progressive assurance using Evidence-based Development

Jeremy.Dick@integrate.biz

Summer Software Symposium 2008
University of Minnisota

"Assuring Confidence in Predictable Quality of Complex Medical Devices"

key messages

- product assurance is best achieved *progressively* by collecting and reviewing *arguments* and supporting *evidence* in parallel with its development
- Evidence-based Development (EbD®) is a uniform approach to progressive assurance fully integrated into the product development lifecycle

approach

- present concepts that have grown out of the discipline of requirements engineering, mainly from aerospace and defense sectors:
  - requirements management
  - rich traceability
  - the W-model
- show example requirements
- show example tool support

agenda

- assurance
- requirements management
- types of evidence
- Evidence-based Development
- conclusions

integrate
systems engineering

assurance

integrate

assurance objectives

core objectives of assurance are:

- to ***build confidence*** and
- to ***reduce risk***

typical assurance concerns may be whether:

- emergent technical solutions will *satisfy their requirements*
- these solutions are derived from *controlled*, *appropriate* and *effective* processes
- deployed systems will be *fit for purpose*

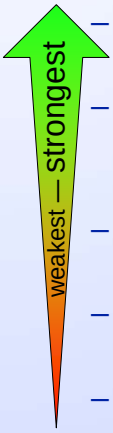
assurance viewpoints

need to answer questions such as:

- are our requirements complete and correct?
  - *requirements validation*
- do our designs discharge our requirements?
  - *design verification (DQ)*
- do implemented systems comply with designs?
  - *system verification (IQ, OQ)*
- are our processes compliant and effective?
  - *design assurance (DQ)*
- are deployed systems fit for purpose?
  - *system validation and certification (PQ)*

assurance techniques

- five basic classes of verification and validation technique:
  - *measurement* – we establish correctness by measurement, test, etc
  - *analysis* – construing correctness by mo… analytical techniques, potentially based o… data
  - *inspection* – construing correctness by examination of some feature or characteristic
  - *read-across* – construing correctness by analogy – appealing, for example, to a similar proven design
  - *escalation* – construing correctness by appealing to the correctness of supporting requirements or designs



integrate
systems engineering

requirements management

integrate

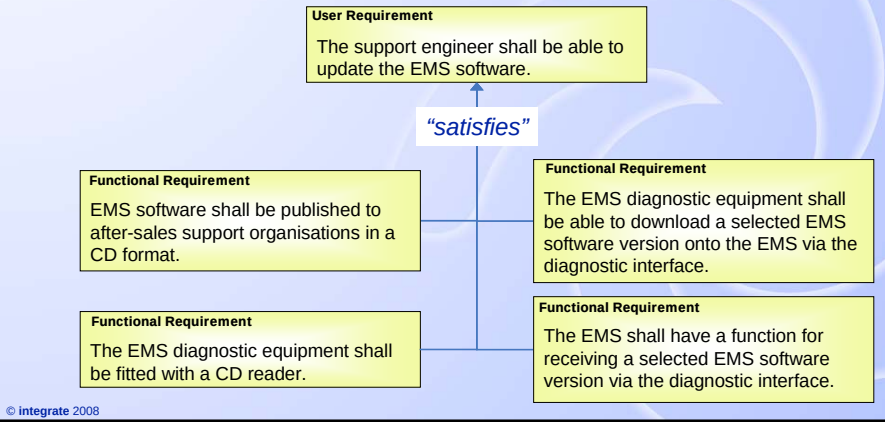
requirements management

- discipline of eliciting, expressing, satisfying, verifying, tracing, evolving and reusing requirements
- focuses on individual statements of requirement
- each requirement statement should be:
  - **singular**: each statement is a single traceable element
  - **identified**: each statement is uniquely identified
  - **understandable**: each statement is clear and precise
  - **unbiased**: does not impose a solution on the next layer
  - **quantified**: each statement has acceptance criteria
  - **testable**: each statement can be validated/verified
  - **traced**: to satisfying requirements and tests

