

## BEYOND THE PAIN OF TRACEABILITY:

Cost Effective Techniques for demonstrating Safety, Security, and Compliance of Software Intensive Systems

**Dr. Jane Cleland-Huang**

Center of Excellence for Software Traceability  
DePaul University

Some of the work described in this talk is  
funded by the US National Science

Foundation under Grants CCF-0959924 and  
CCF-1265178.

DEPAUL  
UNIVERSITY



CoEST  
Center of Excellence for Software Traceability



# What is Traceability?

The ability to identify and document the lineage of each requirement, including its derivation (backward traceability), its allocation (forward traceability), and its relationship to other requirements.

*International Institute of Business Analysis  
Body of Knowledge  
Version 2.0*



Traceability is of particular concern in safety & mission-critical systems.



# Standards require it..



The **Federal Aviation Administration's** (FAA) DO-178B standard specifies that at each and every stage of development “software developers must be able to demonstrate traceability of designs against requirements.

The U.S. **Food and Drug Administration** (FDA) states that traceability analysis must be used to verify that the software design implements all of the specified software requirements, that all aspects of the design are traceable to software requirements, and that all code is linked to established specifications and test procedures.

**Process improvement initiatives** such as CMMI also require traceability.



# Important in both Software & Systems Projects



Strategic Traceability is needed across both systems models and software models.

Regulatory codes

Requirements

Software Control  
Systems

Mechanical models

Design & Test Cases

Electrical models



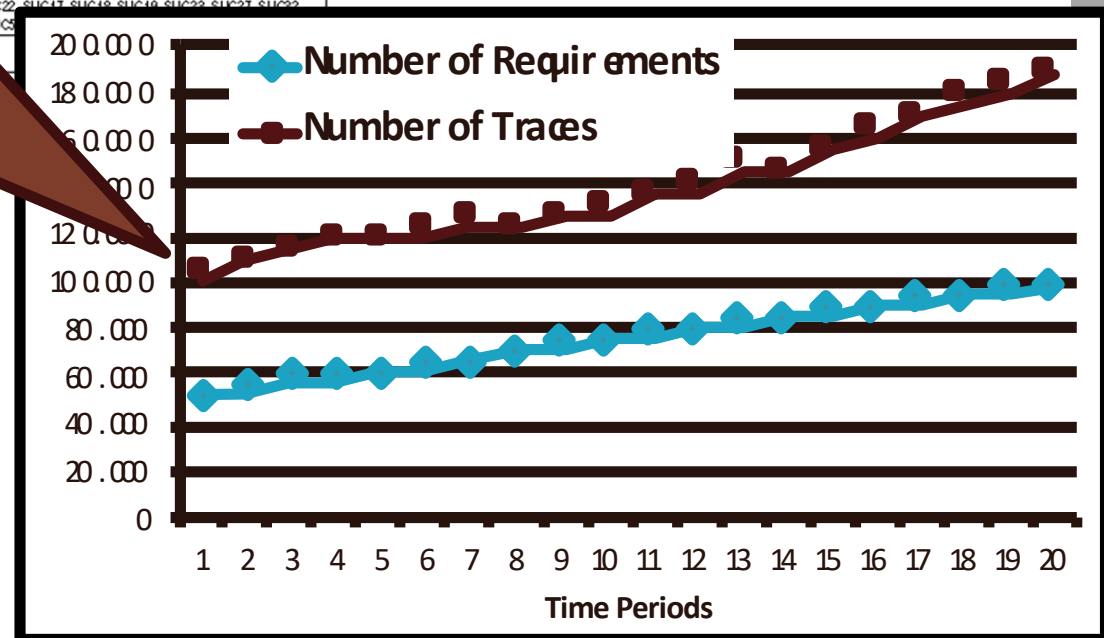
...but

Too much tracing!!

Excessive numbers of traceability links deteriorate into an unwieldy, inaccurate, tangle of relationships.

