Certifiably Safe Software-Dependent Systems: *Challenges and Directions*

ICSE 2014 – Future of Software Engineering

John Hatcliff
Kansas State University
United States

Alan Wassyng
McMaster University
Canada

Tim Kelly
University of York
United Kingdom

Cyrille Comar
AdaCore, Paris
France

Paul Jones
US Food and Drug Administration
United States

Support:

Work supported in part by the US National Science Foundation (NSF) (#1239543), the NSF US Food and Drug Administration Scholar-in-Residence Program (#1355778), the National Institutes of Health / NIBIB Quantum Program, the US Air Force Office of Scientific Research (AFOSR) (#FA9550-09-1-0138), the Ontario Research Fund, and the Natural Sciences and Engineering Research Council of Canada

Foundational Principles

Challenge: establish foundational principles that provide a common basis for software safety assurance across domains and applications.

- Many standards address safety-critical software
- Differing philosophies regarding, e.g.,
  - software criticality
  - requirements on development processes
- Differences between standards make it hard to translate evidence of compliance between standards.
- Number of standards and differences differences between standards
  - can be bewildering to anybody operating in the domain,
  - can be a significant barrier in the education of new practitioners and researchers.

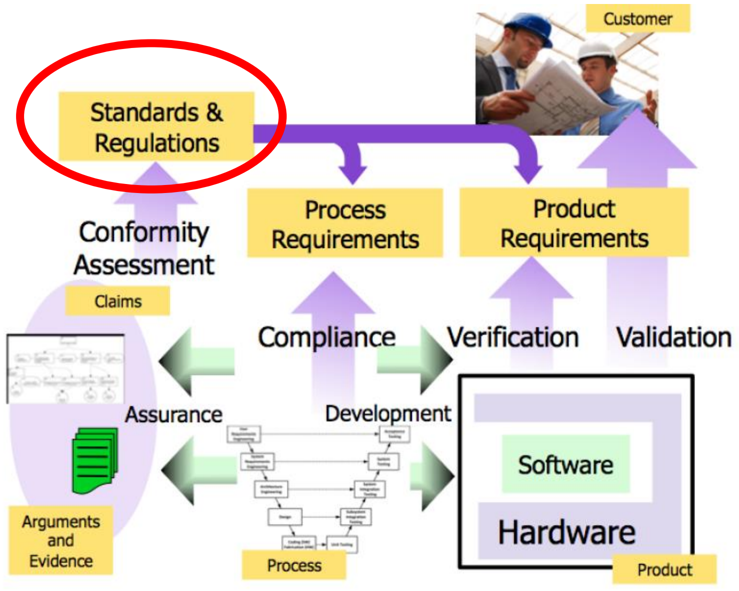
SAE ARP4761 Guidelines and Methods
QualityCoacl
ISO 14971:2009
AUTOSAR
ISO 26262
IEC 60601-1 3rd Edition
RTCA DO-178C Summary of Changes
INNOVATION. COLLABORATION. SOLUTIONS. SUCCESS.

Standards

Criteria in Safety Certification

Challenge: Standards impose requirements on products and processes and specify conformance criteria

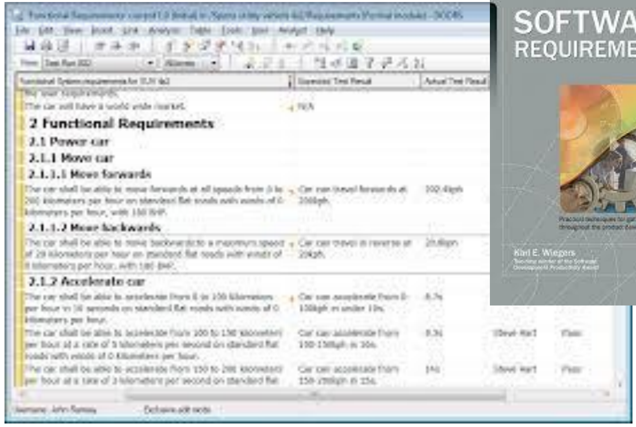
- Standards play a key role in providing uniform community-developed assessment criteria for systems within a safety-critical domain.
- What is the appropriate use of standards in safety certification of a software-dependent safety-critical system?
- What are the necessary and sufficient criteria that should be required in those standards?
- What specific processes and products should these criteria address?



Requirements

Challenge: establish valid requirements, i.e., requirements specifications that are complete, correct, unambiguous, and yet understandable to all stakeholders.

- Mainstream IT…
  - deficiency in requirements is the biggest source of unanticipated cost and delay
  - products issues surface through usage experience
- Certifiably Safe Systems
  - Requires introduction of safety requirements (and techniques for discovering those)
  - Requirements engineering should facilitate the validation needed for assurance
  - assurance by a third party before deployment
- In CS and Software Engineering, most of the focus is on verification rather than gap between real needs and requirements



The car will have a world wide market. N/A

2 Functional Requirements

2.1 Power car

2.1.1 Move car

2.1.1.1 Move forwards

[illegible]

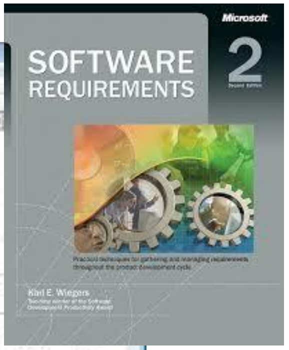
2.1.1.2 Move backwards

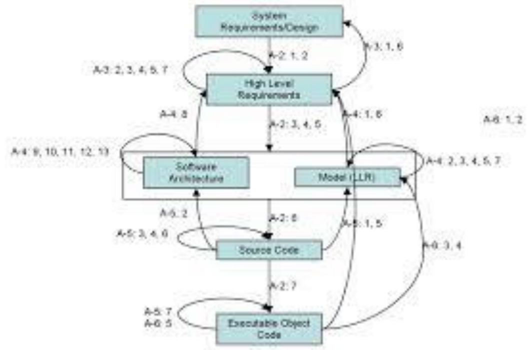
[illegible]

2.1.2 Accelerate car

[illegible]

Username: John Ramsey Exclusive edit mode

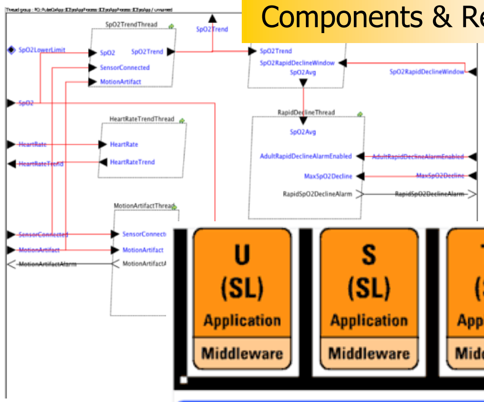


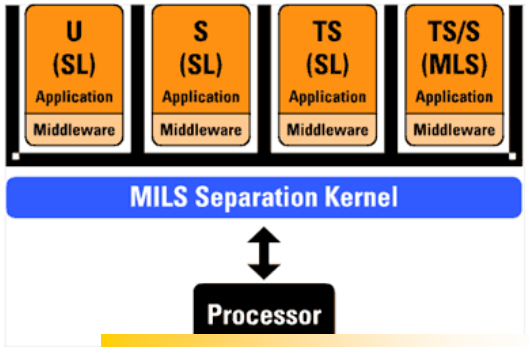


Compositional Certification

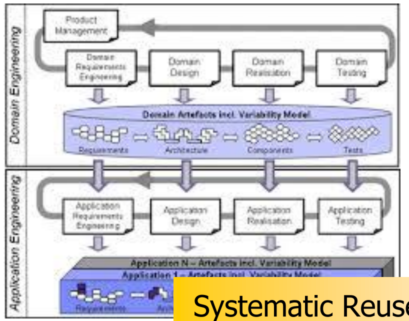
Challenge: Develop engineering and assurance approaches that support compositional certification and reuse of certified components while maintaining the same confidence levels in safety as one would have when assessing complete systems.

- Many modern development strategies emphasize (de)composition and reuse
  - Component reuse, product-lines, platforms, etc.
- Current certification regimes focus on certifying entire systems
  - Mechanisms for reusing certification artifacts and assurance claims in different contexts are limited
- Research is needed to determine the extent to which
  - individual components could be certified to conform to interface properties
  - in the context of rigorously defined architectures that constrain emergent properties, and
  - the interface properties, architectural principles, and associated assurance would be sufficiently strong so as to allow systems assembled from those components to avoid a full assessment of all component implementations to justify correctness and safety.


Components & Reuse



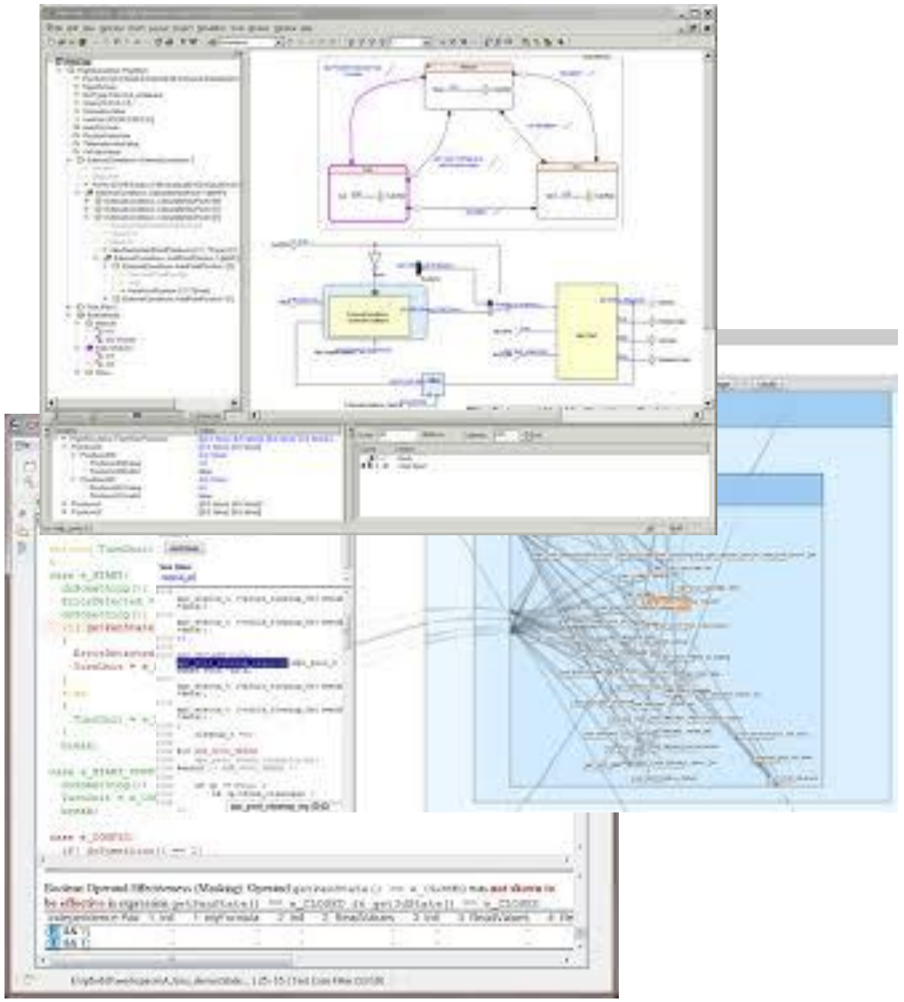
Safety/Security-critical Platforms


Domain Engineering
Product Management
Domain Requirements Engineering
Domain Design
Domain Realisation
Domain Testing
Domain Artefacts incl. Variability Model
Application Engineering
Application Requirements Engineering
Application Design
Application Realisation
Application Testing
Application N – Artefacts incl. Variability Model
Application 1 – Artefacts incl. Variability Model
Systematic Reuse in Product Lines

Use of Tools in Certification

Challenge: If certification can be viewed as a demonstration of conformance to safety-related requirements, how should tools integrate with this process?

