System failures and errors

4th December 2024

Dr Phillip Benachour, School of Computing and Communications

With thanks to Dr Emily Winter Dr Mark Rouncefield for access to 23-24 and 21-22 slides

Week 9 learning objectives

- To **identify** causes of system failure (through looking at some case studies)
- To **understand** different theories as to **why** system errors and failures occur
- To **consider** how systems can be made more dependable

Lancaster University

System errors and failures- case study 1

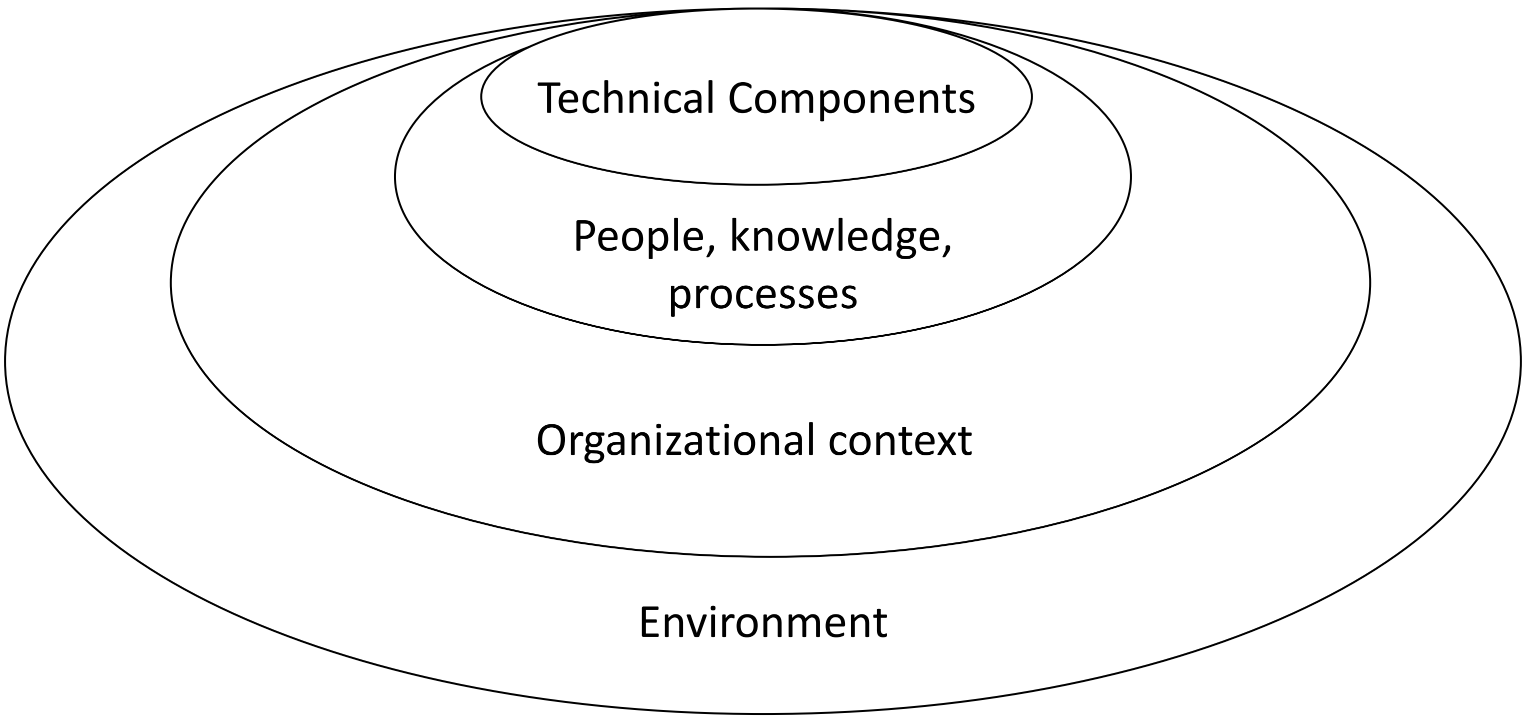
Titanic case study

- Catastrophic failure of a large system
- Very costly failure in terms of:
  - Money
  - Human life
  - Organisational reputation
- Many mistakes made during all phases of design and development