Tag	Name	Traced-to
SUC137		BUC17, BUC18, BUC22, BUC201
SUC138		BUC17, BUC18, BUC22, BUC201
SUC139		BUC17, BUC18, BUC22, BUC201
SUC140		BUC17, BUC18, BUC22, BUC201
SUC141		BUC17, BUC18, BUC22, BUC201
SUC142		BUC17, BUC18, BUC22, BUC201
SUC143		BUC17, BUC18, BUC22, BUC201
SUC144		BUC17, BUC18, BUC22, BUC201
SUC145		BUC17, BUC18, BUC22, BUC201
SUC146		BUC17, BUC18, BUC22, BUC201
SUC147		BUC17, BUC18, BUC22, BUC201
SUC148		BUC17, BUC18, BUC22, BUC201
SUC149		BUC17, BUC18, BUC22, BUC201
SUC150		BUC17, BUC18, BUC22, BUC201
SUC151		BUC17, BUC18, BUC22, BUC201
SUC152		BUC17, BUC18, BUC22, BUC201
SUC153		BUC17, BUC18, BUC22, BUC201
SUC154		BUC17, BUC18, BUC22, BUC201
SUC155		BUC17, BUC18, BUC22, BUC201
SUC156		BUC17, BUC18, BUC22, BUC201
SUC157		BUC17, BUC18, BUC22, BUC201
SUC158		BUC17, BUC18, BUC22, BUC201
SUC159		BUC17, BUC18, BUC22, BUC201
SUC160		BUC17, BUC18, BUC22, BUC201
SUC161		BUC17, BUC18, BUC22, BUC201
SUC162		BUC17, BUC18, BUC22, BUC201
SUC163		BUC17, BUC18, BUC22, BUC201
SUC164		BUC17, BUC18, BUC22, BUC201
SUC165		BUC17, BUC18, BUC22, BUC201
SUC166		BUC17, BUC18, BUC22, BUC201
SUC167		BUC17, BUC18, BUC22, BUC201
SUC168		BUC17, BUC18, BUC22, BUC201
SUC169		BUC17, BUC18, BUC22, BUC201
SUC170		BUC17, BUC18, BUC22, BUC201
SUC171		BUC17, BUC18, BUC22, BUC201
SUC172		BUC17, BUC18, BUC22, BUC201
SUC173		BUC17, BUC18, BUC22, BUC201
SUC174		BUC17, BUC18, BUC22, BUC201
SUC175		BUC17, BUC18, BUC22, BUC201
SUC176		BUC17, BUC18, BUC22, BUC201
SUC177		BUC17, BUC18, BUC22, BUC201
SUC178		BUC17, BUC18, BUC22, BUC201
SUC179		BUC17, BUC18, BUC22, BUC201
SUC180		BUC17, BUC18, BUC22, BUC201
SUC181		BUC17, BUC18, BUC22, BUC201
SUC182		BUC17, BUC18, BUC22, BUC201
SUC183		BUC17, BUC18, BUC22, BUC201
SUC184		BUC17, BUC18, BUC22, BUC201
SUC185		BUC17, BUC18, BUC22, BUC201
SUC186		BUC17, BUC18, BUC22, BUC201
SUC187		BUC17, BUC18, BUC22, BUC201
SUC188		BUC17, BUC18, BUC22, BUC201
SUC189		BUC17, BUC18, BUC22, BUC201
SUC190		BUC17, BUC18, BUC22, BUC201
SUC191		BUC17, BUC18, BUC22, BUC201
SUC192		BUC17, BUC18, BUC22, BUC201
SUC193		BUC17, BUC18, BUC22, BUC201
SUC194		BUC17, BUC18, BUC22, BUC201
SUC195		BUC17, BUC18, BUC22, BUC201
SUC196		BUC17, BUC18, BUC22, BUC201
SUC197		BUC17, BUC18, BUC22, BUC201
SUC198		BUC17, BUC18, BUC22, BUC201
SUC199		BUC17, BUC18, BUC22, BUC201
SUC200		BUC17, BUC18, BUC22, BUC201

The number of requirements and other artifacts grow quickly as the project progresses.





# How will traces be created and used?



## Trace Creators & Maintainers:

Requirements engineers,  
developers, architects.

Who? When? How? Why? What?

## Trace users:

Compliance officers, requirements  
engineers, testers, developers.

Who? When? How? Why? What?



As creators and maintainers are often NOT the same people,  
there needs to be buy-in at the project level!



