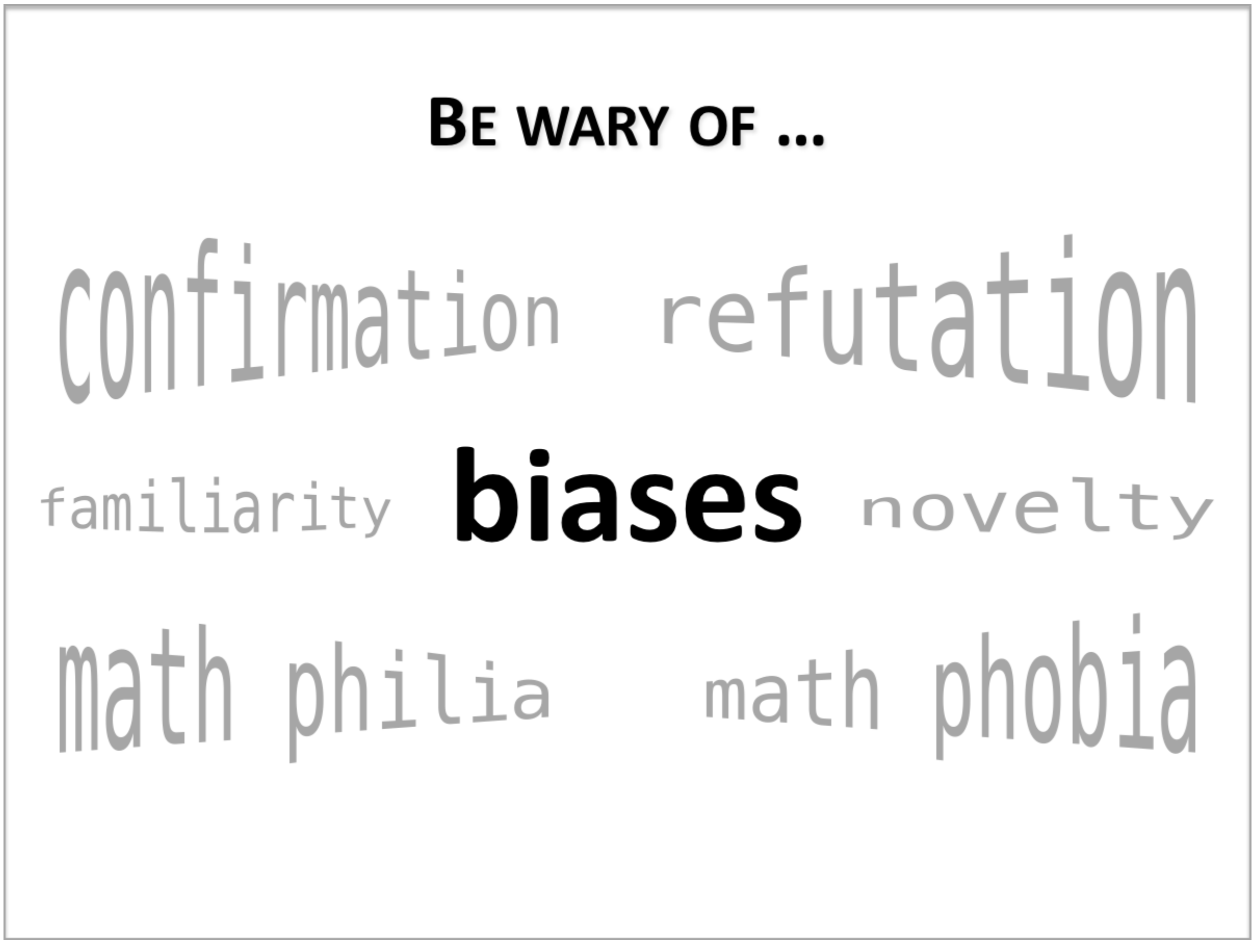
Perhaps a missing assumption can simply be stated.

It may be possible to eliminate an unjustified assumption, or to provide justification for the assumption.

For missing evidence, supplying the evidence may be possible, or perhaps the conclusion may be adjusted slightly to match the evidence. Say, for example the original conclusion was the system would be safe to operate with an ambient temperature between 0 and 100 degrees, but the available evidence only covers the range of 0 to 80 degrees. The temperature range in the conclusion could be adjusted accordingly.