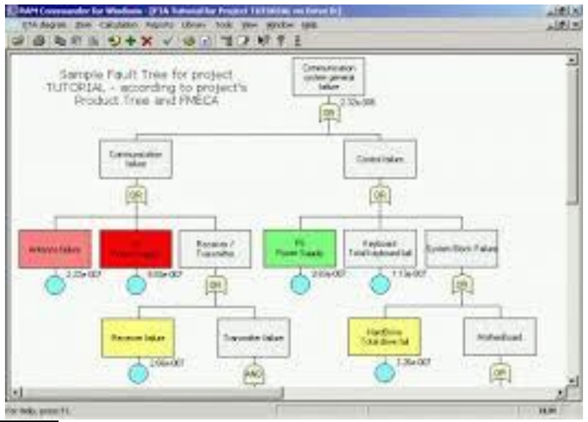
- How do we establish confidence in these tools that participate directly or indirectly in this demonstration?
- How much confidence do we need in the tools?
- How can the claims, evidence, and assurance artifacts associated with the use of different tools be combined to substantiate an overall assurance case for a system?
- How can we create a standards and regulatory ecosphere that will grow the market of certification-relevant tools and encourage innovations within this space?



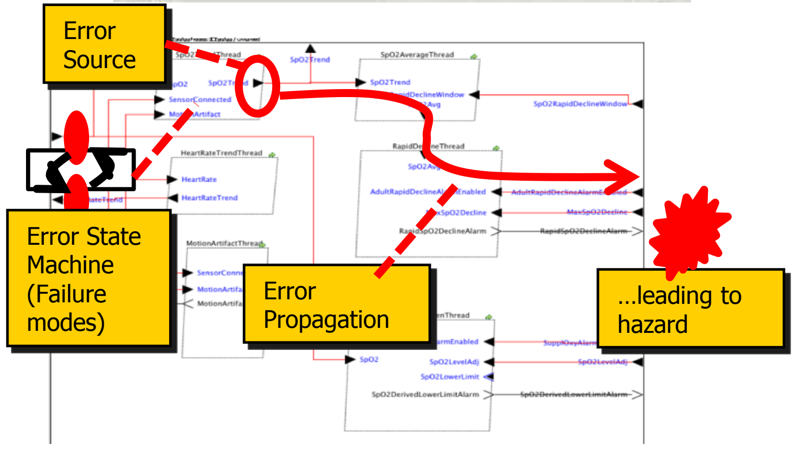
Automation of Hazard Analyses

Challenge: Hazard analysis drives the creation of safety requirements, drive design, and plays a key role in safety assurance arguments. How can we provide the same level of modeling, automated tooling, and methodology support as for other aspects of development?

- Significant modeling and automation tooling for other aspects of the life cycle, but little support for modeling, automation of hazard analysis and integration, e.g., with requirements tools.
- Results in...
  - a lack of rigor in execution and documenting of the analyses
  - difficulties in managing and updating results as design and implementation evolves,
  - difficulties on the part of certification agents in assessing the completeness and accuracy of results
- Challenge...
  - hazard analysis requires reasoning about unplanned interactions,
  - unintended behaviors,
  - behaviors that the system may exhibit when components are failing or when functions are erroneous



Sample Fault Tree for project TUTORIAL - according to project's Product Tree and FMECA



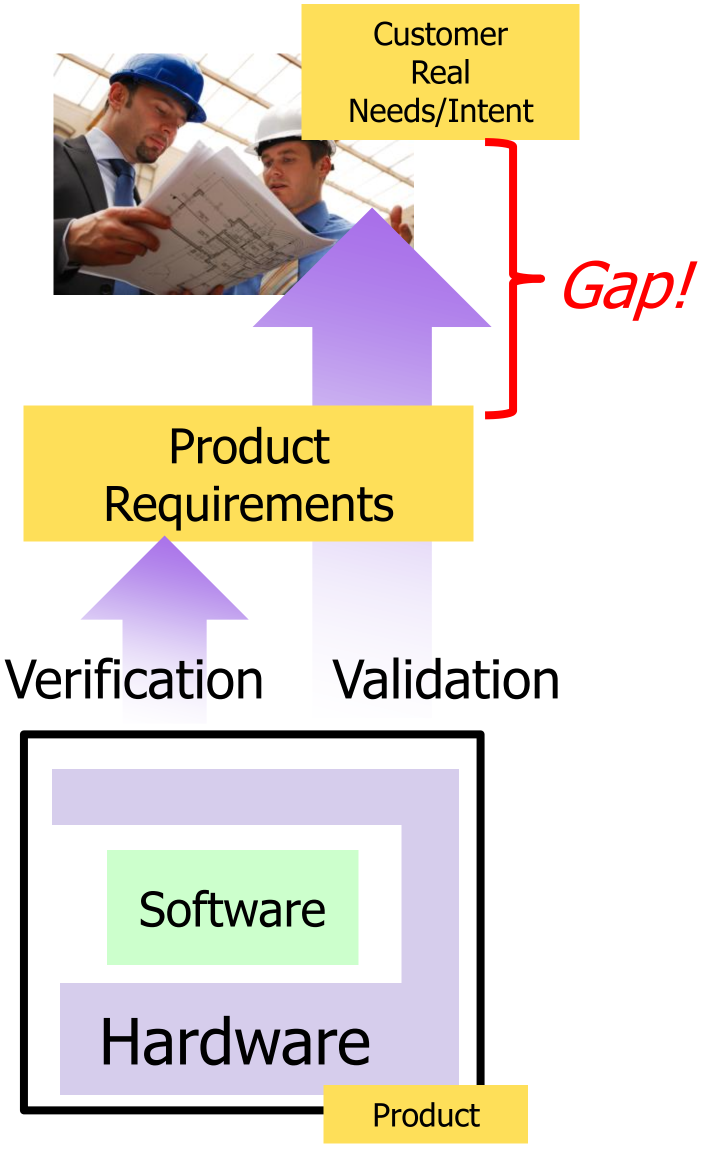
Integration of reasoning about fault propagation with formal architecture descriptions.

Building Competencies

Challenge: Current competence-building practices, including the educational infrastructure and curricula, are not sufficient to engineer emerging safety-critical systems

- Market forces not effective in providing training resources, because these systems are a relatively small segment of the vast software engineering market
- Engineering safety-critical systems increasingly requires breadth as well as domain knowledge far beyond the delivery-capability of the current educational infrastructure
- Management in many development organizations…
  - Is not aware of competence needed
  - Has little appreciate for the dangers introduced by the growing complexity inherent in software based systems

SCC – Requirements with Intent



…most deficiencies and disappointments in safety-critical systems occur because the requirements specification does accurately reflect needs and intent for that system.

Software Development Ecosphere

Goals

Assumptions

Agreement, Shared View ?

Design

Domain Knowledge

Importance of Requirements

Where do errors come from?

Errors made during the requirements stage account for 40 to 60 percent of all defects found in a software project

-- *Davis 1993; Leffingwell 1997*.

Importance of Requirements

What's the hardest part of building a software system?

The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify.

-- Fred Brooks, in "No Silver Bullet: Essence and Accidents of Software Engineering".

Stakeholders in Conventional Software Engineering

Nowhere more than in the requirements process do the interests of all the *stakeholders* in a software or system project intersect

- *Customers* -- fund a project or acquire a product to satisfy their organization's business objectives.
- *Users* -- interact directly or indirectly with the product (a subclass of customers).
- *Requirements analysts* -- write the requirements and communicate them to the development community.
- *Developers* -- design, implement, and maintain the product.
- *Testers* -- determine whether the product behaves as intended.
- *Documentation writers* -- produce user manuals, training materials, and help systems.
- *Manufacturing people* – must build the products that contain software.
- *Sales, marketing, field support, help desk, etc.* -- who will have to work with the product and its customers.

Wiegers, Karl (2009). "Software Requirements" (Chapter 1)