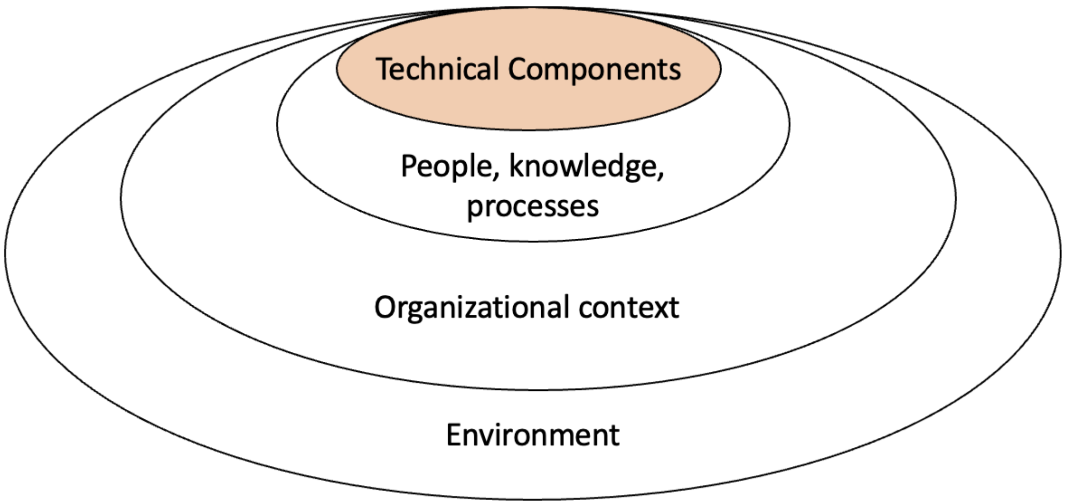
Nature of Titanic as a system

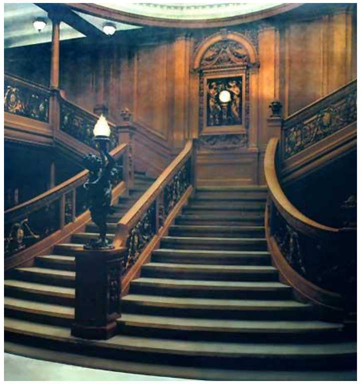
- Very complex socio-technical system
- Safety critical control systems
- Involved latest cutting-edge technology:
  - Data communications
  - Engineering technologies
- Complex management structures
- Complex political and organisational context

