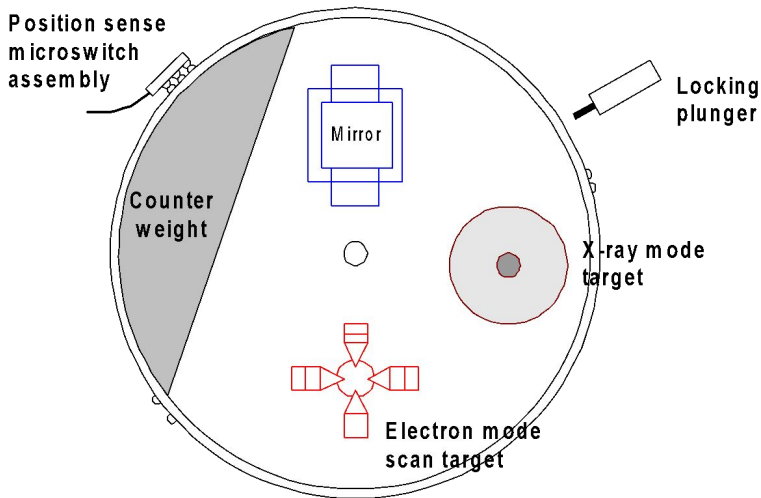


Therac 25

Maps to learning outcomes 1 & 3



Therac 25 Schematic



Therac 25 Overview

- Intended for operation on tumours
 - Uses linear accelerator to produce electron stream and generate X-rays (both can be used in treatments)
- X-ray therapy requires about 100 times more electron energy than electron therapy
 - This level of electron energy is hazardous
 - When X-ray used a different beam path should be taken



Therac 25 Overview

- Selection of beam type controlled by a turntable
 - Therac-25 uses magnets to spread the beam and to reduce electron beam energy when using X-ray therapy
 - If turntable in wrong position, beam is not spread
 - Turntable position and beam activation are both computer controlled



Software in Therac 25

- Initial design required operator to enter data in twice
 - These entries were cross-checked
 - Cross-checks removed to speed up therapy
- Internal problems made worse by interface errors
 - Display did not always correspond with data entered
 - Undocumented error codes
 - These occurred so often the operators ignored them



Software in Therac 25

- Mechanical interlocks from earlier models removed
- Six (known) over-dosage accidents (resulting in several deaths)
 - Hard to be certain as patients were critically ill
- Accidents were caused by
 - X-ray beam used without correct path, i.e. no attenuation / diffusion
 - Synchronisation problems, e.g. changes to level of beam energy after the magnets started moving were not recognised



Software Induced Failures

- Assessment of Therac-25
 - Management problems as well as technical ones
 - Poor control over development
 - Too much trust in software
 - Therac-25 example of bad practice
 - Fortunately, such extreme cases are rare



Discussion Points

- What is the balance between safety and ease of use?
- How often do we see something extended for a second role suffer dependability issues?
- How might the issues have been predicted?
- Was acceptability of increased risk a significant issue?



Further Materials

- N. G. Leveson and C. S. Turner, [An investigation of the Therac-25 accidents](#), *Computer*, vol. 26, no. 7, pp. 18-41, July 1993.

Hazard Analysis

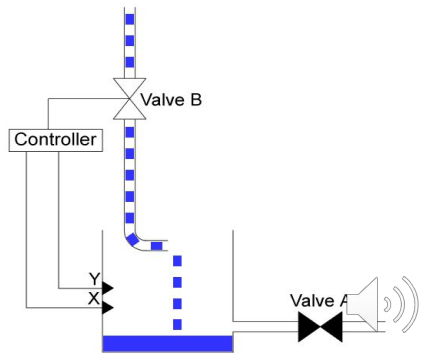
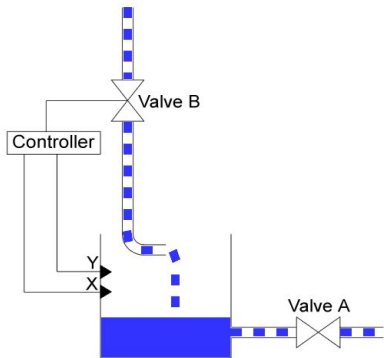
Maps to learning outcomes 2 & 3