Additional Stakeholders in Safety-Critical Systems

Regulatory / Certification Regimes

FDA

U.S. Department of Health and Human Services
Food and Drug Administration

FEDERAL AVIATION ADMINISTRATION

Third-Party Certification Agents

Standards

Requirements

Software

Hardware

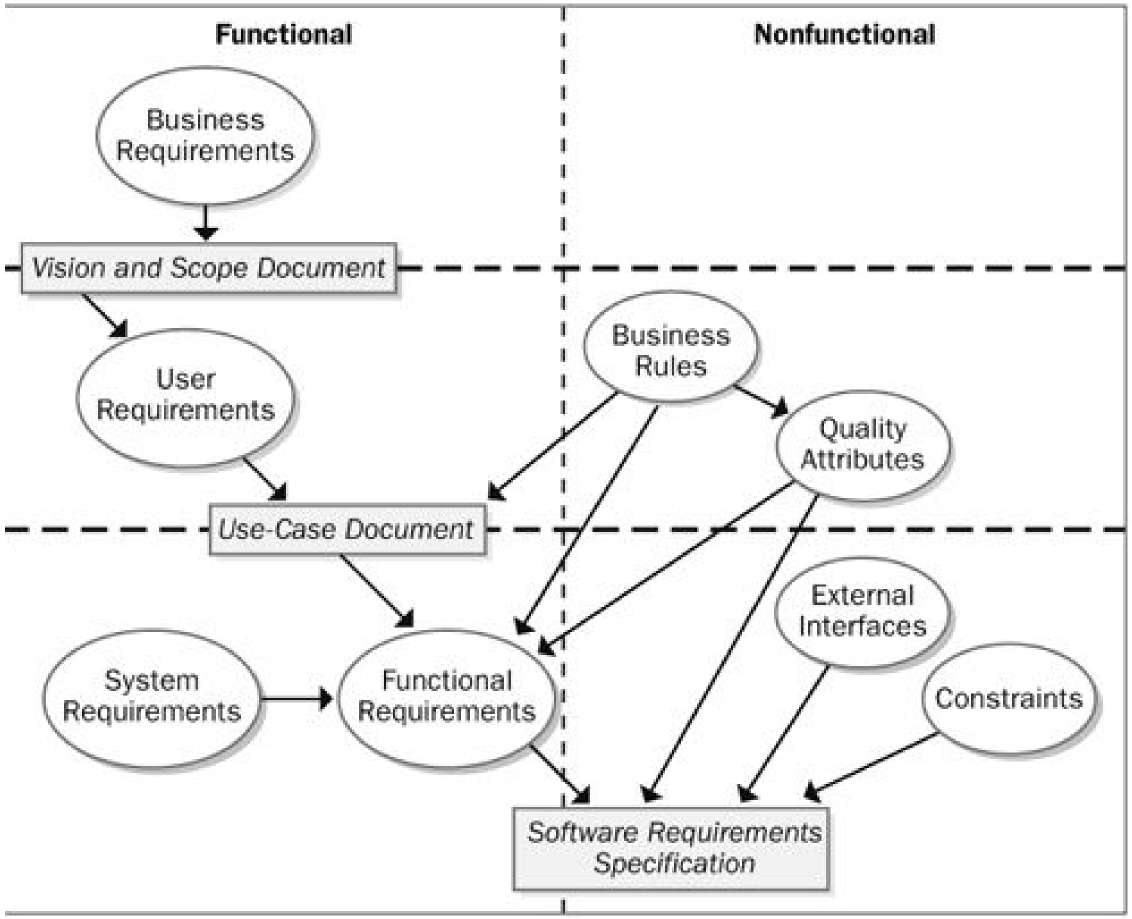
Product

Requirements

Society at Large

Kinds of Requirements

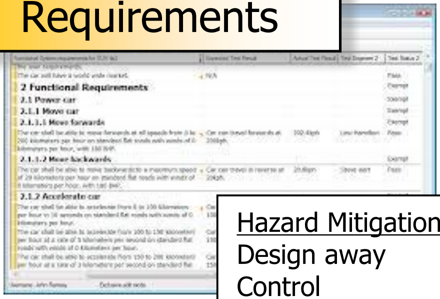
Relating different kinds of requirements



Wiegers, Karl (2009). "Software Requirements" (Chapter 1)

Adding Safety Requirements

System Safety Requirements



2 Functional Requirements

2.1 Power car

2.1.1 Move car

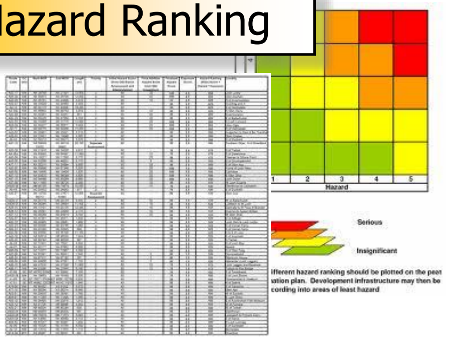
2.1.1.1 Move forwards

2.1.1.2 Move backwards

2.1.2 Accelerate car

<u>Hazard Mitigation</u>
Design away
Control
Alert

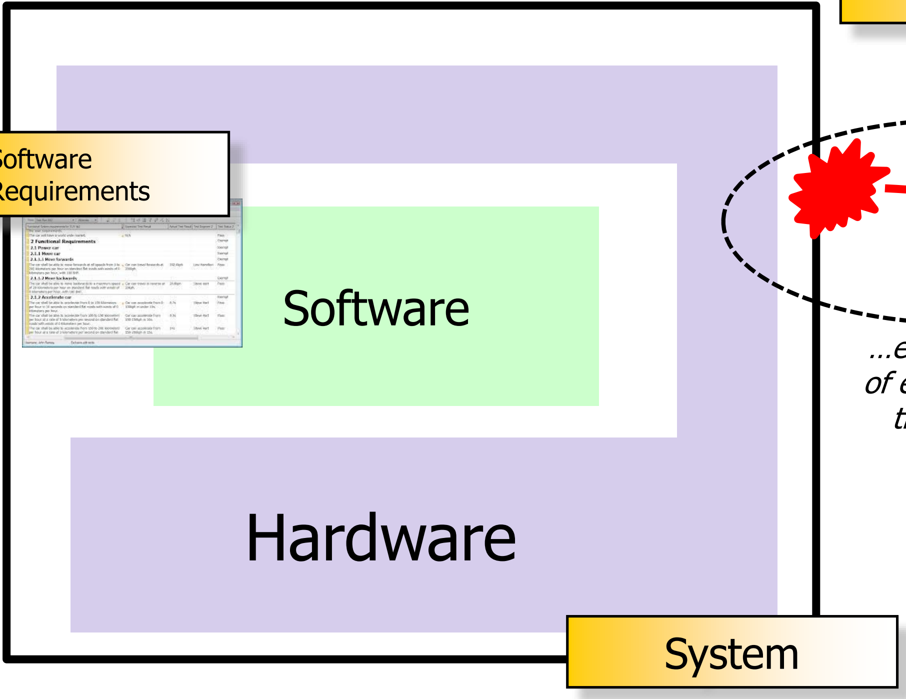
Hazard Ranking

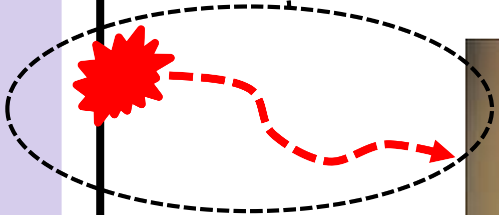


Hazard
Serious
Insignificant

Notions of Loss / Harm

Hazard



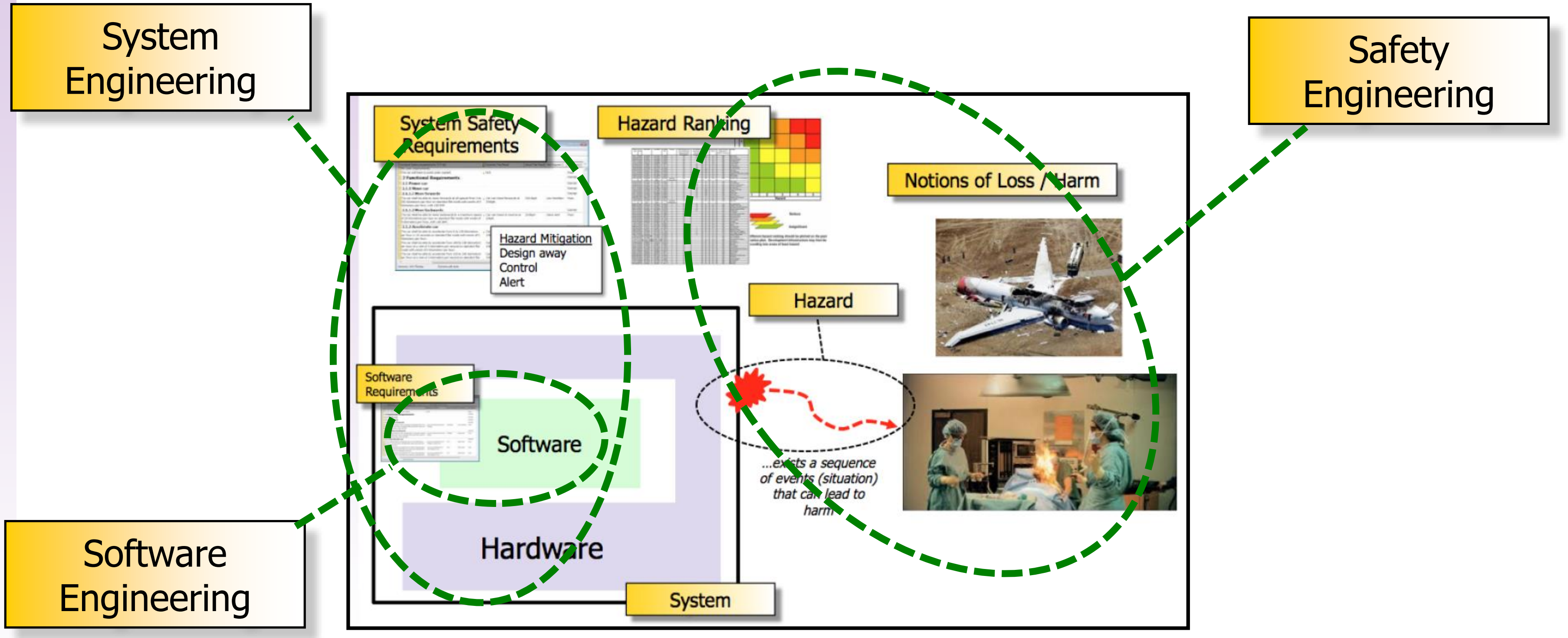


...exists a sequence of events (situation) that can lead to harm

Challenge: Constructing safety requirements requires a number of additional concepts