And, as the final example, insufficient reasoning may be replaceable by a better one, or perhaps the argument can be restructured so this reasoning step is replaced altogether.

Let’s talk now for a bit about some things to be wary of when performing step four.



One thing to look out for is biases, of which there are several types.

The most common type of bias mentioned concerning assurance cases is confirmation bias.

In general this phrase refers to the tendency to interpret new pieces of information in a way that confirms what you already think, rather than to interpret it critically.

But some people (such as myself) may be prone to a bias of a slightly different sort, namely a tendency towards refutation, or, to put it slightly differently, to interpret *everything* critically.

Another pair of biases to be wary of are the love of math and the fear of math.

For some people, seeing numbers (probabilities for example) in an assurance case will cause them to think happy thoughts and be inclined to believe that the case is a good one.

For some other people (me for example) seeing numbers (probabilities in particular) in an assurance case will make them nauseous, and nearly certain that the case is rubbish.

Finally, the third pair of biases that can cause problems is familiarity and novelty. Some folks tend to give more credence to things they know, whereas others tend to give more credence to things that are new.

In general humans are much better at recognizing biases in others than we are at recognizing biases in ourselves. Having multiple people participate in the evaluation of an assurance case is one way to reduce the likelihood of the evaluation being skewed by biases.

Another set of things of which we need to be wary concern the cases themselves.

BE WARY OF ...

centipedes

We need to be wary of any argument that has a whole lot of premises for any particular conclusion.

BE WARY OF ...

book cases

Really big assurance cases should cause concern, even if the individual arguments are not centipedes, but simply the level of detail is very great.

Both centipedes and bookcases are worrisome because understanding them may well exceed the intellectual capabilities of even the brightest evaluator.

BE WARY OF ...

uniformity

Also worrisome are assurance cases in which everything seems to have been given the same level of attention, suggesting insufficient consideration of some of the issues we raised in Module 2 when we discussed the 5Ws.

BE WARY OF ...

automation

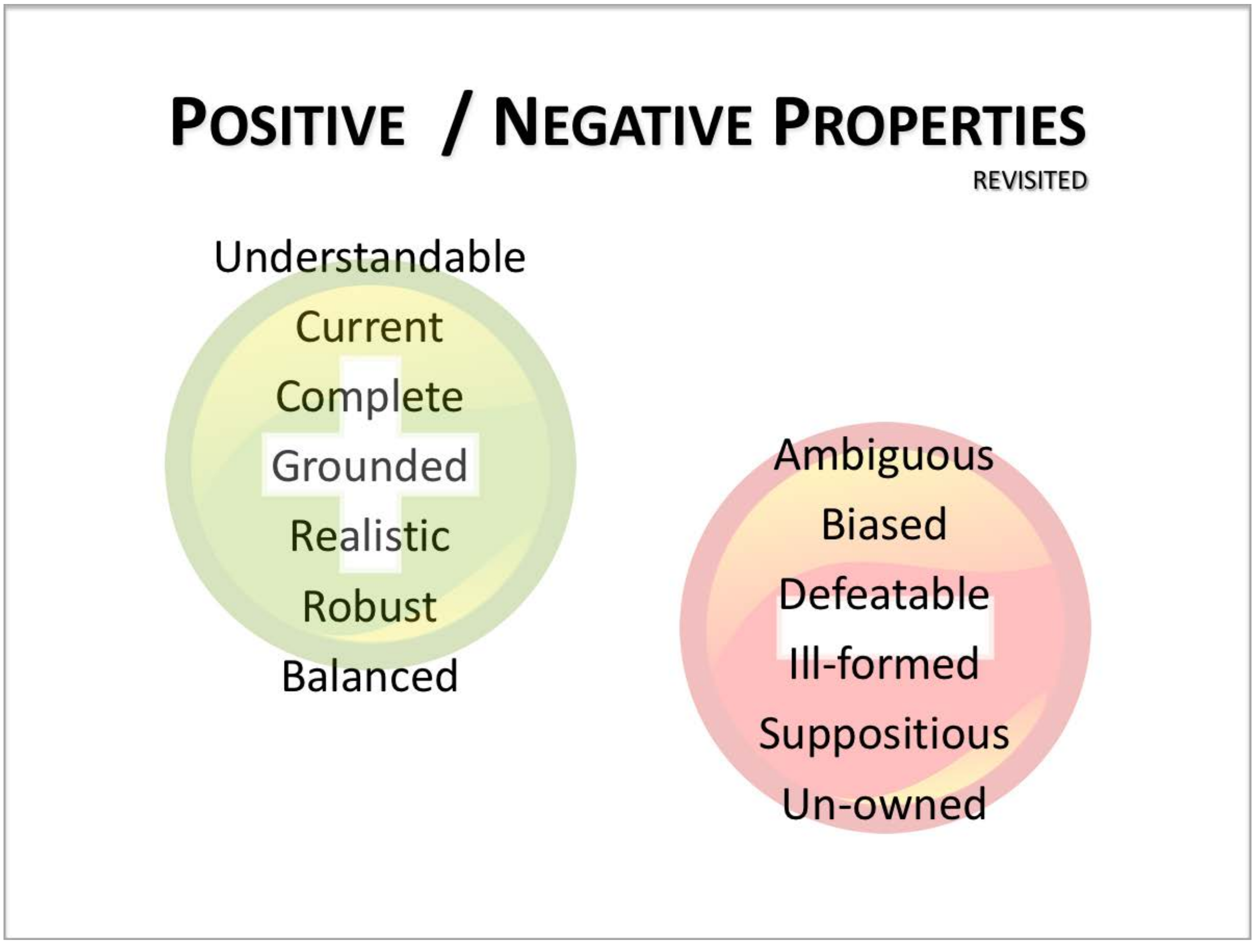
Finally, you should run as fast as you can away from any assurance case that someone says was generated automatically. As we've said several times in every module so far, a primary value of assurance cases arises from how they can stimulate careful thinking. Automation is the antithesis of careful thinking[3].