Example of a Safety System

- Context

- Dangerous for tank to overflow
- *Downstream system* (after Valve A) relies on fail operational delivery
- Given notification, downstream system can tolerate lack of delivery



Preliminary Hazard Analysis (PHA)

- *Initiation / preparatory work*
 - Define overall purpose of system and identify hazards (catastrophic events)
 - PHA identifies hazards, usually based on experience
 - i.e. historical knowledge of that class of systems (NB HAZOPS)
 - Significant issues may arise when developing new types of systems or domain experts not available



Preliminary Hazard Analysis (PHA)

- *Initiation / preparatory work (cont)*
 - Often this amounts to looking at check lists
 - For example, hazardous action of brakes in take-off is fairly well understood, and leads to a list of failure modes
 - Possibly some feasibility studies
 - Shouldn't really be a solo activity, instead often brainstormed
 - Review by application experts, and relevant technologists, including computing and software, ISA, perhaps marketing



Preliminary Hazard Analysis (PHA)

- HAZOP often used to perform hazard analysis
 - Adapted from the chemical industry
- To identify potential process upset scenarios which could lead to significant safety or operability consequences
- To decide whether current design ensures that the risk from each identified scenario is at a suitably low level
- If not, to recommend modifications which will reduce the risk to a suitably low level, or to specify further actions with the objective of identifying suitable risk reduction measures
- Note that *efficient operation* is usually not included
 - Not directly a safety issue unless workload affects the ability (of a computer or human) to make correct decisions



Preliminary Hazard Analysis (PHA)

- The hazard analysis is performed first and then reviewed against existing hazard lists from similar projects
- At each major stage of the lifecycle, decisions taken will be reviewed against the hazard list to ensure
 - The decision is right
 - The hazard analysis, and hence hazard list is not invalidated



HAZOP

- HAZOP based on guidewords being applied to operations to judge potential effect, deviation and hence derive design recommendations
- Guidewords for software are:
 - Early
 - Late
 - Omission
 - Commission
 - Value (error) detectable
 - Value (error) undetectable



Hazard Analysis for Fuel Tank

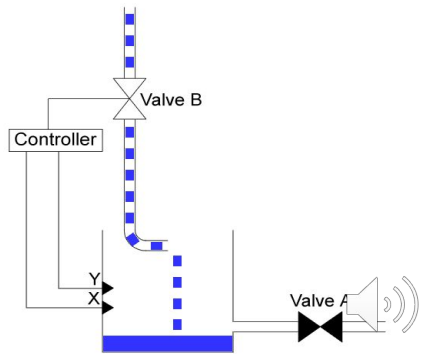
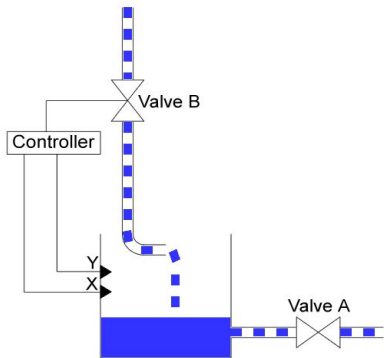
- Components to be analysed
 - Controller
 - Valve A
 - Valve B
 - Sensor X
 - Sensor Y
 - Communications, power supplies, tank and pipes
- An outcome includes set of Hazardous Events and Derived Safety Requirements (DSR)



Example of a Safety System

- Context

- Dangerous for tank to overflow
- *Downstream system* (after Valve A) relies on fail operational delivery
- Given notification, downstream system can tolerate lack of delivery



Hazard Analysis for Fuel Tank

- Controller

- Underline indicates a DSR is probably needed

Guideword	Effect	Comments
Early	Negligible	Could lead to a longer time between updates
Late	Negligible	Could lead to a longer time between updates
Omission	Catastrophic	<u>Complete failure to update both valves leading to a possible hazardous event</u>
Commission	Probably Minor ???	More context would be needed to know if this might be a problem
Value undetectable	Catastrophic	<u>The valves could be set in the wrong position leading to an overflow</u>
Value detectable	Negligible	<u>Assuming appropriate alarms and actions taken</u>