Understanding what went wrong: entire system perspective



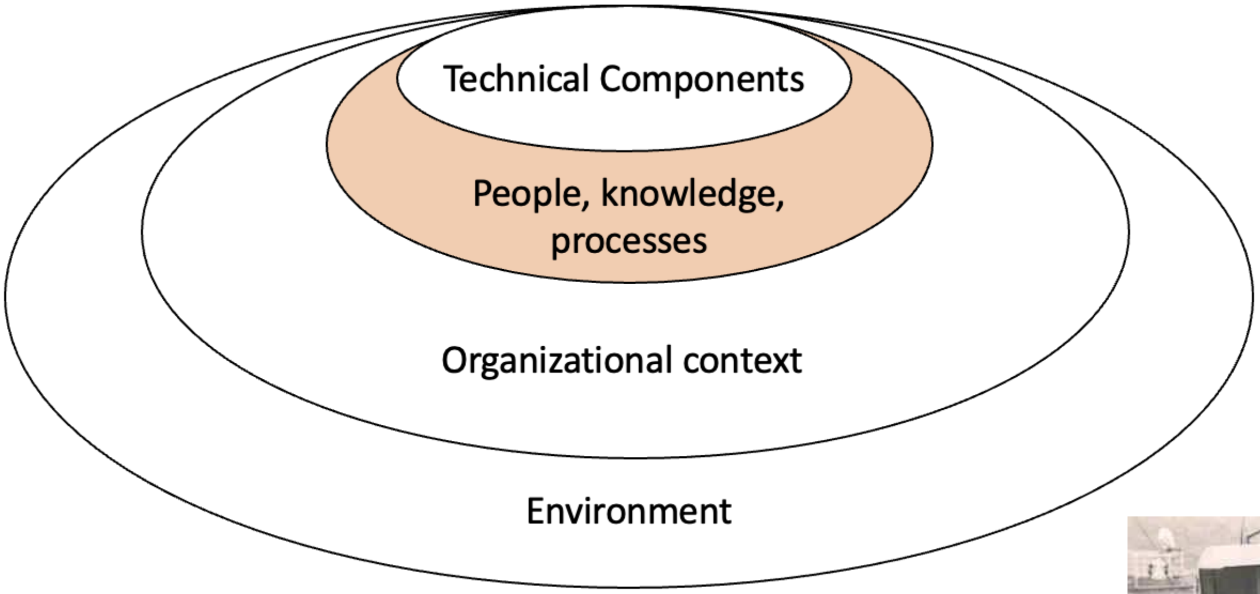
Technical components



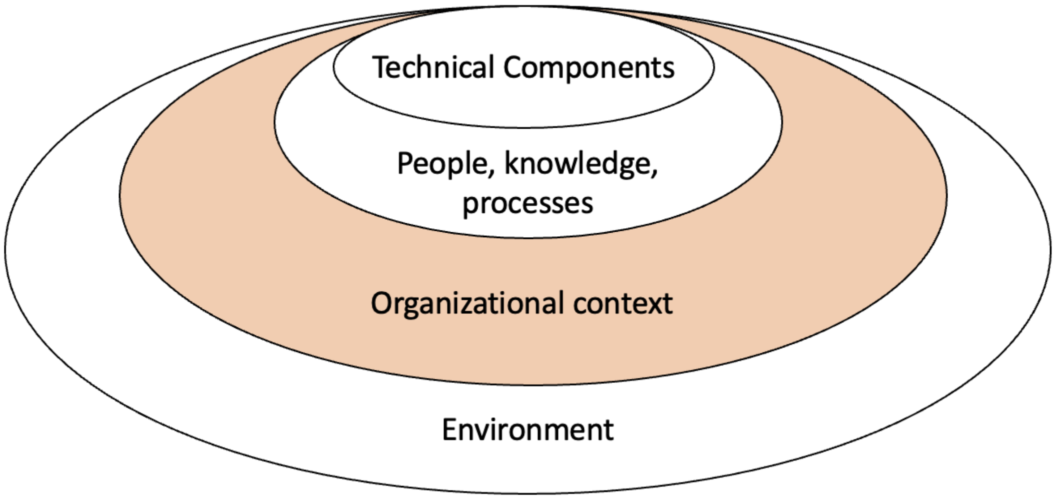



Chronicle / Michael Maloney

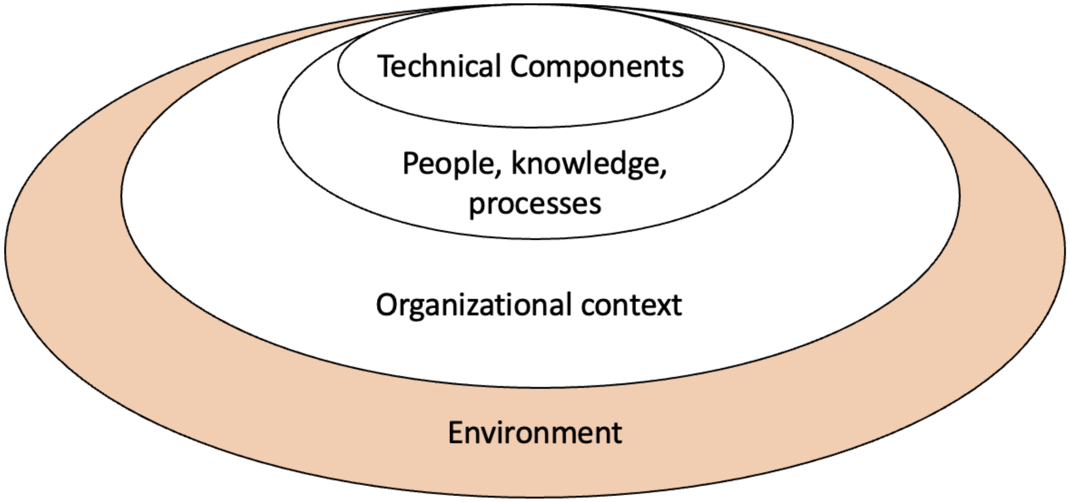
People, knowledge, processes

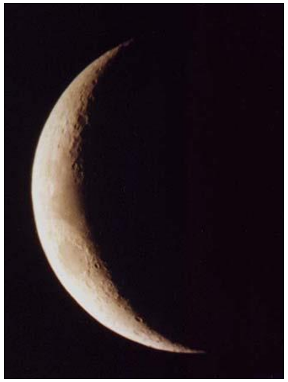


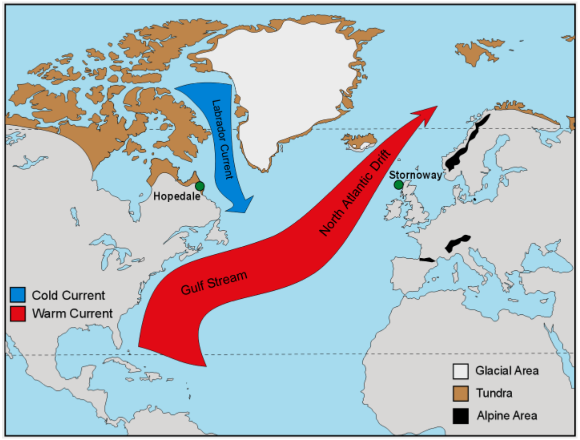
Organisational context



Environment







System errors and failures – case study 2

Post Office case study 'the most widespread miscarriage of justice in UK history' (BBC)