[Question to participants: What questions do you have about the causes of wariness?]

Before ending, let's quickly revisit the list of positive & negative properties.

---

[3] Careful, thoughtful use of automation to generate documentation to support an assurance case (for example, providing links to various bits of evidence) may be appropriate. Automating the creation of arguments (see Module 4) or the detailed evaluation of them is fraught with danger. Perhaps the day will come when sufficient foundations will have been laid for effective creation or evaluation, but those days are not yet here.



I explained the meaning of the positive properties earlier, but deferred discussing the negative ones until now.

Ambiguous is self-evident.

We just finished talking about various ways an assurance case can be *biased*.

As you can probably guess, a *defeatable* assurance case is one in which unresolved defeaters exist for important parts of the argument.

An *ill-formed* case is one that doesn't make it out of step 2 in the four step process.

By *Suppositious* I mean an assurance case with arguments based heavily on assumptions, for which no further argument or premises are provided.

Finally, *un-owned* refers to an assurance case for which the "Who" question hasn't been well answered.

Evaluating an assurance case is not easy, but it is not impossible. Subjectivity *is* necessarily involved, but subjectivity is necessarily involved in evaluating *anything* other than, perhaps, some aspects of pure mathematics.

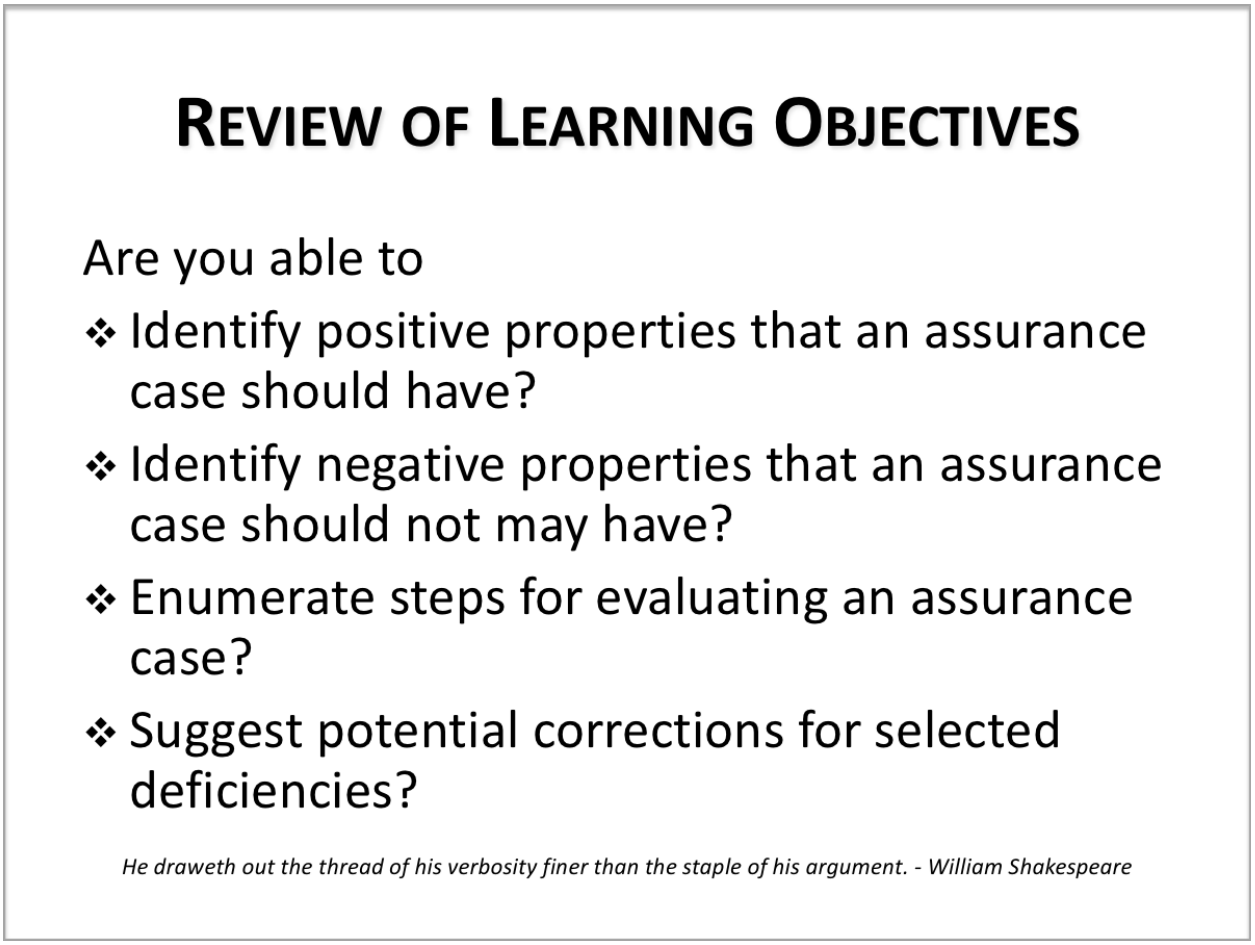
Just as assurance cases provide a framework for making *explicit* conclusions, premises, reasoning (and other argument elements), so too do they provide a framework for making the areas of subjectivity *explicit*, and thus subject to scrutiny.

Subjectivity that is subject to scrutiny is surely more desirable than subjectivity that's hidden.

[Question to participants: Any questions before we end by reviewing the learning objectives?]

At the beginning, I listed four things that I hoped you'd be able to do by the end of this module.

Here are those four things recast in the form of questions.