Hazard Analysis for Fuel Tank


- Valve A, Valve B

Guideword	Effect	Comments
Early	Negligible	Same as controller
Late	Negligible	Same as controller
Omission	Negligible	<u>Assuming appropriate alarms and actions taken</u>
Commission	Probably Minor ???	Same as controller
Value undetectable	Negligible	<u>Assuming appropriate alarms and actions taken</u>
Value detectable	Negligible	<u>Assuming appropriate alarms and actions taken</u>



Hazard Analysis for Fuel Tank

- Sensors A and B

Guideword	Effect	Comments
Early	Negligible	Same as controller
Late	Negligible	Same as controller
Omission	Negligible	<u>Assuming appropriate alarms and actions taken</u>
Commission	Probably Minor ???	Same as controller
Value undetectable	Minor if no common cause failure otherwise catastrophic	<u>Assuming no common cause failure AND appropriate alarms and actions taken</u>
Value detectable	Negligible	<u>Assuming appropriate alarm and actions taken</u> 

Discussion Points

- How practical do you think HAZOP is?
- For a system such as a car braking system and / or ABS, do you
 - Think the keywords are right?
 - What might be suitable components?
- How do you believe DSRs should be handled?



Further Materials

- J. A. McDermid and D. J. Pumfrey, [A development of hazard analysis to aid software design](#), Proceedings of COMPASS, 1994.



Failure Mode Effect Analysis (FMEA)

Maps to learning outcomes 2 & 3



Varieties of Failure Analysis Techniques

Primarily concerned with cause and effect – events (including failures) and their consequences

- top-down – proceed from hazardous events back towards possible causes – focussed, efficient
- bottom-up – proceed from possible failures of primitive components towards consequences – time consuming, can catch problems otherwise overlooked
- Fault-tree analysis (FTA) – top-down technique, based on AND/OR graphs, used for hardware and software
- Event-tree analysis (ETA) – bottom-up, becoming more popular, can be used for hardware and software



Varieties of Failure Modes Analysis Techniques

- Failure modes and effect analysis (FMEA)
 - Bottom-up, used quite frequently, but quite onerous
 - More concerned with what the failure modes are than causality
- Failure modes effect and consequences analysis (FMECA)
 - Variant of FMEA more concerned with consequences of failure (cf hazard severity)



Varieties of Failure Modes Analysis Techniques

- Distinction often made between *logical* and *quantitative* analysis
 - Logical shows causal structures only
 - Quantitative associates probabilities with events, often as part of risk assessment (common in FMECA)
- Techniques are (potentially) complementary



Relationship Between Analyses

- Hazard analysis used to show how hazards occur and gauge their importance
- FTA used to show how the hazardous events occur
- FMEA examines the impact of an individual failures and their failure modes



Relationship Between Analyses

- An advantage of FMEA is it may uncover different hazardous events
- By the end of the failure analysis, it is important the analysis is complete, consistent and coherent
- Through the causal (failure) analysis, understanding of system and its assumptions will be enhanced since the hazard analysis
- Important these are clearly captured in safety case
- See earlier lecture on safety arguments



Varieties of Failure Modes Analysis Techniques

- FMEA and FMECA largely derived from experience, producing tables, e.g. nose wheel steering system

Failure Mode	Phase	Probability per hour	Effect	Symptoms
Loss of one hydraulic supply	Ground	10^{-6}	None as dual supply	Indicator on Secondary Panel

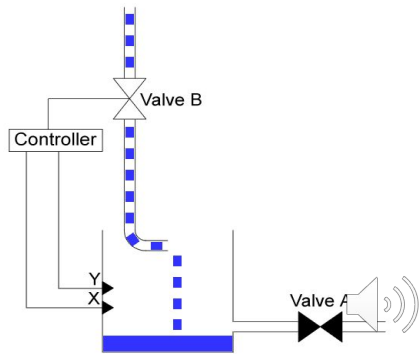
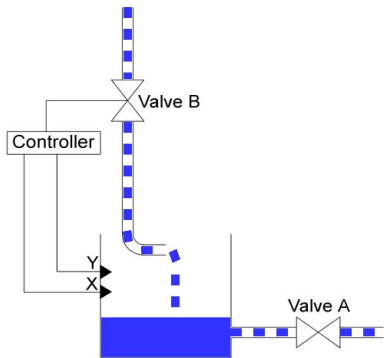
- Tables produced for all systems and components
- Usually quantitative, and important part of safety case
- Does not need to be linked directly with FTA or ETA but clearly complementary
 - One difference is FMEA / FMECA deals with effect and the FTA / ETA deals with relationships
- Usually not applied to software