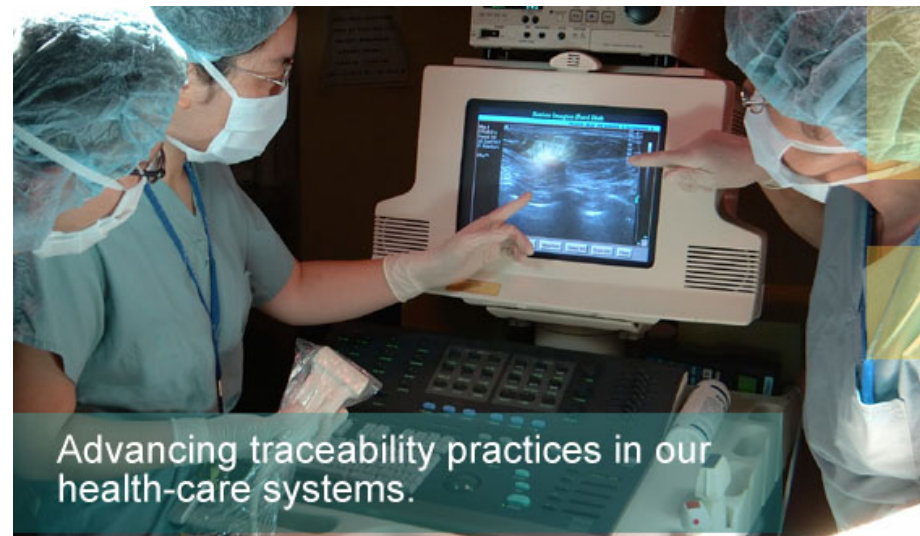
# How this talk is structured

- Introduction to Traceability ✓
- Benefits of Traceability
  - Trace Example: Safety Analysis
  - Trace Example: Architectural Preservation
  - Trace Example: Regulatory Compliance
- Mistakes People Make
- Remedies
  - Creating and Maintaining Trace Links
  - Using Trace Links
  - Comprehending Trace Results
- Next Steps



[illegible]

- 
- Supporting regulatory compliance in safety critical software systems.





# A Brief Look at 3 Tracing Scenarios



Safety  
Analysis

John is the safety engineer. He needs to demonstrate that all identified hazards have been addressed in the design, and ultimately in the deployed system



Architectural  
Preservation

Les is a Software Architect. He needs to ensure that architectural knowledge and the integrity of the software is preserved despite maintenance and change.



Change  
Impact

Mary is a requirements engineer. She needs to understand the impact of a change upon regulatory codes. Will we be in compliance if we make this change?

# How this talk is structured

- Introduction to Traceability ✓
- Strategic ROI Driven Traceability ✓
  - Trace for a purpose
  - Trace slices
  - Tracing Architectural Concerns
- Mistakes People Make
- Remedies
  - Creating and Maintaining Trace Links
  - Using Trace Links
  - Comprehending Trace Results
- Next Steps



# Tracing Mistakes People Make

FOCUS: SAFETY-CRITICAL SOFTWARE

## Strategic Traceability for Safety-Critical Projects

Patrick Möser, Ilmenau Technical University

Paul L. Jones and Yi Zhang, US Food and Drug Administration

Jane Cleland-Huang, Delft University

// An evaluation of traceability information for 10 submissions prepared by manufacturers for review at the US Food and Drug Administration identifies weaknesses spread traceability problems that affected regulators' ability to evaluate products safety in a timely manner. //



**FAILURE OF SAFETY-CRITICAL** software systems to operate correctly can cause serious harm to the public—consider devices such as pacemakers, nuclear power systems, and train signals, all of which rely on safety-critical software. Therefore, teams building safety-critical software products must perform rigorous risk analyses to identify potentially unsafe conditions and their contributing factors. Many projects conduct this process using techniques

such as failure modes and effects analysis, fault tree analysis, and hazard and operability studies. The risk analysis produces a set of system-level requirements specifically designed to mitigate or eliminate faults and reduce the likelihood of accidents.<sup>1</sup> These requirements relate to a broad range of factors including training, testing, process improvements, hardware, human factors, and software design constraints.

In this article, we focus on

traceability's role in establishing evidence that device specifications and implementations address identified hazards and their risk control measures (see the "Traceability Standards in Safety-Critical Projects" sidebar).<sup>2</sup> Creating and maintaining trace links can be an arduous, error-prone, and costly process that can have a significant effect on the overall costs and time-to-market for a product.<sup>3-7</sup> Traceability practices, therefore, need to be strategically planned and carefully implemented to provide cost-effective support for evaluating and demonstrating a specific system's safety and security.<sup>8</sup> When traceability isn't implemented strategically, individual stakeholders might create traces that they personally consider to be important or attempt to provide complete trace coverage without considering how the resulting trace links will be used. A trace-link approach to traceability has been shown in practice to be difficult to implement, almost impossible to maintain, and not particularly helpful for providing evidence that a system or device is safe for its intended use.