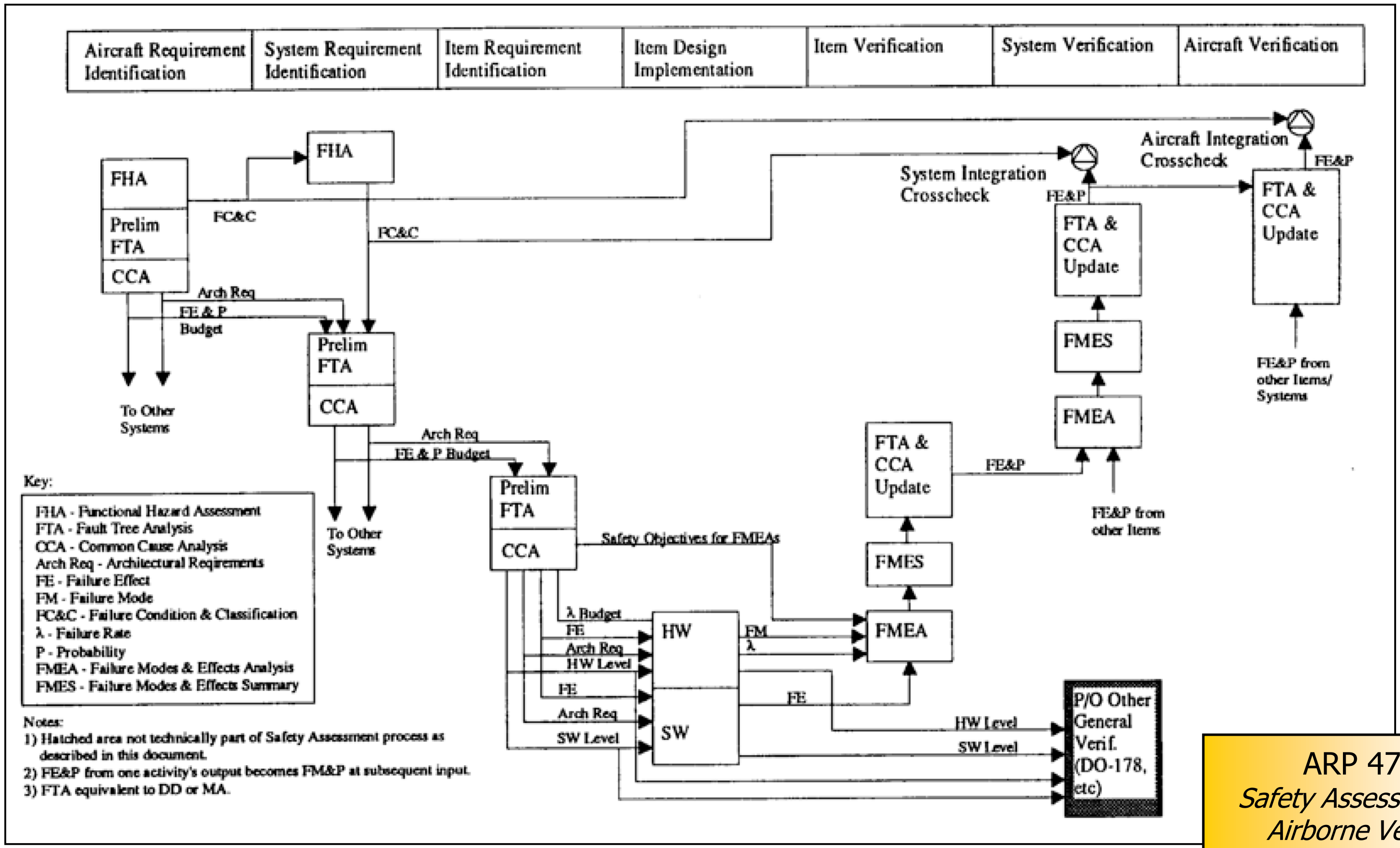
Partitioned Tasking

Challenge: Activities in this space are often partitioned into different disciplines and engineering roles, and incomplete understanding and information flow contributes to gaps between requirements and customer intent/safety



Safety Assessment / Risk Management

Challenge: Safety analysis, system engineering, etc. includes a number of new concepts, techniques, etc. How much of this does a software engineer on a safety-critical system need to know?



Aircraft Requirement Identification
System Requirement Identification
Item Requirement Identification
Item Design Implementation
Item Verification
System Verification
Aircraft Verification
FHA
Prelim FTA
CCA
FC&C
Arch Req
FE & P Budget
To Other Systems
Prelim FTA
CCA
System Integration Crosscheck
Aircraft Integration Crosscheck
FE&P
FTA & CCA Update
FMES
FMEA
FE&P from other Items/ Systems
FE&P from other Items
Safety Objectives for FMEAs
λ Budget
FE
HW Level
SW Level
HW
SW
FM
λ
P/O Other General Verif. (DO-178, etc)

Key:
- FHA - Functional Hazard Assessment
- FTA - Fault Tree Analysis
- CCA - Common Cause Analysis
- Arch Req - Architectural Requirements
- FE - Failure Effect
- FM - Failure Mode
- FC&C - Failure Condition & Classification
- λ - Failure Rate
- P - Probability
- FMEA - Failure Modes & Effects Analysis
- FMES - Failure Modes & Effects Summary

Notes:
1) Hatched area not technically part of Safety Assessment process as described in this document.
2) FE&P from one activity's output becomes FM&P at subsequent input.
3) FTA equivalent to DD or MA.

ARP 4761
Safety Assessment for Airborne Vehicles

Importance of Experience / Domain Knowledge

Challenge: Even if engineers are well-versed in techniques and processes, will there still be difficulties in closing the gap?

"Indeed what evidence there is suggests that the most significant factor in achieving low hazardous failure rates is domain knowledge" –

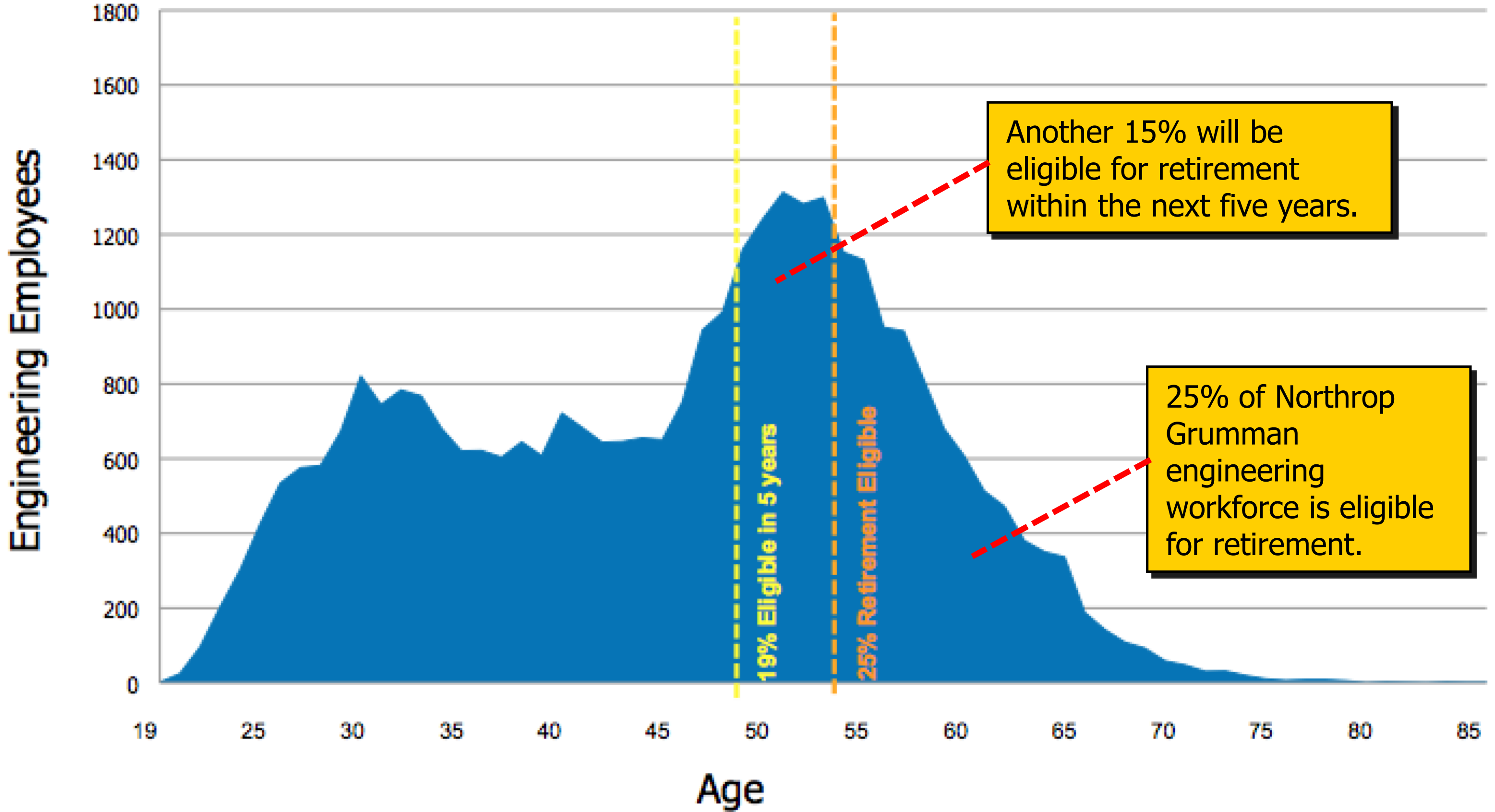
McDermid – *Software Safety: Where's the Evidence?*

Example Workforce Demographics

Challenge: In the safety-critical industry, a lot of the "experience resource" may be retiring soon. There is a new to properly train and integrate new generations

2012 Engineering Demographics

NORTHROP GRUMMAN



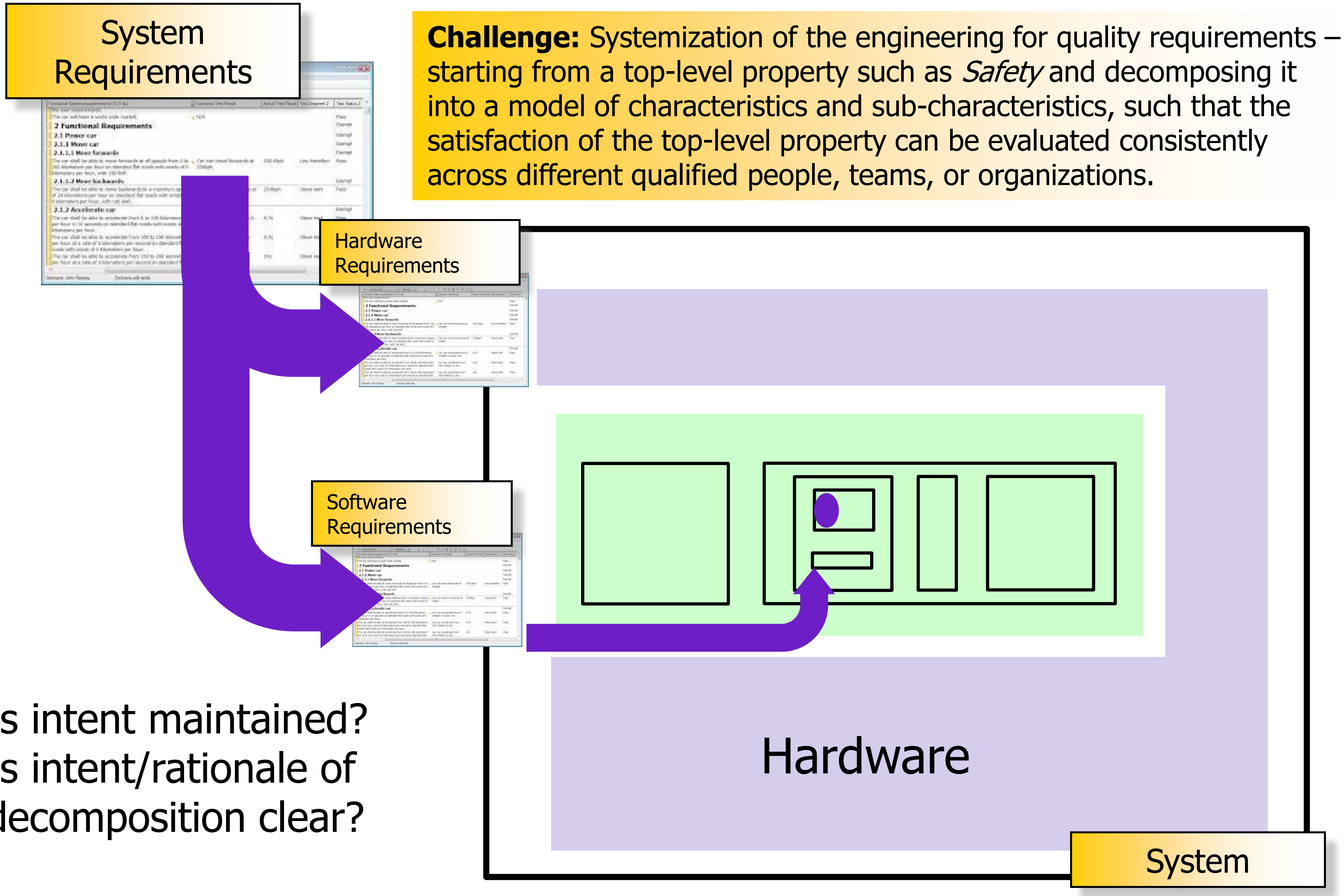
Safety / Systems / Software Engineering

Summarizing…

- The processes of system engineering, safety engineering, and software engineering are not well-integrated
  - Isolation starts with the *competence-building infrastructure*
  - …continues through *vocabularies of discourse*, *paradigms*, and *tools*
- Gaps across safety engineering, systems engineering, and software engineering need to be bridged
- Gaps across domain experts, system developers, software developers, and safety experts should be bridged

Decomposition Creates Gaps

Challenge: Systemization of the engineering for quality requirements – starting from a top-level property such as *Safety* and decomposing it into a model of characteristics and sub-characteristics, such that the satisfaction of the top-level property can be evaluated consistently across different qualified people, teams, or organizations.