Example of a Safety System


- Context

- Dangerous for tank to overflow
- *Downstream system* (after Valve A) relies on fail operational delivery
- Given notification, downstream system can tolerate lack of delivery



FMEA for the Fuel Tank

- A decision would be needed whether a loss of one sensor means the system goes into a fail-safe state
- In effect we have a potential need for new DSRs - underlined

Failure Mode	Phase	Probability per hour	Effect	Symptoms
Loss of controller	N/A	< 1 per hour	Loss of service due to protection	Alarms go off
Loss of sensor X	N/A	< 1 per hour	Loss of service due to protection	Alarms go off
Loss of sensor Y	N/A	< 1 per hour	Loss of service due to protection	Alarms go off
Loss of both sensors	N/A	< 1 per hour	Loss of service due to protection	<u>The controller should recognise un-expected sensor values</u> 


FMEA for the Fuel Tank

Failure Mode	Phase	Probability per hour	Effect	Symptoms
Loss of valve A	N/A	< 1 per hour	Loss of service due to protection	Valves should fail in safe state. <u>The controller should recognise un-expected sensor values</u>
Loss of valve B	N/A	< 1 per hour	Loss of service due to protection	Same as Loss of Valve A
Loss of both valves	N/A	< 1 per hour	Loss of service due to protection	Same as Loss of Valve A



FMEA for the Fuel Tank

- Slide features components not in original Hazard Analysis
- FMEA has un-covered some new issues
 - Questions the quality of the hazard analysis but
 - Demonstrates the worth of FMEA

Failure Mode	Phase	Probability per hour	Effect	Symptoms
Loss of communications	N/A	< 1 per hour	Potentially catastrophic due to losses of control and protection mechanisms	Alarms as components should make appropriate sound. <u>Assumes audible alarms do not require comms</u>
Loss of power supply	N/A	< 1 per hour	Loss of service due to protection	Valves should go into safe state
Loss of protection system	N/A	< 1 per hour	Potentially catastrophic	??? as failure modes of the protection mechanism considered 

Discussion Points

- How practical is FMEA as a technique for complex systems?
- How would you apply FMEA to software-based systems?
- At what level might you define a component?
 - Ariane 501
 - Piper Alpha
 - Therac-25



Further Materials

- J. Koch, [Jet Propulsion Laboratory Reliability Analysis Handbook](#), 1990.
- Goddard Space Flight Center , [Performing a Failure Mode and Effects Analysis](#), 1996.
- DoD, [Procedures for performing a failure mode effect and critical analysis](#), MIL-P-1629, 1949.
- DoD, [Procedures for performing a failure mode effect and criticality analysis](#). 1980.



Fault Tree Analysis (FTA)

Maps to learning outcomes 2 & 3



Fault Tree Analysis – Overview

- The classic *deductive* analysis technique, which works back from undesired event to basic causes
- Useful for both qualitative and quantitative analysis
- Developed by Bell Labs and the USAF in early 1960s to investigate potential causes of inadvertent launch of Minuteman missile
- Now the most common diagrammatic safety analysis technique
- US Nuclear Regulatory Commission Fault Tree Handbook is widely accepted as the definition of *standard* fault tree symbology and method



Fault Tree Notation 1 – Events

Standard Event Symbols



Basic Event

An initiating fault requiring no further development



Undeveloped Event

An event which is not developed further, either because it is considered unnecessary, or because insufficient information is available



Intermediate Event

An event arising from the combination of other, more basic events



Normal Event

An event which is expected to occur as part of the normal operation of the system