Lancaster University

Post Office case study

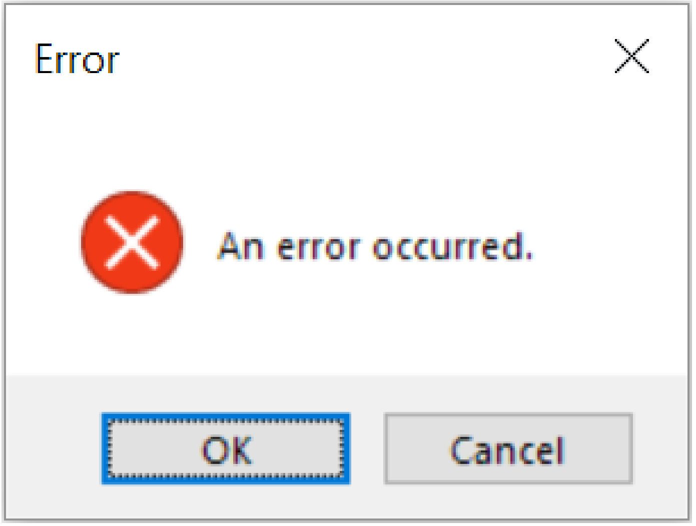
Lancaster University

What happened?

- 1999: a new accounting software system produced by Fujitsu (Horizon) was installed at the Post Office
- Between 2004 and 2014, over 700 post office branch managers received criminal convictions
- Branch managers were accused of faulty accounting and theft. However, in fact Horizon was faulty and had falsely suggested cash shortfalls
- Severe implications, with many people wrongly imprisoned

Post Office case study – technical components

- The Horizon system was faulty, with many errors and bugs
- Lord Justice Holroyde: ‘there were serious issues about the reliability of Horizon’



Post Office case study – people, knowledge and processes

- PO staff members complained of bugs in the system, but were not taken seriously
- Conclusion drawn that Horizon software must be correct; and that PO staff had stolen money

Post Office case study – organizational context and environment

- Over-trust in technology?
- Lack of respect for workers?
- Embarrassment that an expensive tech contract was failing?
- Failings in the legal system – legal presumption of proper functioning of computers? See this legal paper for more information: https://doi.org/10.14296/deeslr.v18i0.5240

Post Office case study – environment

Presuming the correct working of computers – an unsafe presumption?

While the convenience that was sought to be achieved by repeal of s. 69(1)(b) of the Police and Criminal Evidence Act 1984 is understandable, a presumption that a computer 'works correctly' in itself is unsafe and, for anyone with expertise in the area, will appear wholly unreal, because it suggests a binary question of whether the computer is working or not.[14] The reality is more complex.[15] All computers have a propensity to fail, possibly seriously. That is to say, they have a latent propensity to function incorrectly.