We present six practices for strategic traceability, derived from our own observations of effective traceability in industrial projects and supported by current literature.<sup>9-14</sup> We also identify nine recurring problems, each of which reduces the effectiveness of traceability verification efforts and increases the difficulty experienced by regulators in evaluating product safety. All the observations in this article are based on actual observations, but the illustrative examples are either fictitious or built on obfuscated data.

### Effective Practices for Tracing in Safety-Critical Projects

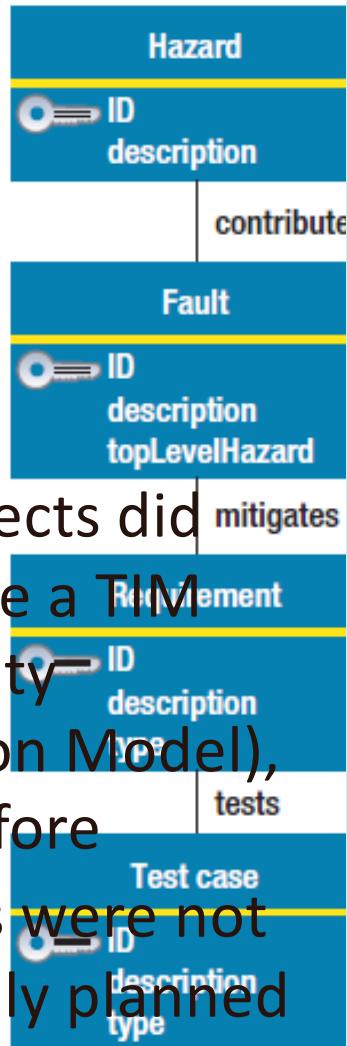
Although all the cases reported in this article are safety-critical in nature, many of the problems that we discuss are also applicable to software and

Our observations are based upon over a decade of engagements in industrial projects, and a study of the traceability components of Medical Device submissions to the FDA.

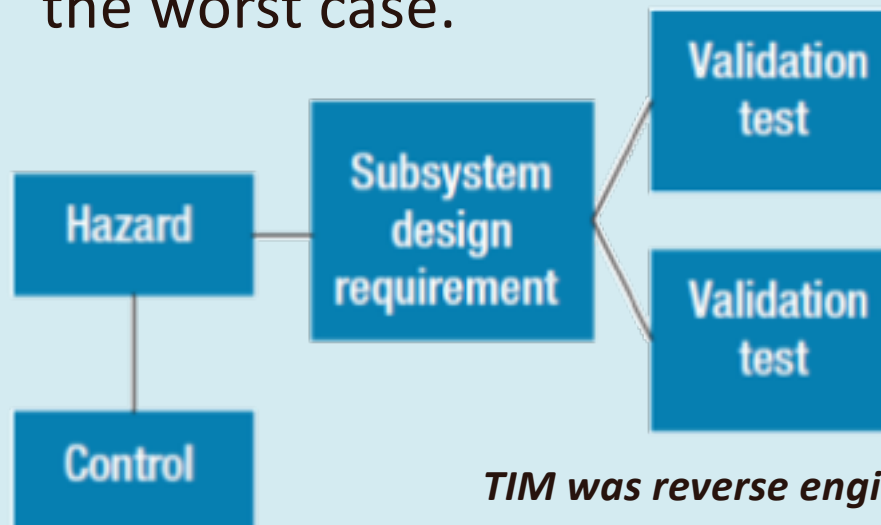


# Mistake # 1: Failure to plan

Most projects did not include a TIM (Traceability Information Model), and therefore TraceLinks were not strategically planned and/or represented.