Fault Tree Notation 2 – Gates

Standard Gate Symbols



AND

All input events must occur for the output to occur



OR

The occurrence of one or more input events will cause the output to occur



EXCLUSIVE OR

The output will occur if exactly one of the inputs occurs



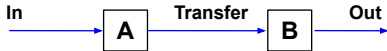
PRIORITY AND

The output occurs if the input events occur in a specific sequence



Fault Tree Analysis Steps 1

- Select an event
 - initially the top event
- Identify immediate, necessary and sufficient causes of this event
 - *immediate* – avoid missing out intermediate events – the *think small* principle. In the simple system below, the immediate causes of *No output from B* are *B fails* and *No transfer from A*, but not *A fails* (ONE MORE STEP AWAY)



- *necessary* – events which must occur together for the top event to occur – linked with an AND gate
 - *sufficient* – events which alone are sufficient to cause the top event – linked with an OR gate
- Repeat



Fault Tree Construction Rules

Additional rules have evolved to help ensure correct construction of fault trees:

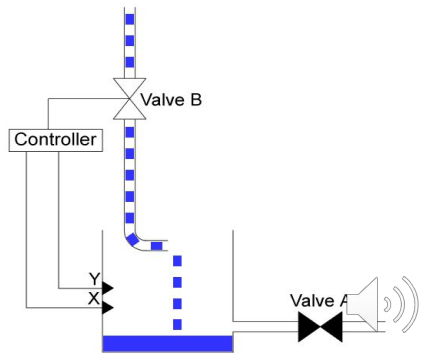
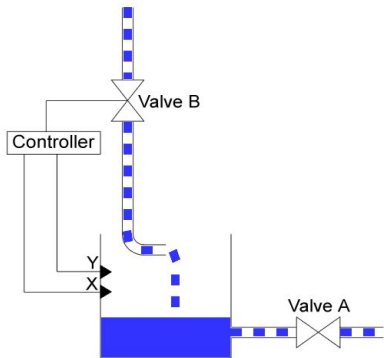
- *All* inputs to a gate should be defined before any one is examined in more detail
- Output of a gate must never be directly from input to another gate – *must always be* a named intermediate event
- Text in boxes should be complete – *what* event is and *when* it occurs
- Causes always chronologically *precede* consequences – sounds obvious, but important in closed-loop control



Example of a Safety System

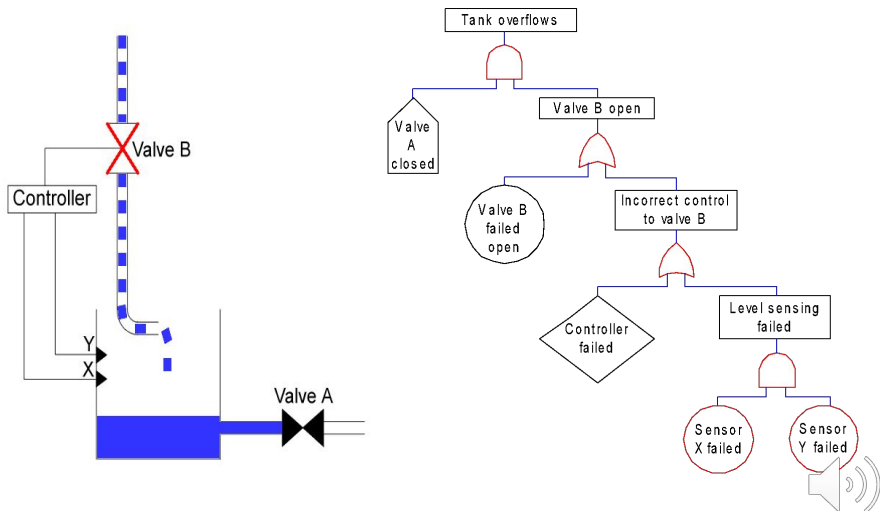
- Context

- Dangerous for tank to overflow
- *Downstream system* (after Valve A) relies on fail operational delivery
- Given notification, downstream system can tolerate lack of delivery