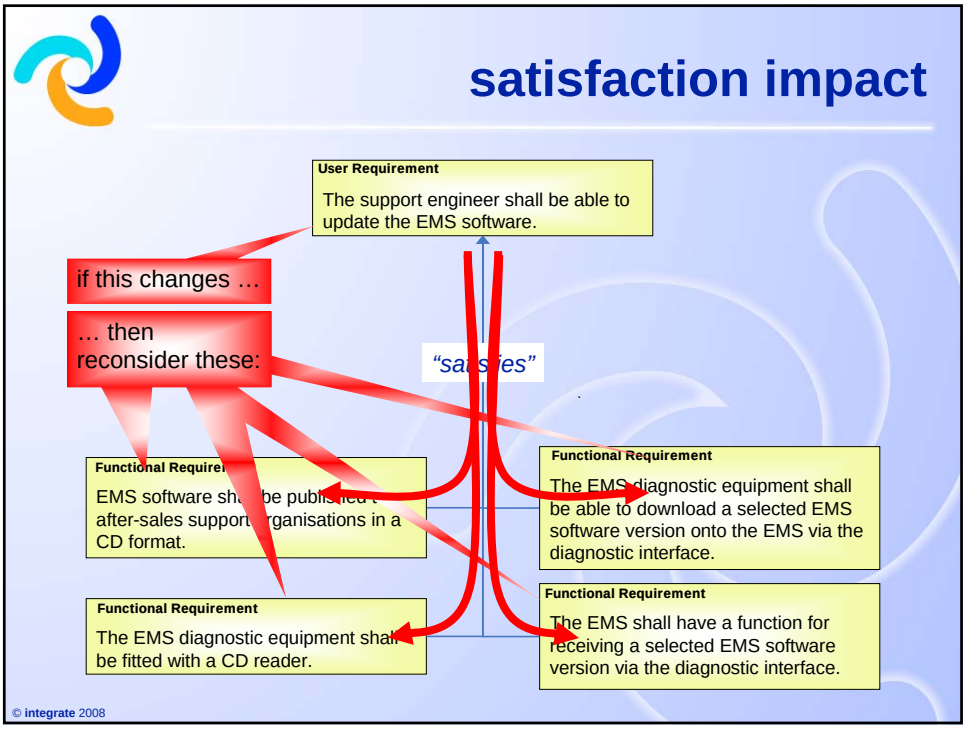
requirements tracing

a means of

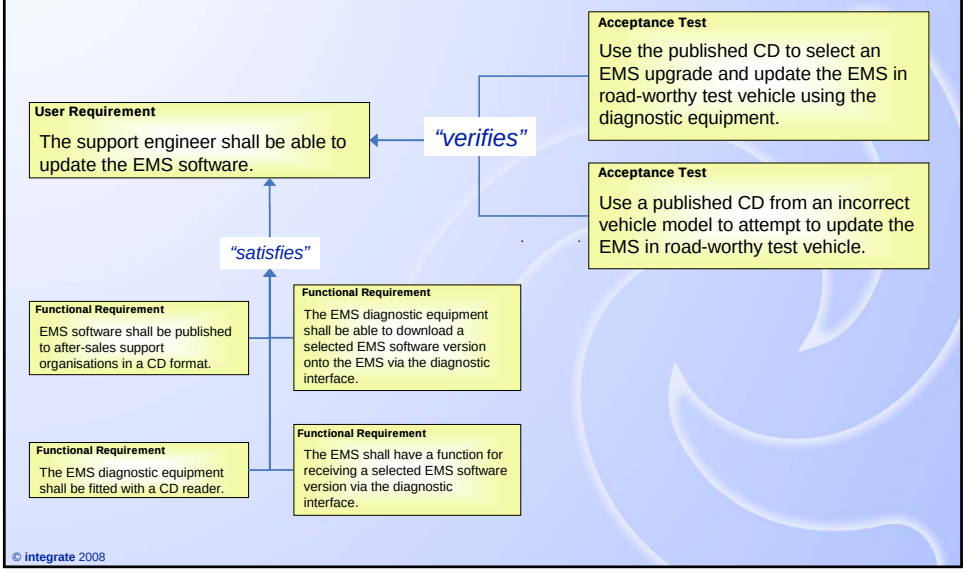
- recording relationships between artefacts
- performing impact analysis