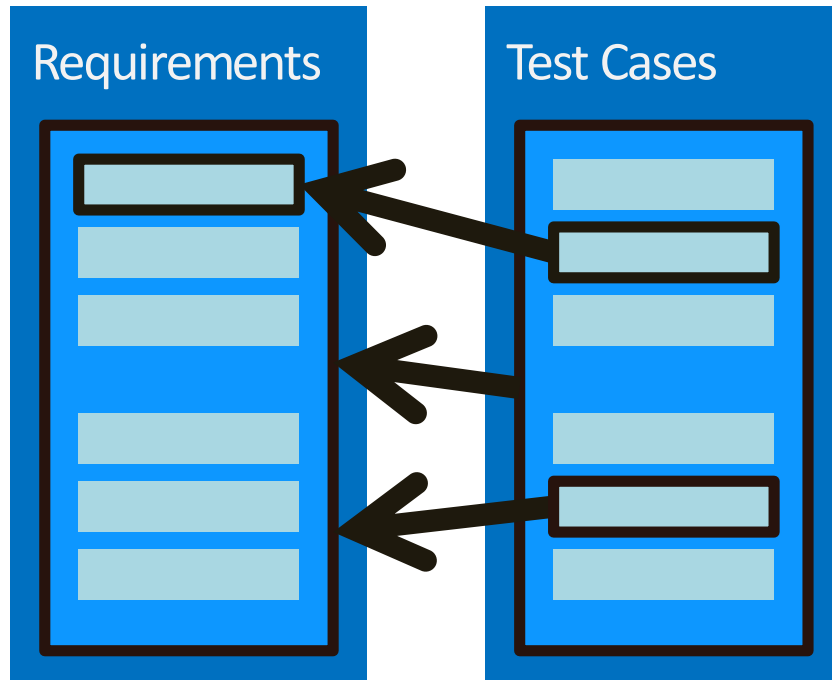


1. Difficult to understand/use the trace links.
2. In this example, links were created without careful planning. Faults were not included in the trace paths. Hazards were traced directly up to 15 subsystem design requirements in the worst case.



*TIM was reverse engineered from Trace matrices.*

## Mistake 2: Trace granularity not clearly defined.



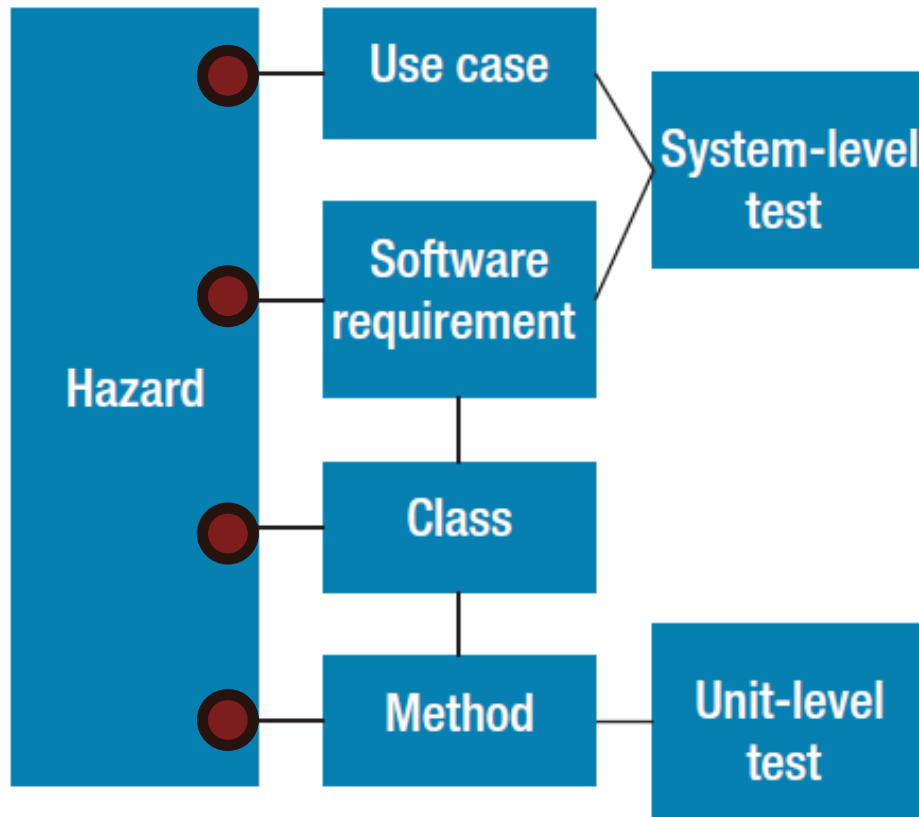
Fine grained (source) to coarse grained (target) links don't provide enough information to evaluate whether the target artifacts are fully satisfied.

Ill-defined trace granularity leads to unacceptably high, unacceptably low, or mismatched links, making it difficult to determine whether hazards and faults are fully addressed.