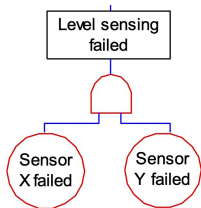
Fault Tree for Tank Overflows

- The fault tree would need to be extended
- To allow for additional components and features based on the DSRs



Fault Tree for Tank Overflows

- Common cause failures are fundamentally important
- Currently there is a logical AND for the two sensors
- This can easily become a logical OR, e.g.
 - Both are in a similar physical position
 - This is in effect what zonal analysis does
 - Both share a power supply
 - Both come from the same manufacturer
 - Both measure the same property and then do the same calculation
 - The calculations are hosted on the same processing device
 - Etc..
- Common cause failures can lead to a relatively simple fault tree becoming much more complex

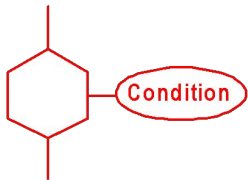


Further gates



M OF N GATE

The output occurs if M of the N inputs (in this case two of the four) occur



CONDITIONING EVENT (IF gate)

The output occurs if the input occurs and if the condition described is also true

- Also specialised gates
 - e.g. summation and comparator, where inputs can be *weighted*
 - rarely used



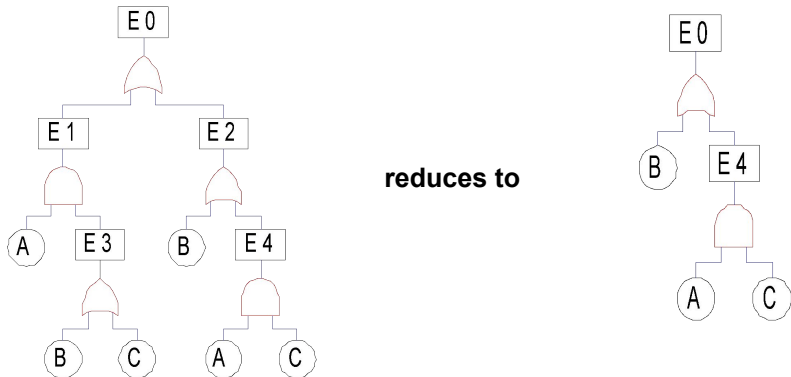
Using Fault Trees for Probabilistic Analysis

- If probabilities of basic events are known, and basic events are *independent*, fault trees can be used as the basis for calculating the probability of top events
- For probability calculations, fault trees must be reduced to *minimal cut-set* form (see next slide)
- Calculation progresses up the tree from basic events
- Simplest case is non-repairable systems using only AND and OR gates. In this case
 - $P(A \wedge B) = P(A) \cdot P(B)$
 - $P(A \vee B) = P(A) + P(B) - P(A) \cdot P(B)$
 - if $P(A)$ and $P(B)$ are very small, $P(A \vee B) \cong P(A) + P(B)$



Fault Tree Analysis – Minimal Cut Sets

- A minimal cut set is the smallest possible combination of events which will cause the top event to occur:



- Boolean algebra methods (e.g. Karnaugh maps, DeMorgan's theorem) may be used to simplify fault trees



Using Fault Trees for Probabilistic Analysis

- For example, the fuel tank overflowing hazardous event
- If the likelihood of a sensor failure is 100 days
- With a simple independent failure model, the likelihood of dual failures is 10,000 days
- With no repair and the system having to be operational, then the likelihood of the tank overflowing is at least 10,000 days



Using Fault Trees for Probabilistic Analysis

- If a faulty sensor was detected and repaired within a day, then the likelihood is approx. 10,000 days
- Reason is once every 100 times the maintenance day occurs the second failure will occur
- An important issue is on a maintenance day there is no *current* way of detecting the second failure
- We may also be relying on two sensors for zonal reasons
- Therefore an important decision is needed