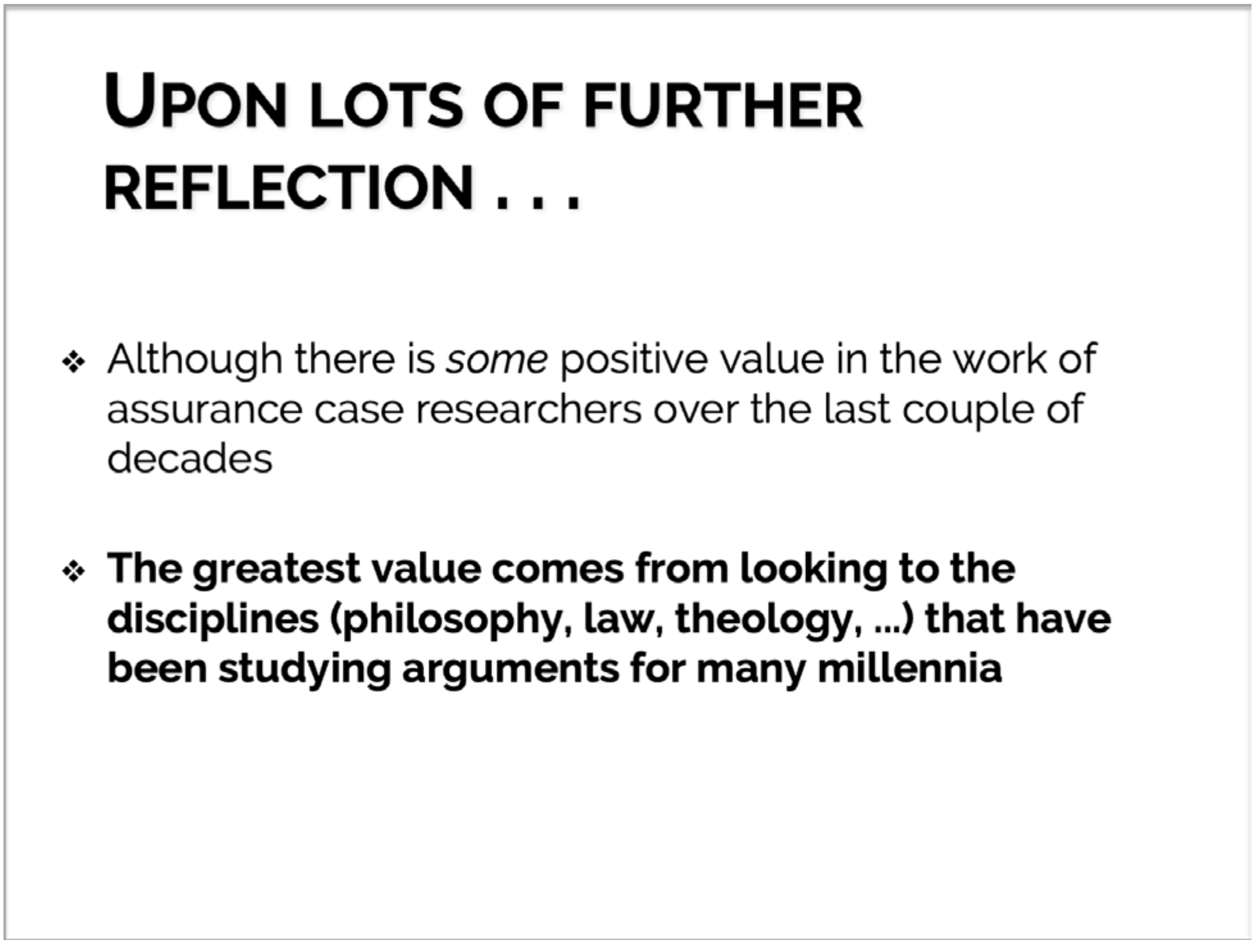


Think to yourself how you'd answer these questions.

After you've thought about the questions for a little bit, please ask me any questions that you still have for me.

If you have questions or comments about this material contact the author at `c.michael.holloway@nasa.gov.`

The following material is included without written commentary as a preview of what is likely to be coming when I revise this module.