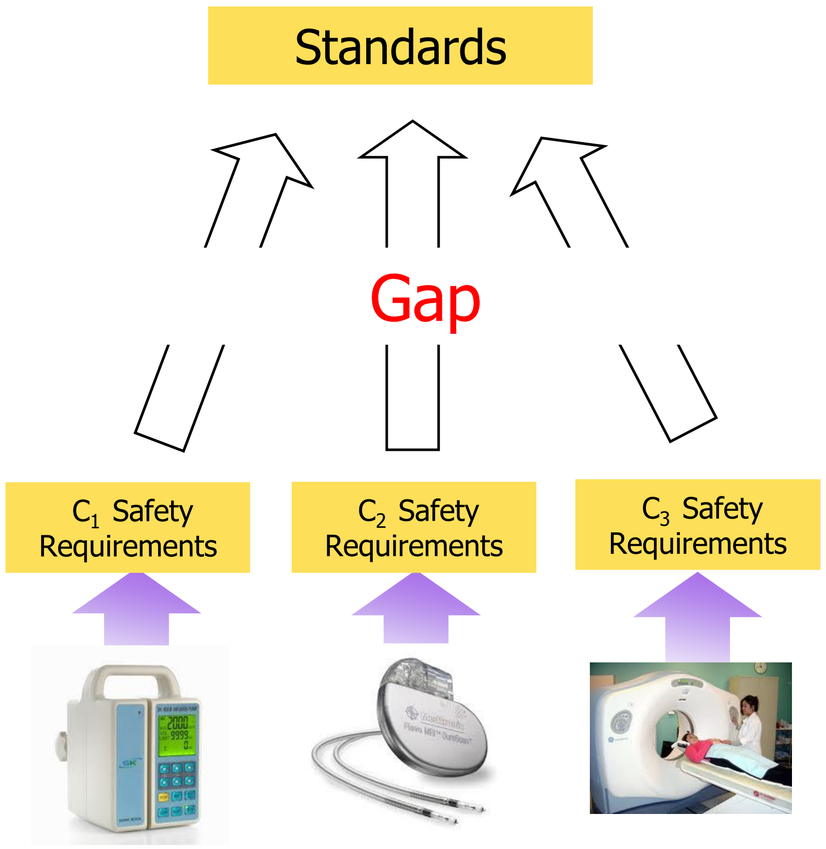


Is intent maintained?
Is intent/rationale of decomposition clear?

Nature of Criteria

- We desire that a standard state *criteria* that when applied to system C will result in C being safe
- However, a standard is not meant to be applied to a single system C but a family of systems $C_1, \ldots, C_n$.
- Each of the C_i may vary in their notions of loss, hazards, safety requirements, etc.

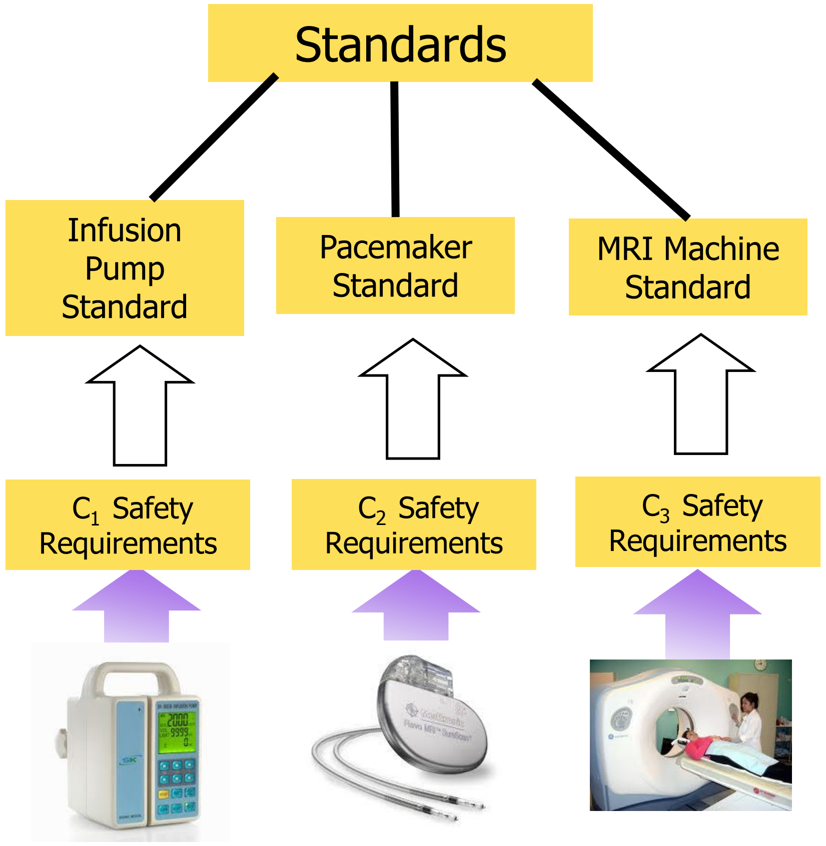
Challenge: There must always be a *gap* and/or *degree of indirection* between criteria leading to safety that a standard can specify and safety requirements of particular products. What is the most effective kind of criteria for a standard to state?


Standards
Gap
C_1 Safety Requirements
C_2 Safety Requirements
C_3 Safety Requirements

Nature of Criteria

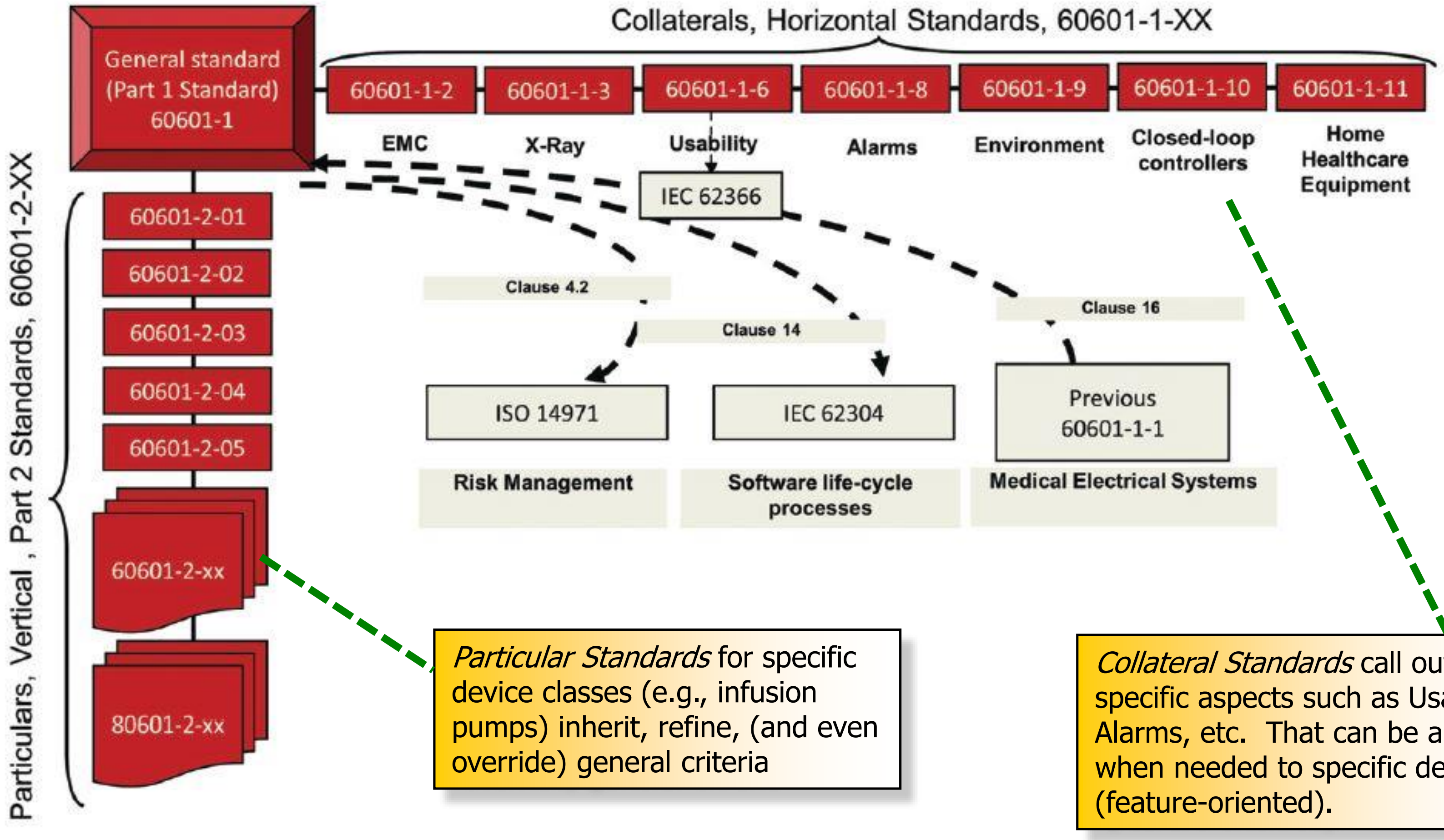
- One approach to reducing the gap being the *general* criteria of a standard and the safety requirements of a particular product is *standard refinement*
- Refined standards identify criteria for *specific classes* of devices
- But there is still generality, e.g., the Infusion Pump standard must apply to pumps from many different manufacturers
    - Thus, there are still gaps…

Challenge: Can standards better apply product-line and feature-ordered/aspect-oriented approaches to hone in on safety requirements for particular systems?