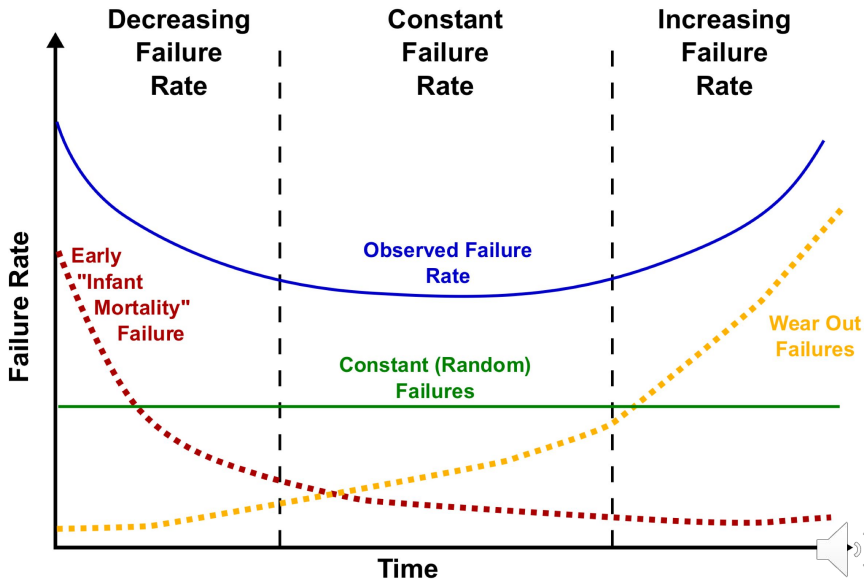


Using Fault Trees for Probabilistic Analysis

- Realistically though towards the end of life, the likelihood of the second sensor failing is more likely
 - This is the bath tub curve, i.e. at the start of use a failure is more likely, in the middle less likely and towards the end of expected life more likely
 - Changing both sensors at the same time helps alleviate this
 - For a fail operational system you would need to replace one, having a *testing phase* and then replace the other
 - The bath tub curve adds further complexity as the testing phase may need to be longer
- Models of failure would be needed to optimise the maintenance cycle to enhance safety
 - These can be quite complex
 - Cost would obviously be a factor



Bath Tub Curve (from Wikipedia)



Using Fault Trees for Probabilistic Analysis

- Other complexities include
 - Voter reliability
 - Shared power supplies
 - If the failure rate of these is 1,000 days, then the likelihood of the hazardous event is approx. 1,000 days



Using Fault Trees for Probabilistic Analysis (cont)

- More advanced methods exist which permit computation of reliability and availability of repairable or phased-mission systems, and are able to handle logic gates other than the basic

AND and OR

- see the Fault Tree Handbook, Villemeur, or papers by Bennetts
- there are also fault tree tools which automate calculation
- If basic events are *not* independent, then some other method (e.g. Cause-Consequence analysis – extended form of ETA) must be used for probabilistic analysis



Fault Tree Analysis – Pros and Cons

- Advantages
 - Thorough, systematic method
 - Well-defined semantics and clear structure of diagrams
 - Widely accepted and applied
 - Can be used for probabilistic analyses
 - Identifies single points of failure leading to top events
- Disadvantages
 - Can be difficult to express complex situations, especially those typically found in computer systems
 - Does not identify groups of faults with identical effects



Discussion Points

- At what stage is it practical to perform a fault tree analysis?
 - Early design concept stage
 - Design review
 - Implementation
 - Pre-deployment
- How easy is it to provide a complete fault tree?
- How easy is it to perform on software?

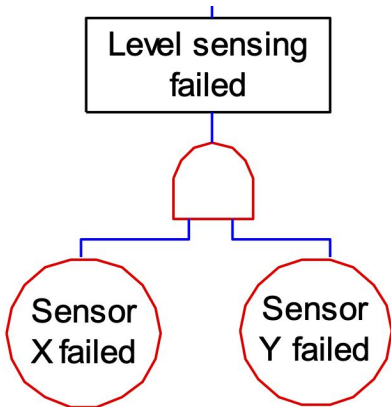


Further Materials

- R. Bennets, On the analysis of fault trees, IEEE Trans. Reliability, pp. 175–185, 1975.
- R. Butler and S. Johnson, [Techniques for Modeling the Reliability of Fault-Tolerant Systems With the Markov State-Space Approach](#), NASA, Ref: 1348, 1995



Original Fault Tree



Revised Fault Tree

SYMBOLS



OR Gate



AND Gate