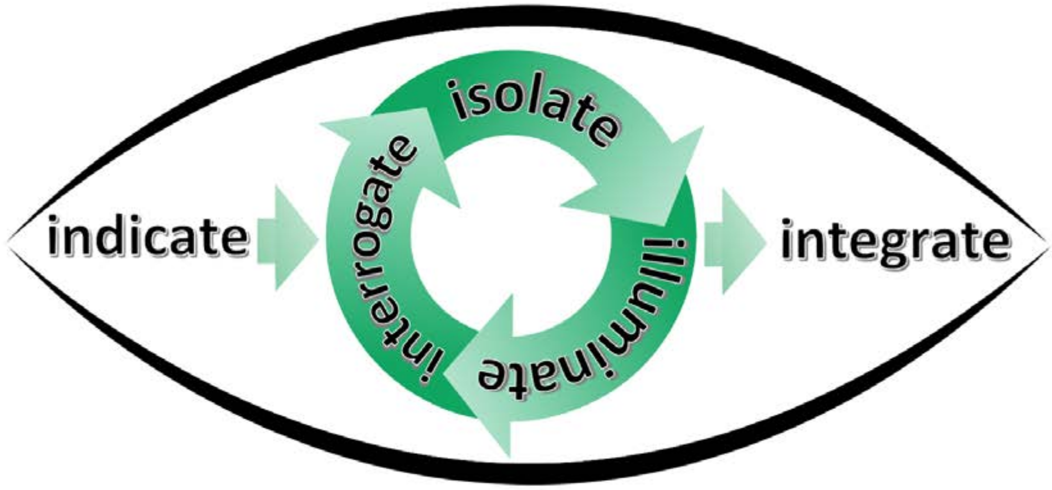
GOVIER'S ARG CRITERIA ARE AN EXAMPLE

For an argument to be considered *cogent* …

- ❖ It must have **a̲cceptable** premises. "That is, it [must be] reasonable for those to whom the argument is addressed to believe these premises."
- ❖ The argument's premises must be **r̲elevant** to its conclusion. "By this we mean that the premises state evidence, offer reasons that support the conclusion, or can be arranged into a demonstration from which the conclusion can be derived."
- ❖ The premises and reasoning provide good **g̲rounds** for the conclusion, that is, they "give sufficient reason to make it rational to accept the conclusion." (Note: Govier does not require explicit reasoning to be articulated, so she refers only to premises in this condition, but the idea is the same.) For the purposes of evaluating arguments for assurance cases, our standard must often be quite a bit higher than just "rational".

Govier, Trudy. 2010. *A Practical Study of Argument*. 7th edition. Belmont, CA: Cengage Learning.

the iTest



the iTest

"indicate, *v.*" www.oed.com/view/Entry/94416. 1. ***transitive***. To point out, point to, make known, show ...