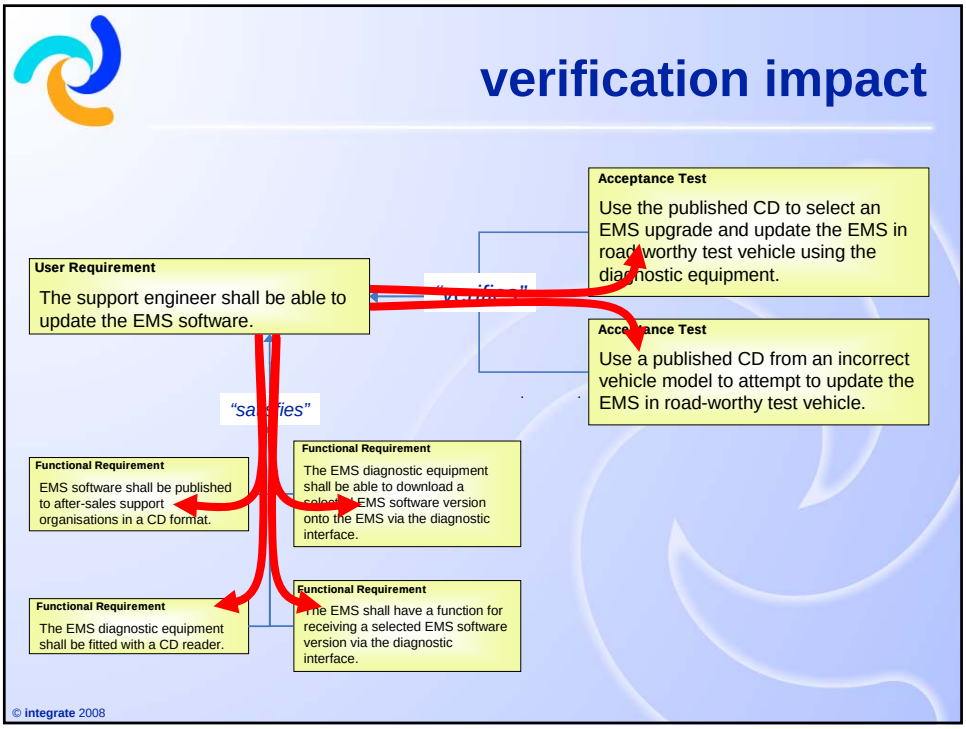
satisfaction impact



verification tracing



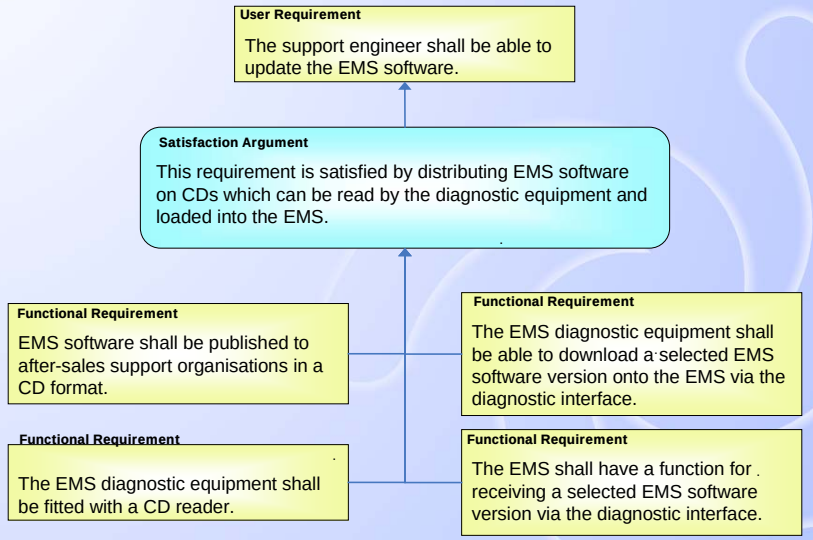
verification impact



arguments to enrich traceability

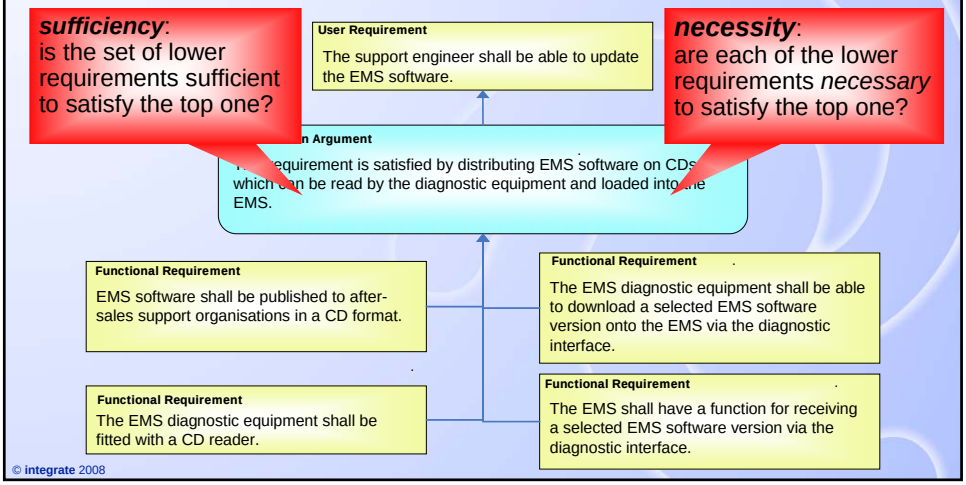
- rich traceability records underpinning rationale in the form of *arguments*
- an argument is gathered in stages over time
  - progressively enriched as understanding deepens
  - changes character throughout the lifecycle
- in general terms, an *argument* aims to demonstrate a *conclusion*, based upon the truth of a set of *premises*
- safety cases are a well understood example