From https://doi.org/10.14296/deeslr.v18i0.5240

System errors and failures – case study 3

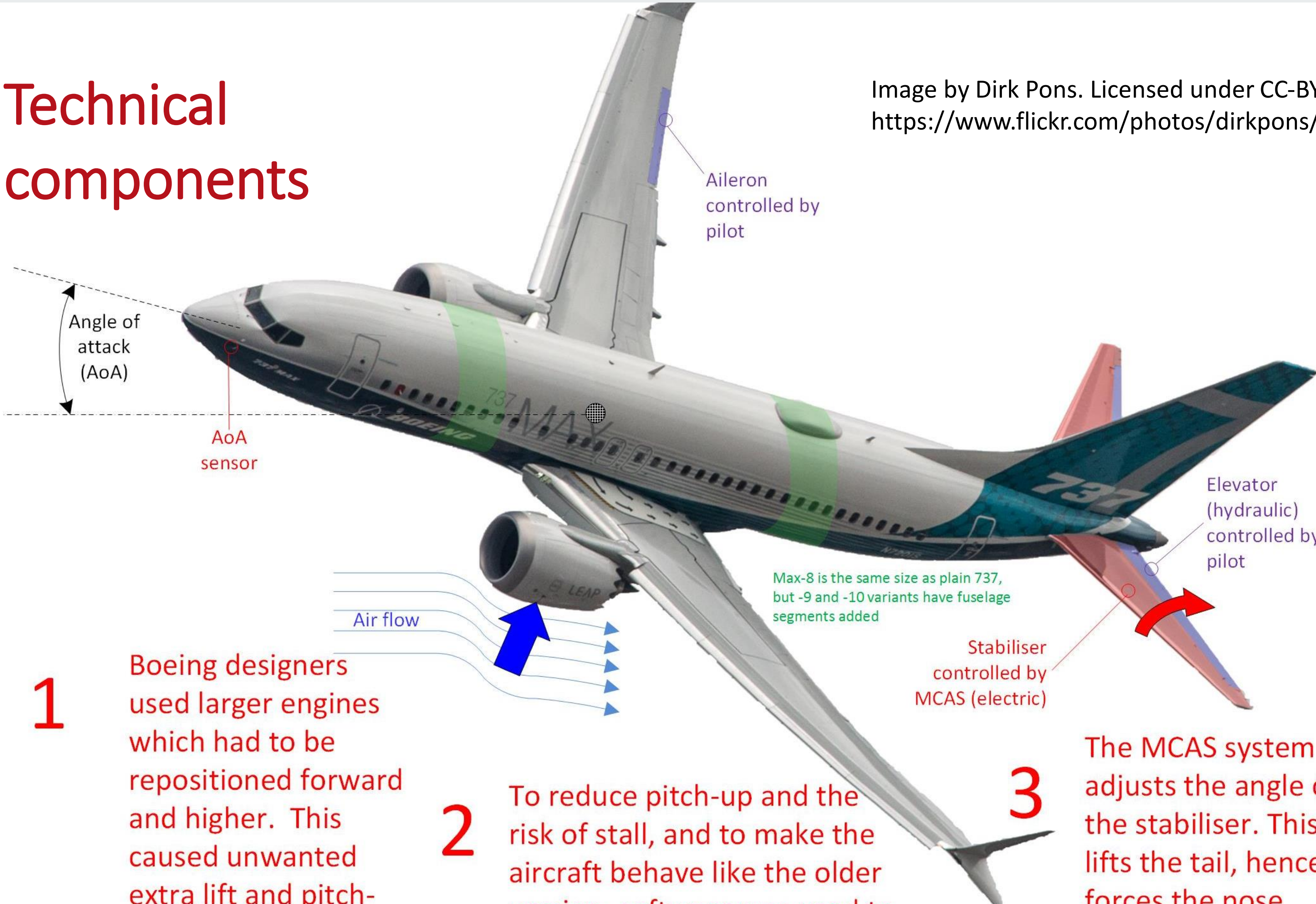
Boeing 737 MAX case study

Boeing 737 MAX case study

What happened?

- October 2018 and March 2019: all passengers, pilots and cabin crew die in Boeing 737 Max crashes

Technical components



1 Boeing designers used larger engines which had to be repositioned forward and higher. This caused unwanted extra lift and pitch-up at high angle of attack.

2 To reduce pitch-up and the risk of stall, and to make the aircraft behave like the older version, software was used to automatically push the nose down, the Maneuvering Characteristics Augmentation System (MCAS). This uses the AoA sensors.

3 The MCAS system adjusts the angle of the stabiliser. This lifts the tail, hence forces the nose down. The system is covert, forceful, and persistent.

Boeing 737 MAX case study – people, knowledge and processes; organizational context

- Software solution chosen for what was a hardware problem (size of engine, and design of plane)
- Seems to have been little open communication around the risks of the system
- Pilots raised concerns which were not listened to.
- Some pilots were not even aware of the new system and how it worked

Boeing 737 MAX case study – environment

Lancaster University

- Market forces pushing airline companies to make larger, faster planes – and for cheaper

- See https://spectrum.ieee.org/how-the-boeing-737-max-disaster-looks-to-a-software-developer for more information (written by a developer and pilot)

- Also Netflix documentary ‘Downfall’ (released in 2022)

Theories and models to understand system failures

Different levels of failure (multi-causal approach)

- **Regulatory failures** - lack of information; under-trained personnel; lack of regulation
- **Managerial Failures** -safety climate, lines of command and responsibility, quality control
- **Hardware Failures** - design failure; requirements failure; implementation failure
- **Software Failures** - requirements failures; specification failures
- **Human Failures** - slips, lapses & mistakes; team factors, human error

Failure in complex systems

- Failure in one part may coincide with the failure of a different part
- This combination can cause cascading failures of other parts
- In complex systems these are many possible combinations

What characterizes a complex system?

Complex interactions:

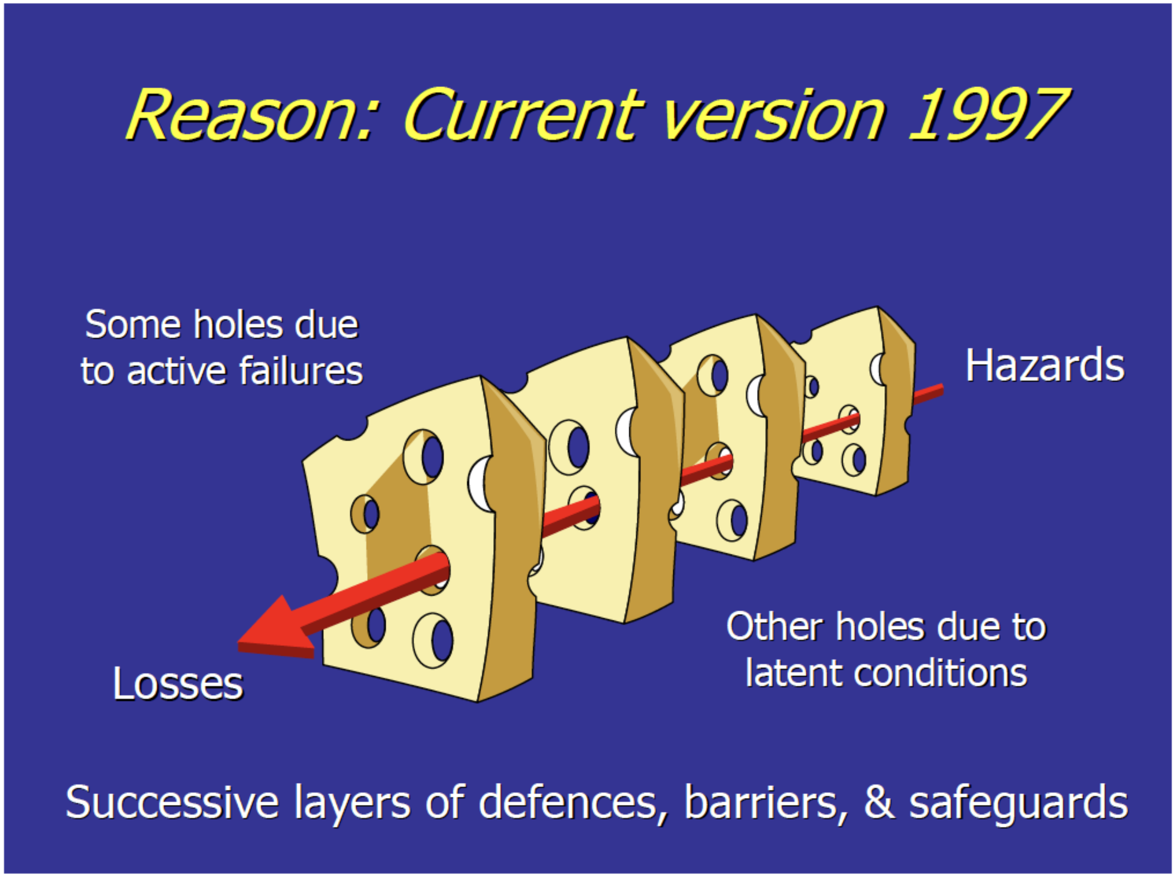
- Unfamiliar, unplanned, or unexpected sequences which are not visible or immediately comprehensible

Tightly coupled:

- Time-dependent processes
- Rigidly ordered processes (sequence B must follow sequence A)
- Very little slack

If a system has interactive complexity and is tightly coupled it is particularly prone to failure.

Reason's Swiss Cheese Model



Reason, J., E. Hollnagel, and J. Paries. "Revisiting the Swiss cheese model of accidents." *Journal of Clinical Engineering*27.4 (2006): 110-115.

Limitations of the Swiss Cheese Model

- Leveson (2004) critique of the model: *"Note that independence of the barriers is assumed and some randomness in whether the "holes" line up"*
- Dekker (2002): "layers of defence are not static or constant, and not independent of each other either. They can interact, support or erode one another"
- Dekker: the Swiss Cheese Model doesn't explain what the holes are, how and why they got there, how the holes line up, etc.