Standards
Infusion Pump Standard
Pacemaker Standard
MRI Machine Standard
C_1 Safety Requirements
C_2 Safety Requirements
C_3 Safety Requirements

Example of Standard Refinement

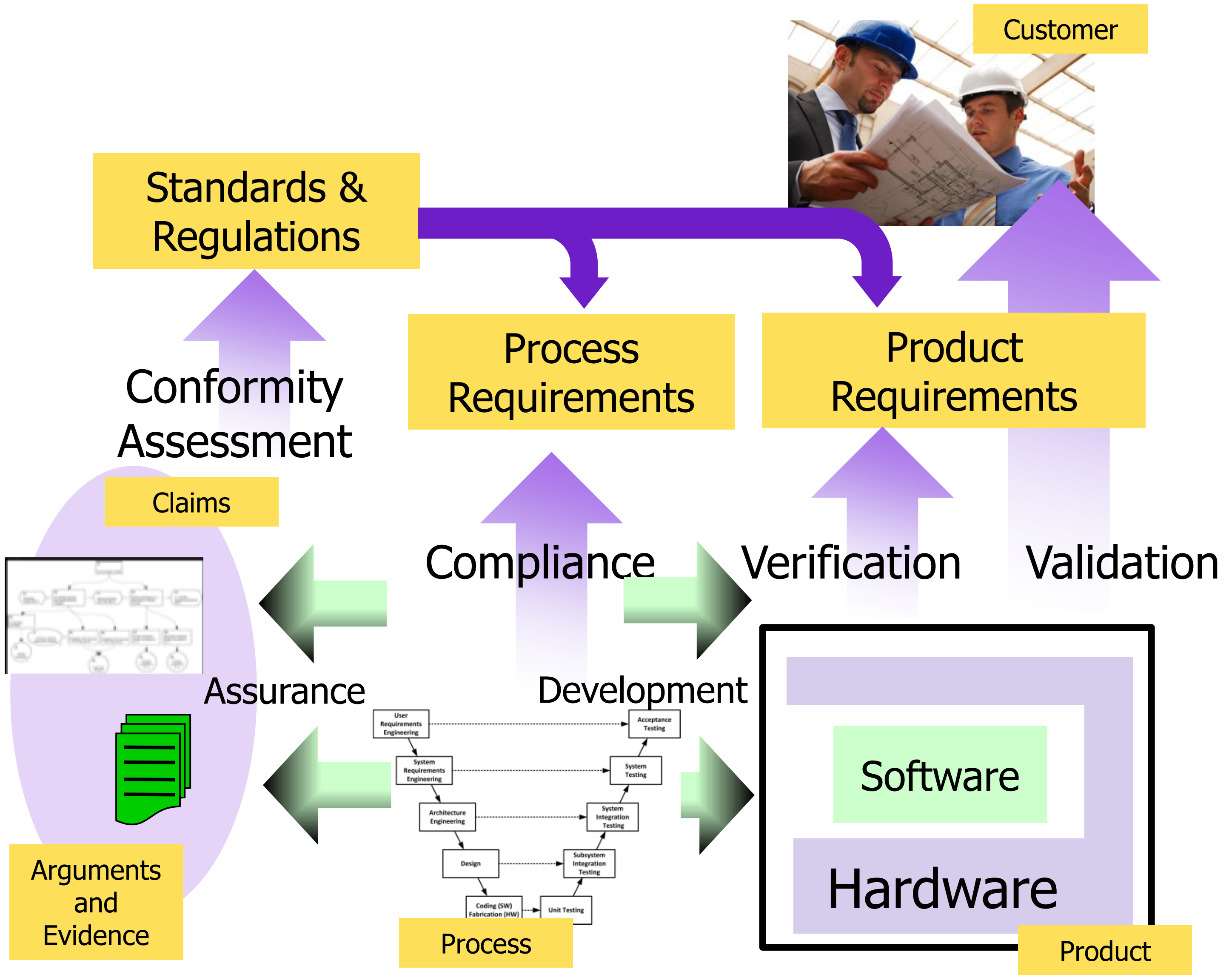
The 60601 family of medical device safety standards provides an example of standard refinement/aspect that allow more general requirements to be tailored to particular contexts…



Particular Standards for specific device classes (e.g., infusion pumps) inherit, refine, (and even override) general criteria

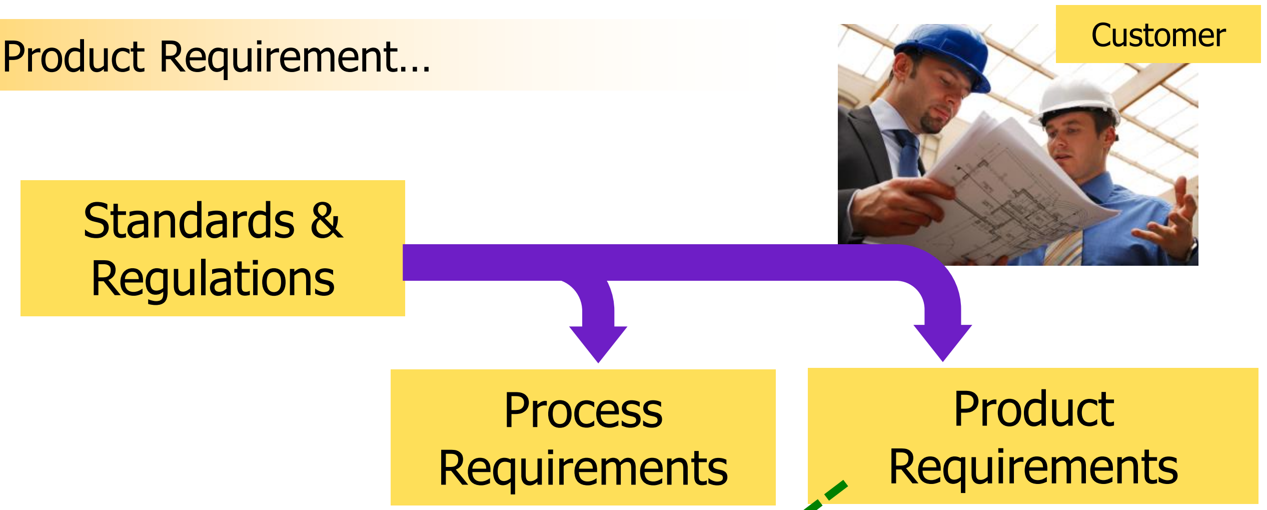
Collateral Standards call out specific aspects such as Usability, Alarms, etc. That can be applied when needed to specific devices (feature-oriented).

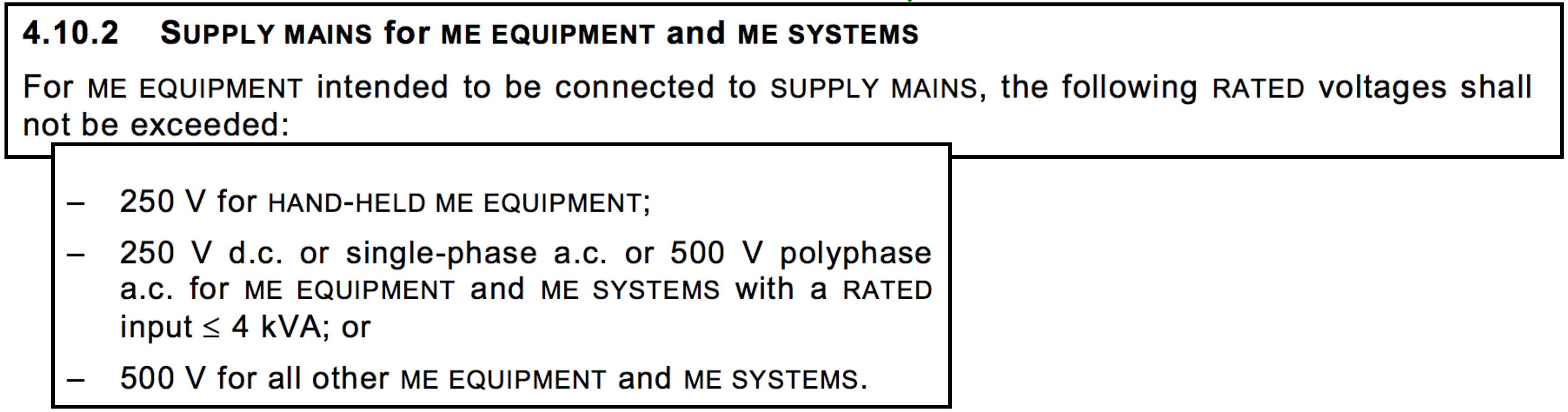
Nature of Criteria



Nature of Criteria

60601 Example of Product Requirement…





4.10.2 SUPPLY MAINS for ME EQUIPMENT and ME SYSTEMS

For ME EQUIPMENT intended to be connected to SUPPLY MAINS, the following RATED voltages shall not be exceeded:

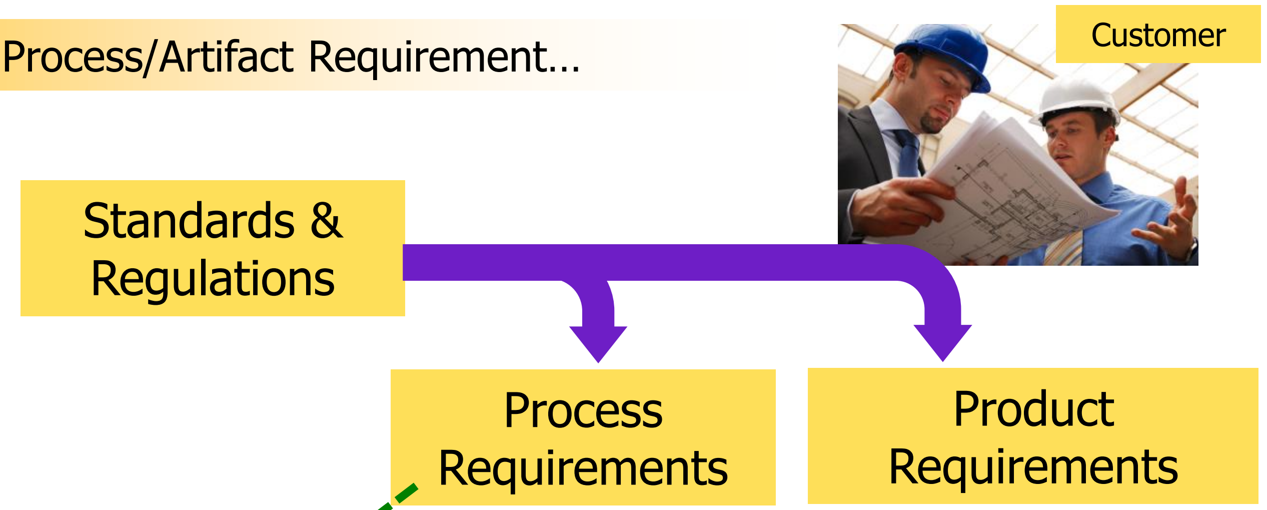
– 250 V for HAND-HELD ME EQUIPMENT;

– 250 V d.c. or single-phase a.c. or 500 V polyphase a.c. for ME EQUIPMENT and ME SYSTEMS with a RATED input ≤ 4 kVA; or

– 500 V for all other ME EQUIPMENT and ME SYSTEMS.