"isolate, *v.*" www.oed.com/view/Entry/100081. 1. ***transitive***. To place or set apart or alone; to cause to stand alone, detached, separate, or unconnected with other things ...

"illuminate, *v.*" www.oed.com/view/Entry/91536. 4. ... to make luminous or clear; to elucidate.

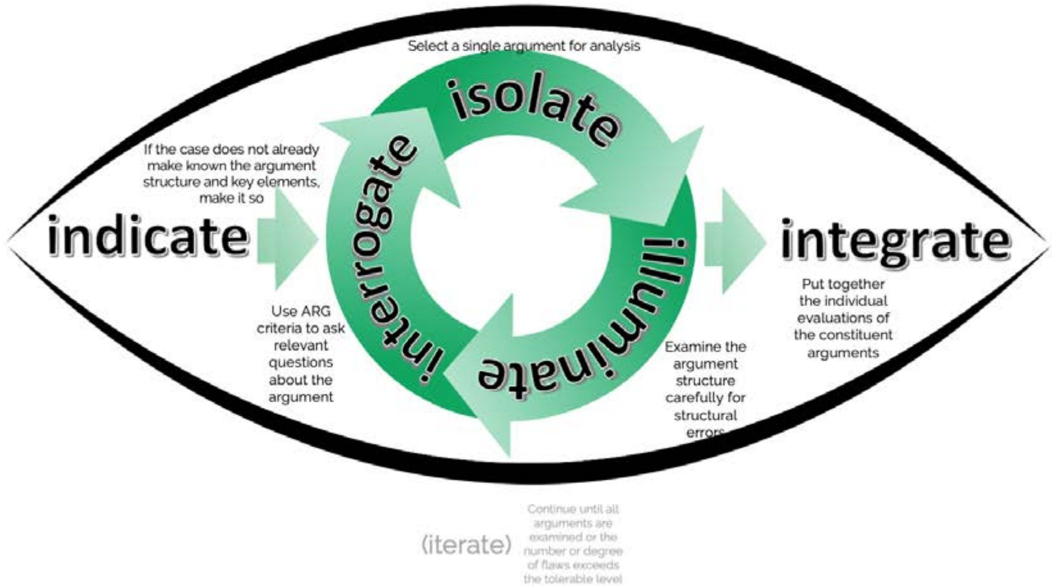
"interrogate, *v.*" www.oed.com/view/Entry/98260. 1.a. ***transitive***. To ask questions of, to question ... , esp. closely or in a formal manner; to examine by questions.

"iterate, *v.*" www.oed.com/view/Entry/100310. 4. ***intransitive***. ... To employ iteration ... "iteration, *n.*" www.oed.com/view/Entry/100312. 1. a. Repetition of an action or process ...

"integrate, *v.*" www.oed.com/view/Entry/97353. 1.b. To complete or perfect (what is imperfect) by the addition of the necessary parts

OED Online, Oxford University Press, December 2019, Accessed 19 February 2020.

the iTest



the iTest

indicate if the case does not already make known the argument structure and key elements, make it so

isolate select a single argument for analysis

illuminate examine the argument carefully for structural errors

interrogate use the ARG criteria to ask relevant questions about the argument

integrate put together the individual evaluations of the constituent arguments