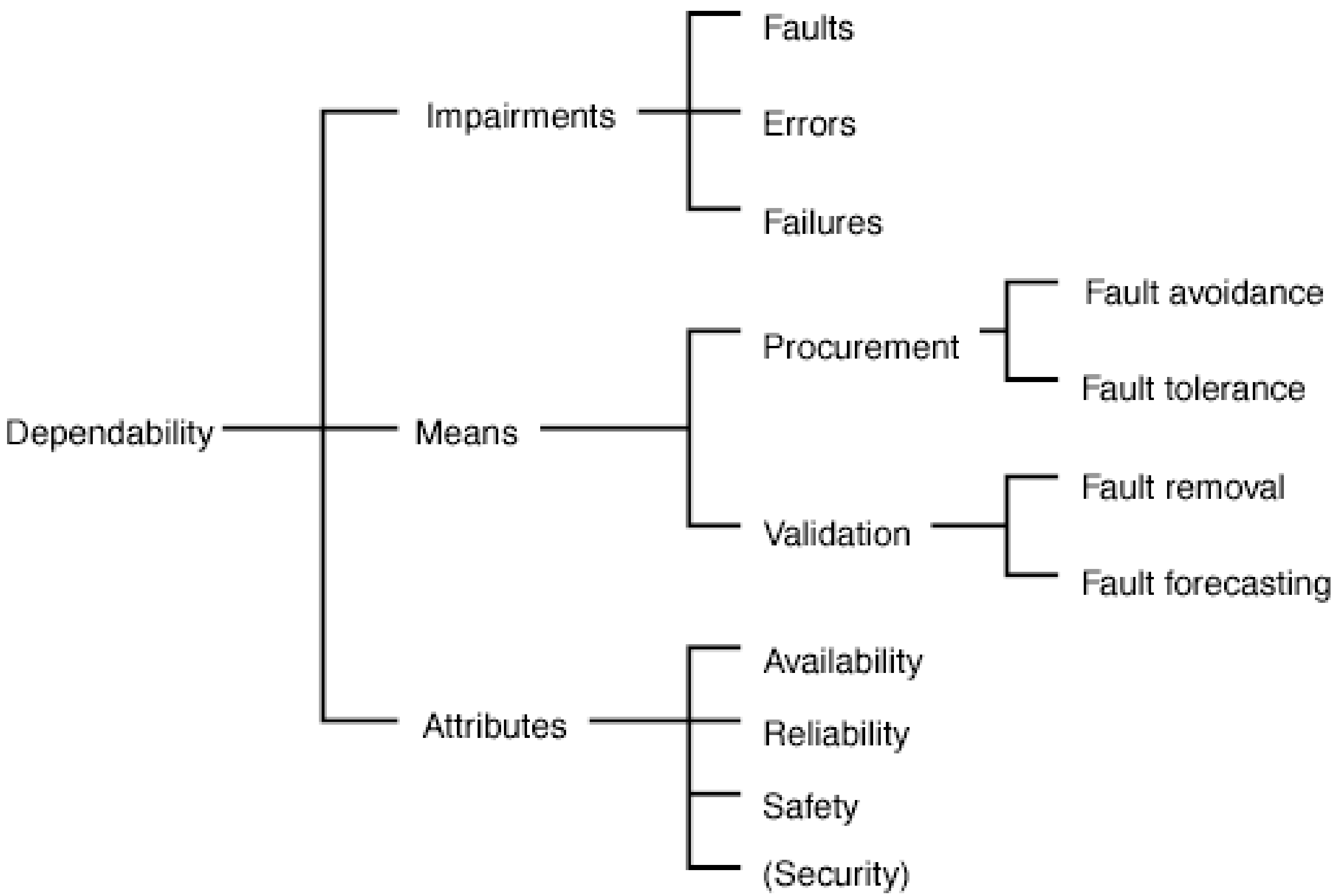
Understanding dependability

The concept of dependability

For most complex socio-technical systems, dependability is the most important property.

- Judgement about the user’s trust in a system.
- Reflects the extent of the user’s confidence that it will operate as expected and will not ‘fail’ in normal use.
- “Dependability is defined as that property of a computer system such that reliance can justifiably be placed on the service it delivers.” (Mellor)

Laprie's model



Laprie's model: impairments

Lancaster University

Faults, errors and failures:

- **System failure** – when the system does not deliver the service its users expect
- **System error** – where the behaviour of the system does not confirm to its specification
- **System fault** – incorrect system state not expected by the designers of the system
- **Human error or mistake** – human behaviour that results in faults being introduced into a system

Laprie's model: means

- **Fault avoidance** – preventing the occurrence or introduction of faults
- **Fault tolerance** – delivering correct service, though faults are present
- **Fault removal** – reducing number or severity of faults
- **Fault forecasting** – estimating number of faults, future occurrence, consequences

Laprie’s model: primary attributes of dependability

- **Availability** – ability of system to deliver services when requested
- **Reliability** – ability of the system to deliver services as specified
- **Safety** – ability of the system to operate without catastrophic failure
- **Security** – ability of the system to protect itself against accidental or deliberate intrusion

Laprie's model: secondary attributes of dependability

Lancaster University

- **Timeliness –** the ability of the system to respond in a timely way to user requests.
- **Survivability** – the ability of a system to continue to deliver its services to users in the face of deliberate or accidental attack
- **Recoverability** – the ability of the system to recover from user or system errors.
- **Maintainability** - the ease of repairing the system after a failure has been discovered or changing the system to include new features.

Understanding (and designing for) human error

What are human errors?

- Can be difficult to distinguish between safe and erroneous behaviour

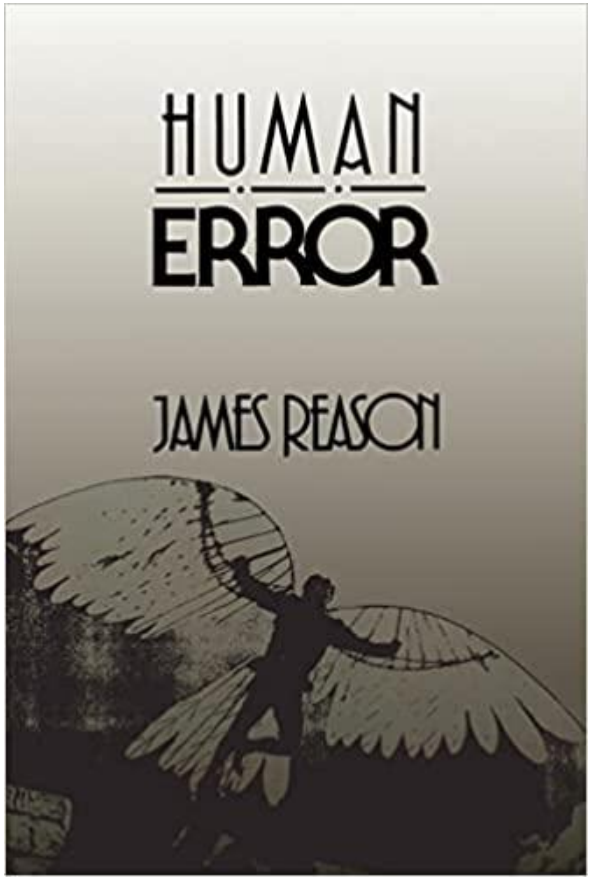
- For example, are the following actions erroneous?