  - communicate a *rigorous* and *defensible* argument, supported by *evidence*, that a system is acceptably safe to operate in a particular context

satisfaction argument

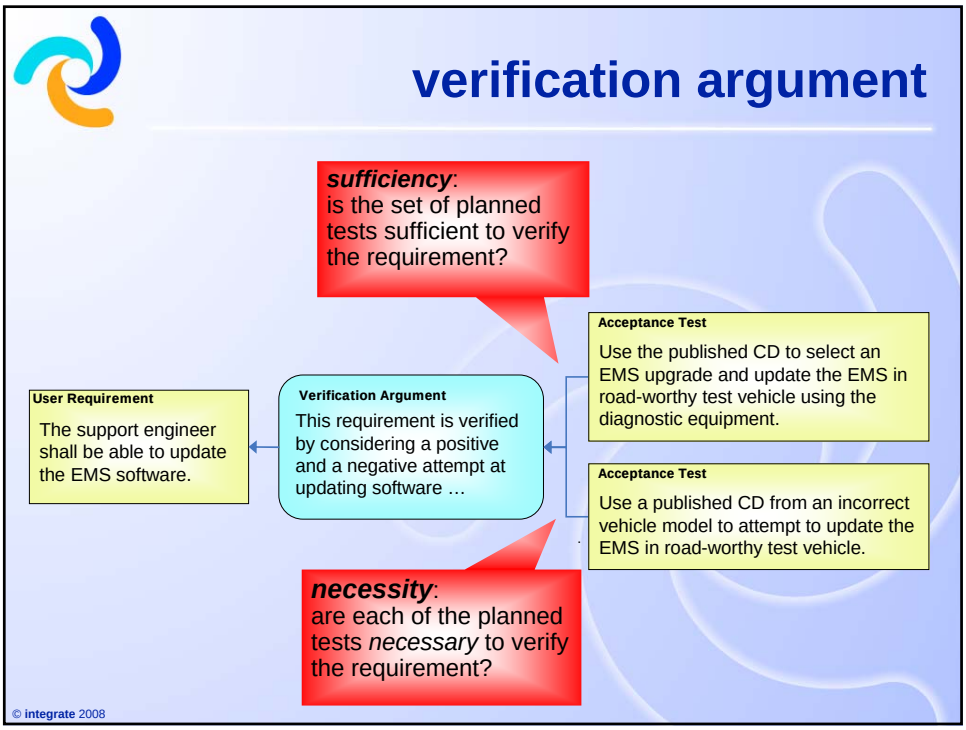


reviewing arguments

- vital to review the *relationship* between layers
- two key questions:



verification argument



rich traceability benefits

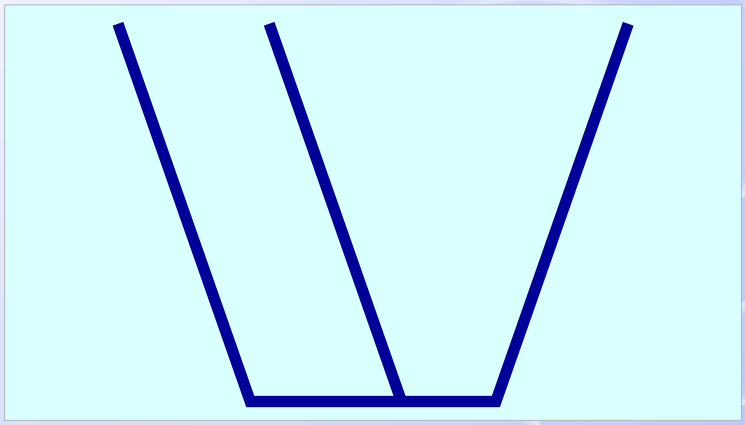
- engenders greater thought and confidence
- documents design thinking and rationale
- focuses review of key relationships
- improved ability to manage change
- placeholder for collection of evidence
- progressive construction of assurance case