Links can be:

- Too coarse grained
- Too fine-grained
- Inconsistently granulated

## Mistake 3: Redundant Traceability Paths.



Redundant paths lead to inconsistent data i.e. links between A and B traversed along path A->B->C may my different from links between A->X->C.

Not clear how incompatibilities should be resolved.

In this example the TIM shows redundant trace paths between sets of artifacts. Trace Links were created at multiple granularities.



## Mistake 4: Lack of Project-Wide Unique IDs

...  
During the [...], the timeout SHALL (5.5a) be set to 60 seconds. Upon completion of the [...],  
the default SHALL (5.5b) be set to 30 seconds.  
After the specified interval [...], the [...] SHALL (5.5c) turn off and [...].  
...

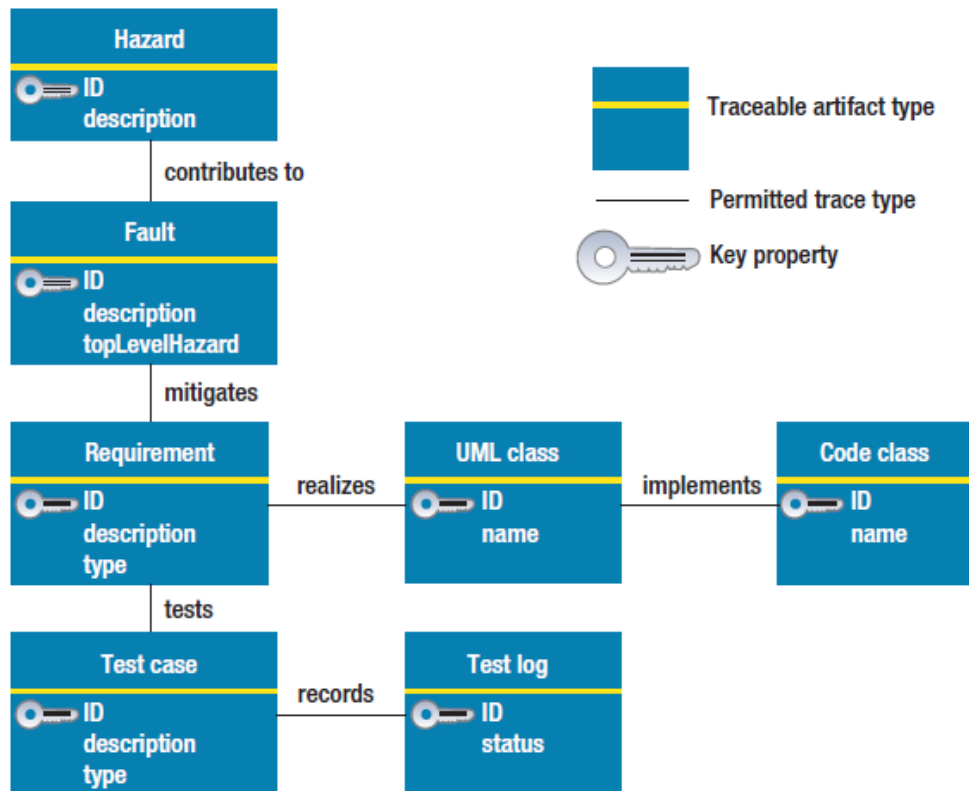
For traceability purposes, ID's should be

- ❑ Unique
- ❑ Represent primitive requirements
- ❑ Stable

In this example, the IDs are sequentially inserted for phrases in the text. i.e. hard to change and maintain in a way that preserves the above properties.