  - Deliberately not following a rule or instruction
  - Following a rule or instruction
  - Saving time by taking a short-cut

Generic Error Modeling System (James Reason)

GEMS defines human error as:

- The failure to perform some plan or task properly
- The failure to apply the correct plan



Generic Error Modeling System – types of action

Lancaster University

1. Skills based performance

- Routine, practiced things done without much cognitive effort

2. Rules based performance

- Following a set of rules or procedure

3. Knowledge based performance

- Applying knowledge

Generic Error Modeling System – types of error

Lancaster University

1. Slips (related to skills-based performance)

- "execution failure"; user's intentions are correct, but actions not carried out properly

2. Lapses (related to skills-based performance)

- "execution failure" – for example, forgetting to do something

3. Mistakes (related to rule- and knowledge-based performance)

- "planning failures"; inappropriate set of actions is carried out

GEMS – Generic Error Modelling System

Lancaster University

Advantages of the model

- Provides a useful framework to thinking about designing systems that minimize, detect and correct, and tolerate human error

Limitations

- Focuses on non-deliberate error, rather than deliberate (e.g, taking short-cuts)
- High level, ignores the importance of context

Why study human error?

- Human error has many negative consequences
- Common reaction – blaming the user

However, systems engineers should be asking:

1. How can we design systems that minimize potential for human error?

2. How can we design systems that can detect and correct human error?

3. How can we design systems that tolerate human error?

Addressing human error

Challenges:

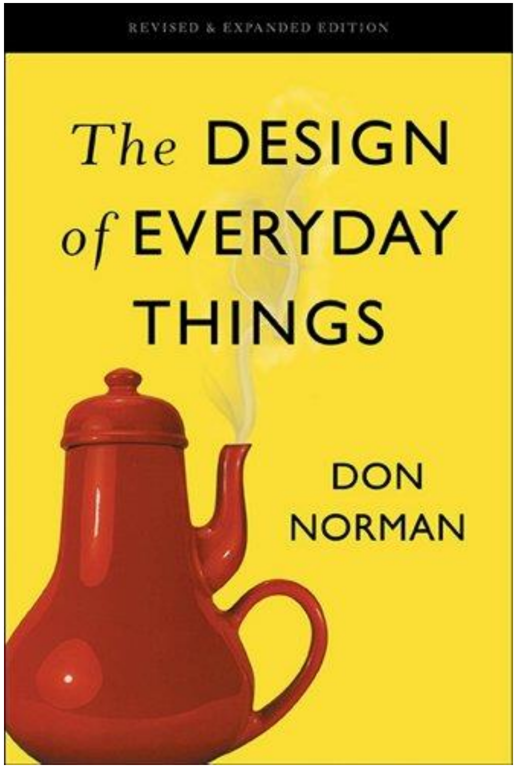
- Humans are inherently fallible and errors are inevitable
- Human behaviour is varied (for example, skills-based, rules-based, knowledge-based)

General approach: error-tolerance rather than error-avoidance

- **"It is now widely held among human reliability specialists that the most productive strategy for dealing with active errors is to focus upon controlling their consequences rather than upon striving for their elimination"** (Reason)

Planning for error

- Increase system visibility - don’t hide complexity behind automated mechanisms
- Take errors into account in operator training – include error scenarios, etc.
- Design interfaces with human user behaviour in mind
- Norman: design for errors. Assume errors will occur and plan for error recovery. For example, make it easy to reverse actions.



Is automation the answer?

Not necessarily!

- Automation addresses skills- and rules-based tasks, leaving complex knowledge-based tasks to humans.
- Automation may hinder understanding, by decreasing system visibility and increasing complexity
- Automation shifts the error source from operator/user errors to design errors, which may be harder to detect and fix

Recap

System errors and failures – key points

- System failures are the result of many compounding factors
- Failures are more likely in complex systems
- Ensuring dependability is crucial for complex systems
- Many failures result from human errors. It is important to design in such a way to minimize human error

Suggestions for additional reading

Lancaster University

- Chris Johnson - website - http://www.dcs.gla.ac.uk/~johnson/ - especially online book ‘Failure in Safety Critical Systems’
- Reason, (1990) *Human Error* Cambridge. Cambridge University Press.
- “To Err is Human” – chapter in Donald Norman (1988) *The Design of Everyday Things*. New York, Doubleday.
- For theories of Human Error and Resilience, look for work by “Erik Hollnagel”

Thank you for attending, any questions?