integrate
systems engineering

types of evidence

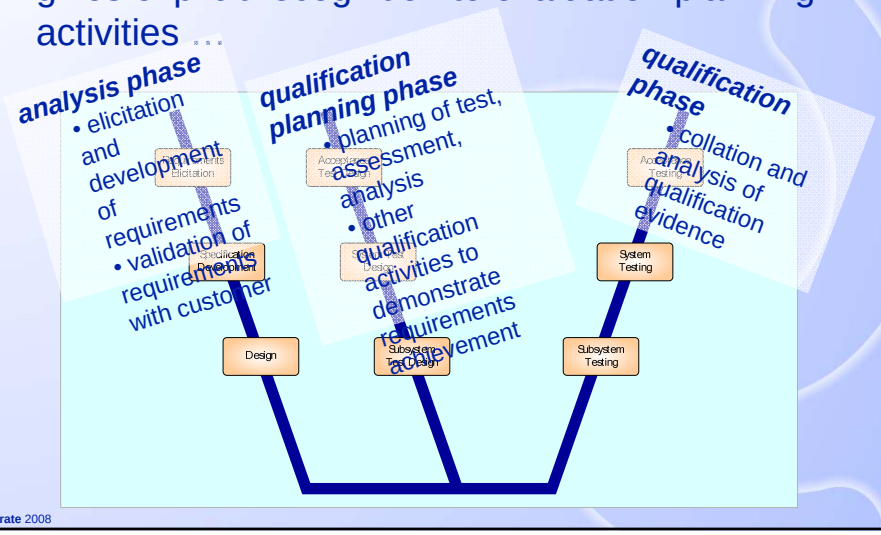
lifecycle – the ‘W’ model

- a refinement of the more familiar ‘V’ model



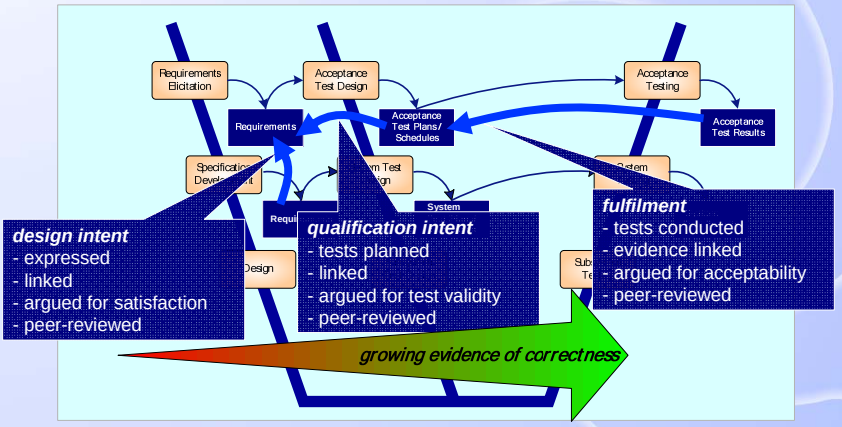
lifecycle – the ‘W’ model

- gives explicit recognition to evaluation planning activities …



lifecycle – the 'W' model

- … allows representation of three key phases of a development artefact's lifecycle



assurance emphasis

emphasis of assurance changes throughout the lifecycle:

- initial focus is on ***intent***
  - to build confidence that we are getting there
- ultimate focus is on ***fulfilment***
  - to endorse the fact that we have got there

classes of argument

- different classes of argument are needed to support the different lifecycle phases
- **analysis argument**
  - expresses *how* a set of lower level goals *will* satisfy a higher level goal
- **qualification planning argument**
  - expresses *how* a given set of qualification activities *will* deliver sufficient confidence that a goal has been achieved
- **qualification argument**
  - expresses *why* evidence from qualification activities supports the claim that the goal *has* been achieved