Nature of Criteria

14971 Example of Process/Artifact Requirement…



4.3 Identification of hazards

The manufacturer shall compile documentation on known and foreseeable hazards associated with the medical device in both normal and fault conditions.

This documentation shall be maintained in the risk management file.

Compliance is checked by inspection of the risk management file.

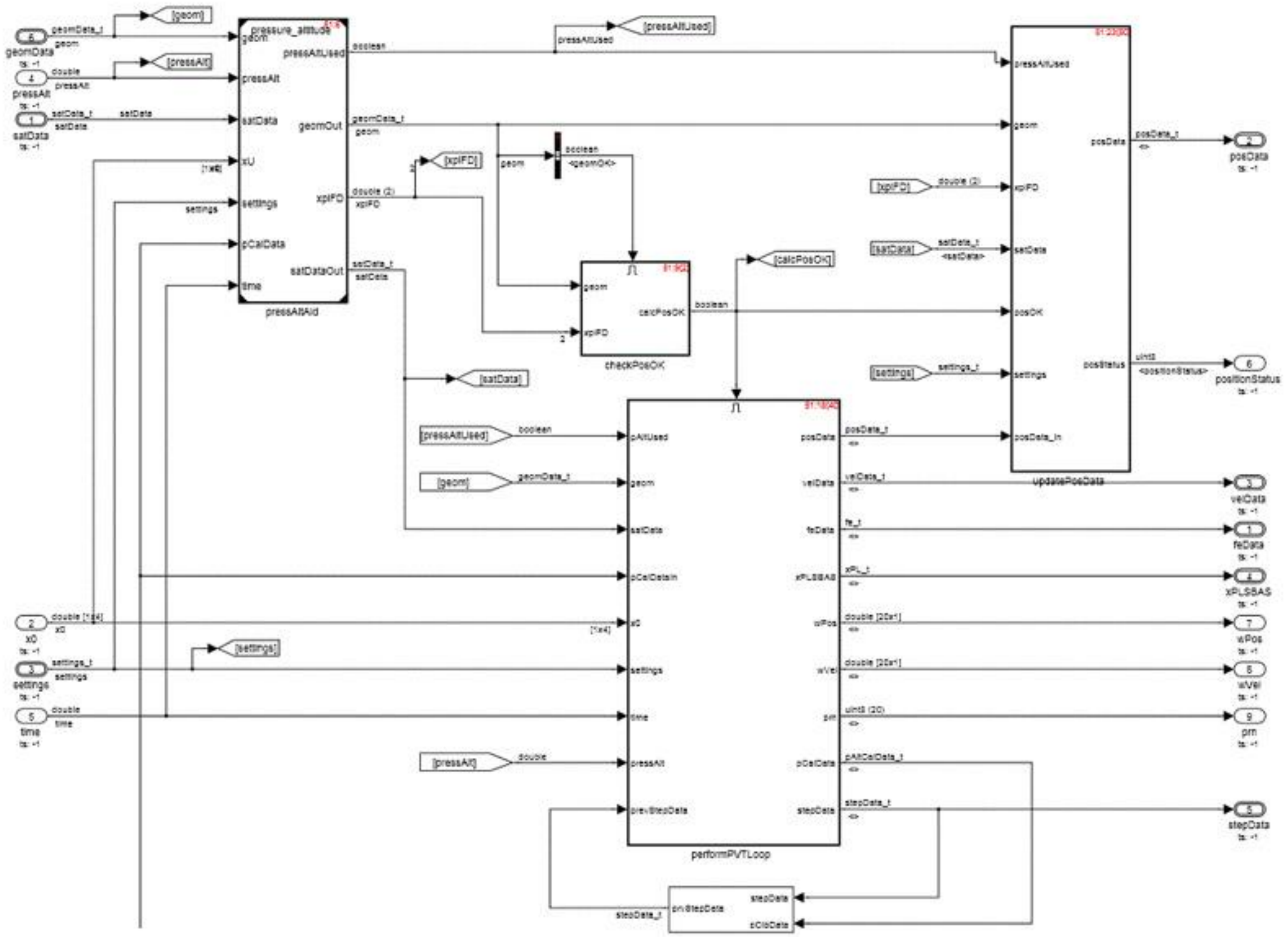
No criteria for evaluating the goodness of this process step!

Completeness Issues

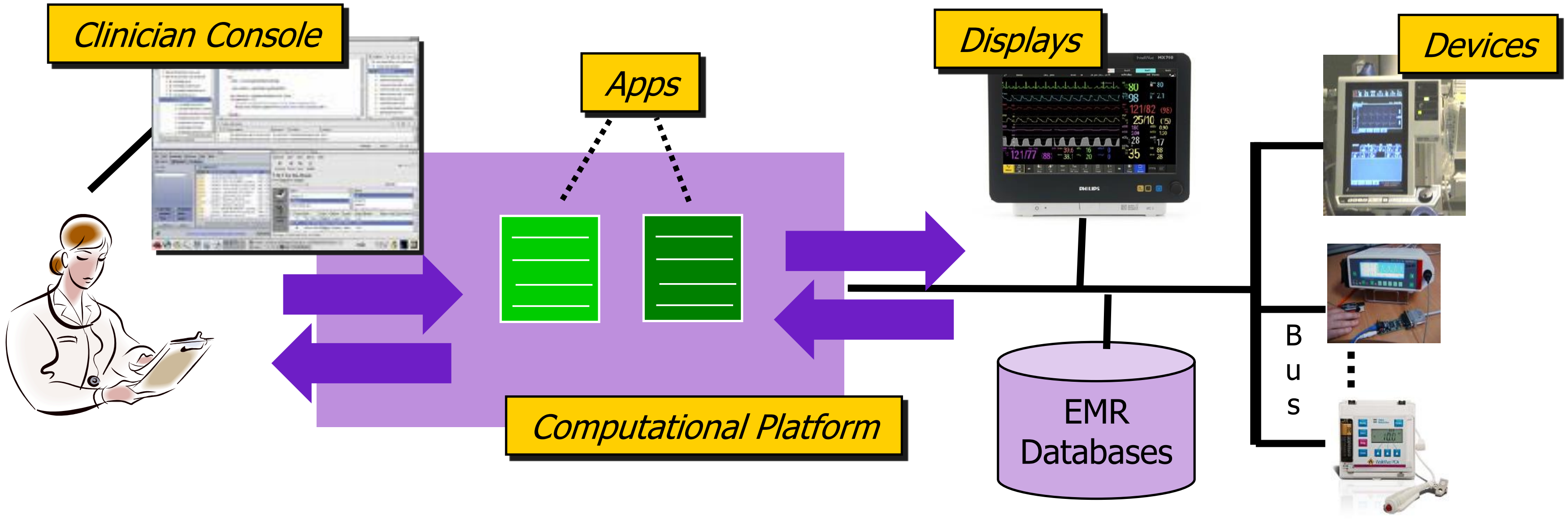
- Identification of *all* hazards
- …with *all* associated safety and security
- Identification of *all* necessary interfaces to system, etc.

Graphic-based Designs as Requirements

Should graphic-based designs be considered as requirements? Are we really capturing intent?



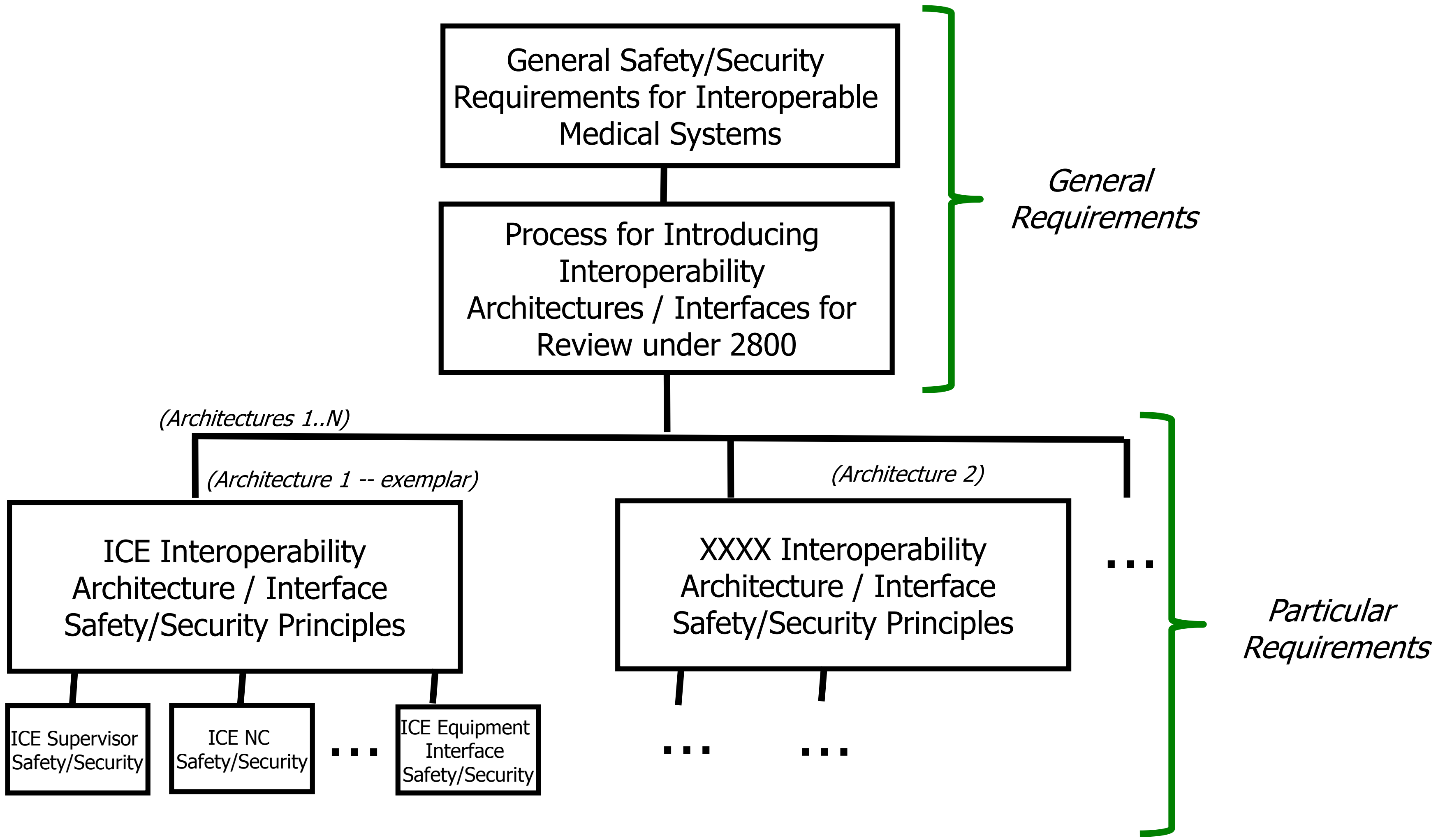
Medical Application Platforms



- A *Medical Application Platform* is a safety- and security-critical real-time computing platform for…
  - Integrating heterogeneous devices, medical IT systems, and information displays via communications infrastructure, and
  - Hosting applications ("apps") that provide medical utility via the ability to acquire information from and update/control integrated devices, IT systems, and displays