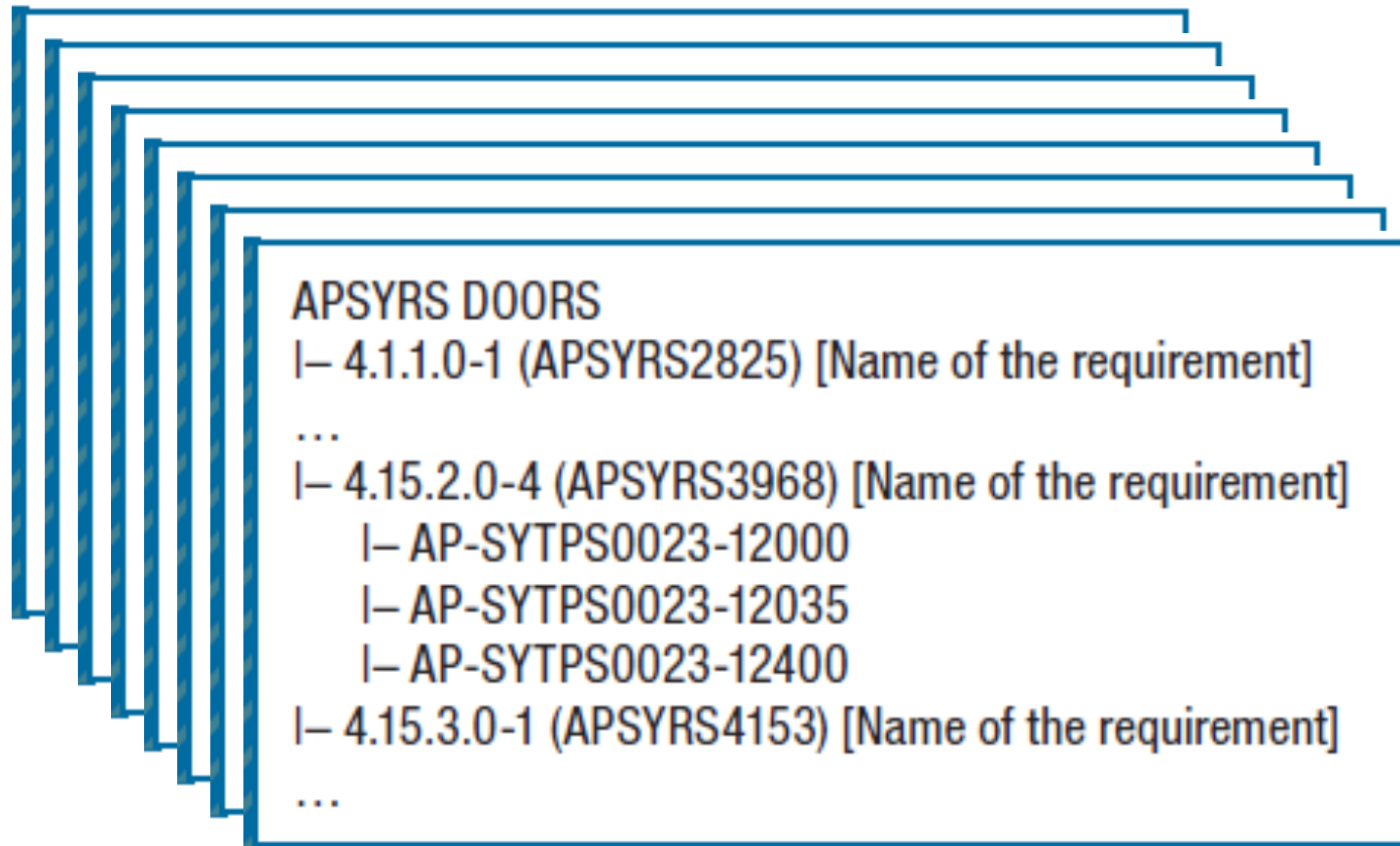
# Mistake 5: Important Links Missing



1. **Widows:**  
e.g. Mitigating requirements not traced from test cases.
2. **Orphans:**  
e.g. Test cases not traced to requirements.

Missing links means that certifiers/approvers are unable to determine whether a hazard has been fully addressed etc.

## Mistake 5: The “DOORS” dump



Pages...  
and  
pages..  
and  
pages...  
of links.

Such traceability ‘evidence’ submitted to approving agencies is difficult to process and ‘smacks’ of obfuscation. Instead of arranging information in massive tables.. Collate it into **trace slices**.



# Mistake 6: Traceability as an Afterthought

Trace Links that are created at the end of the development process for certification purposes often suffer from:

- ❑ Incorrectness/incompleteness
- ❑ Unavailability earlier in the project to support:
  - ❑ Impact analysis
  - ❑ Architectural preservation
  - ❑ Compliance validation



# How this talk is structured

- Introduction to Traceability ✓
- Strategic ROI Driven Traceability ✓
  - Trace for a purpose
  - Trace slices
  - Tracing Architectural Concerns
- Mistakes People Make ✓
- Remedies
  - Trace Query Languages
  - Commercial tools
  - Just in Time Tracing (JITT)
- Next Steps