intent

intent

fulfilment

integrate
systems engineering

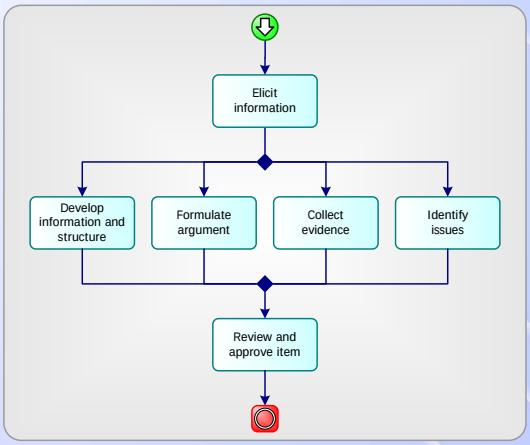
evidence-based development in practice

EbD® principles

- EbD® builds on rich traceability
  - includes W-model thinking
  - includes lifecycle-based argumentation
  - ***broadens*** scope to whole-system assurance
  - ***adds*** confidence-driven argumentation
  - ***focuses*** on summarising what has been done (and why), with links to externally held, discipline-specific evidence
  - ***adds*** explicit link to process requirements
- provides the *evidential backbone* for the system development process

EbD® micro-process

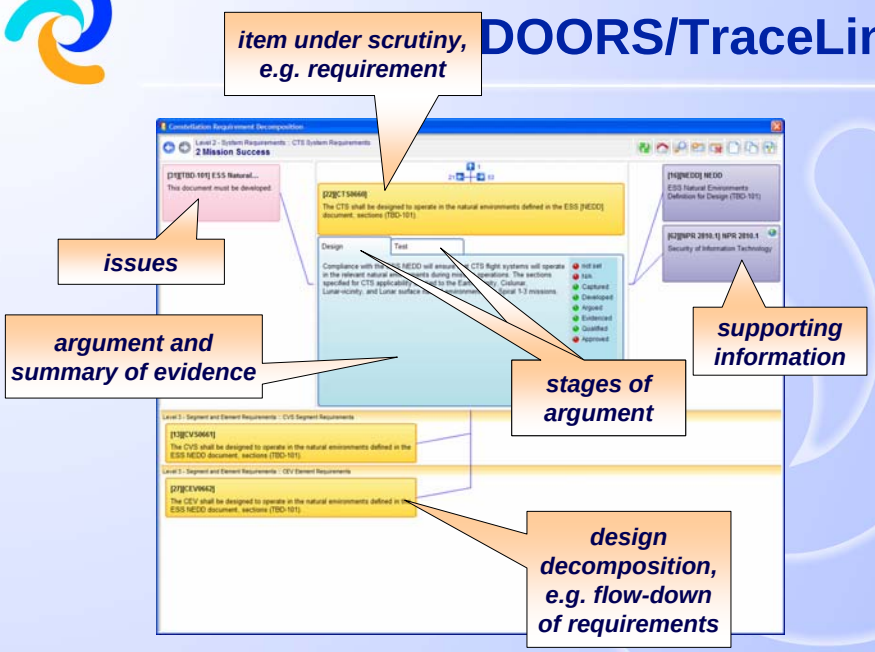
- micro-process applied to every development artefact (requirement, test case, etc)



example tool — DOORS/TraceLine

- DOORS is a leading requirements management tool from Telelogic (an IBM company)
- DOORS/TraceLine is an extension for managing and visualising information and its traceability
- a powerful and intuitive browser provides a highly visual environment in which you can
  - view, navigate and edit linked information held in DOORS
  - arrange this information in task and viewpoint-specific views
  - create graphical and textual content and traceability reports

DOORS/TraceLine



integrate
systems engineering

conclusions

EbD® benefits

- progressive assurance
  - collection of growing body of evidence for fitness-for-purpose of a system in parallel with its development
- recognise the evolution and diversity of viewpoints
  - arguments relating to intention and fulfilment
  - accommodate the broad range of analysis techniques
- uniform approach
  - collection, presentation and review of arguments
  - focus the practitioner on the same ‘mental model’ at every stage
- improve integrity and cost-effectiveness of certification

want to know more?

workshop tomorrow morning

"Evidence-based Development: processes and practices"

presentation

- expressing effective requirements
- requirements development and tracing
- types of argument
- reviewing arguments
- the assurance case
- example tool demo

workshop

- worked examples

integrate
systems engineering

QUESTIONS?

Jeremy.Dick@integrate.biz

www.integrate.biz
+44(0)1225 859991
+44(0)7875 415365