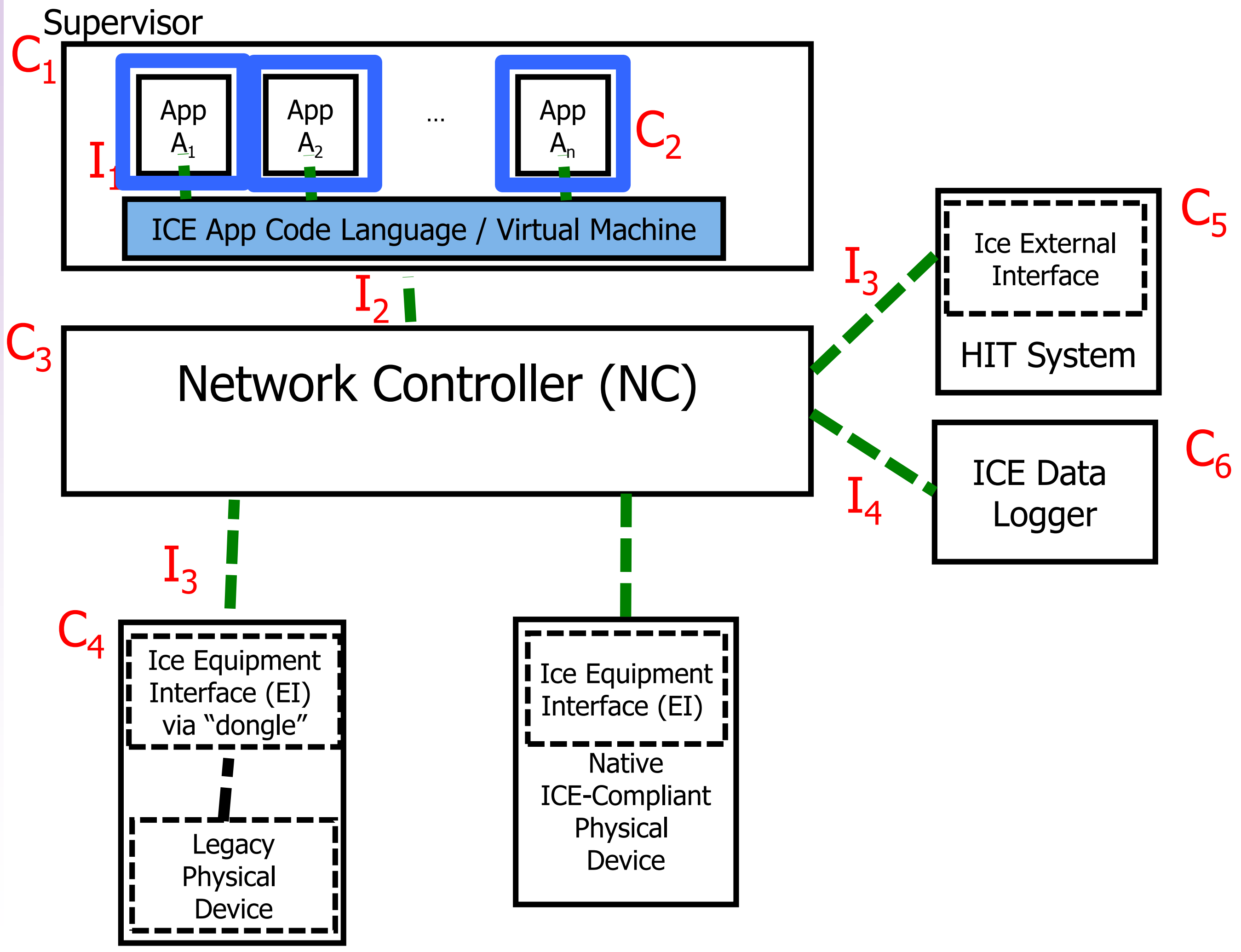
Possible Structure for 2800

To achieve the goals of being architecture neutral (allowing for multiple architectures to emerge) while being specific enough to support interoperability safety/security,… we are proposing the following structure for organizing horizontal requirements (and 2800 family of standards, in general)…


General Safety/Security Requirements for Interoperable Medical Systems
Process for Introducing Interoperability Architectures / Interfaces for Review under 2800
General Requirements
(Architectures 1..N)
(Architecture 1 -- exemplar)
(Architecture 2)
ICE Interoperability Architecture / Interface Safety/Security Principles
XXXX Interoperability Architecture / Interface Safety/Security Principles
…
Particular Requirements
ICE Supervisor Safety/Security
ICE NC Safety/Security
…
ICE Equipment Interface Safety/Security
…
…

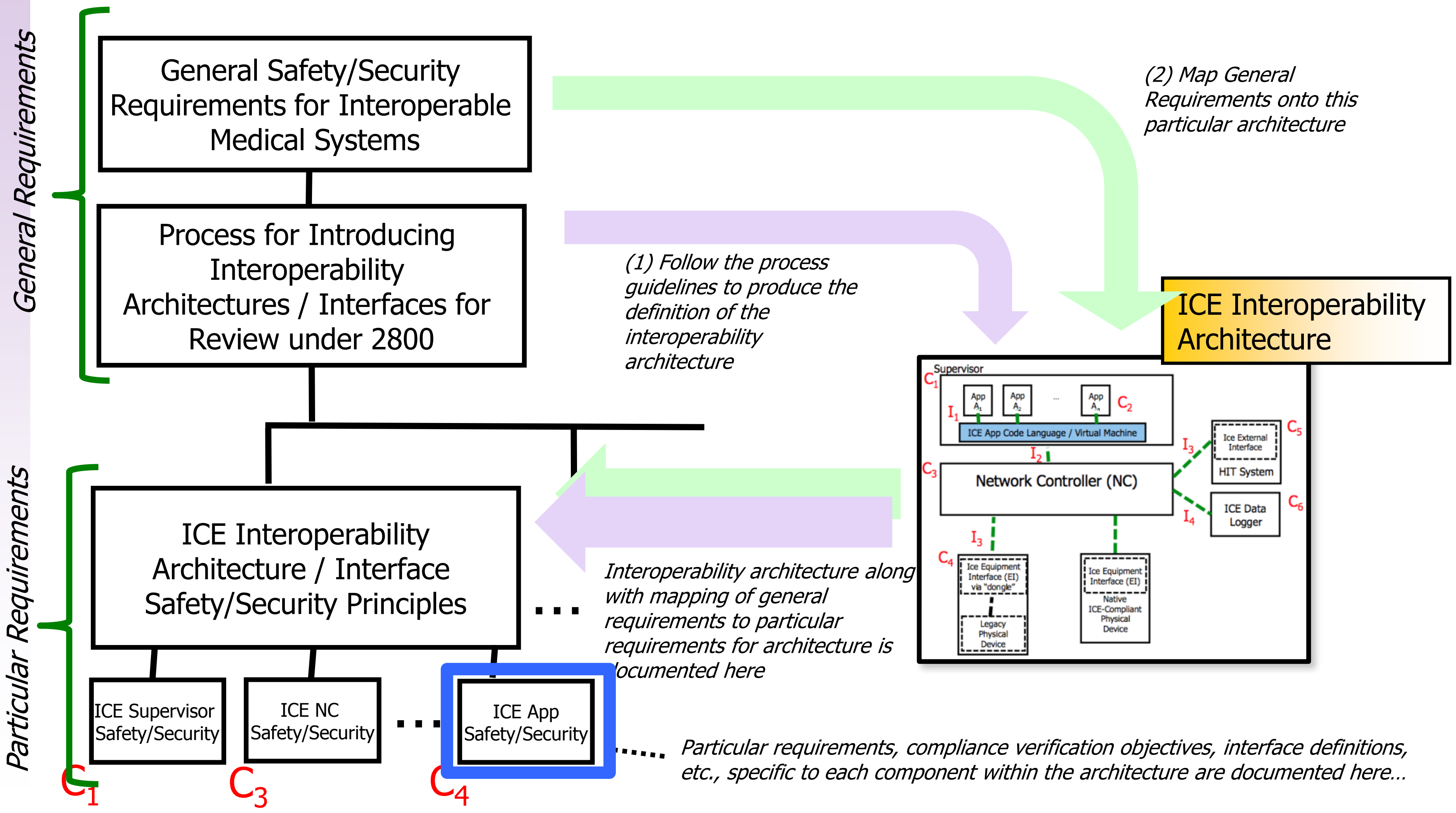
Interoperability Architecture

Defining an interoperability architecture (Step 2) – identify interoperability points and interfaces. Define functional and non-functional (real-time, safety, security) properties of interfaces



Interoperability Architecture

Below we summarize how the definition of a particular interoperability architecture gives rise to the structuring of Particular Requirements for that architecture…


General Requirements
General Safety/Security Requirements for Interoperable Medical Systems
Process for Introducing Interoperability Architectures / Interfaces for Review under 2800
(2) Map General Requirements onto this particular architecture
(1) Follow the process guidelines to produce the definition of the interoperability architecture
ICE Interoperability Architecture
Supervisor
C_1
App A_1
App A_2
...
App A_n
C_2
I_1
ICE App Code Language / Virtual Machine
I_2
C_3
Network Controller (NC)
I_3
Ice External Interface
HIT System
C_5
I_4
ICE Data Logger
C_6
I_3
C_4
Ice Equipment Interface (EI) via "dongle"
Legacy Physical Device
Ice Equipment Interface (EI)
Native ICE-Compliant Physical Device
Particular Requirements
ICE Interoperability Architecture / Interface Safety/Security Principles
. . .
Interoperability architecture along with mapping of general requirements to particular requirements for architecture is documented here
ICE Supervisor Safety/Security
ICE NC Safety/Security
. . .
ICE App Safety/Security
Particular requirements, compliance verification objectives, interface definitions, etc., specific to each component within the architecture are documented here…
C_1
C_3
C_4

Summary

- Requirements development continues to be a key concern…
  - …but not just system/software requirements, also requirements within standards (and getting those to be as effective as possible)
  - Need to better bridge safety/system/software engineering
  - Decomposition of quality properties
  - Better "mathematization" of requirements to support more automated analyses
- Education often seems to be at the heart of our problems