Reliability and Safety Engineering

Don Machado

Latarsha Baumann

First Edition, 2012

ISBN 978-81-323-0977-2

Published by:
Academic Studio
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

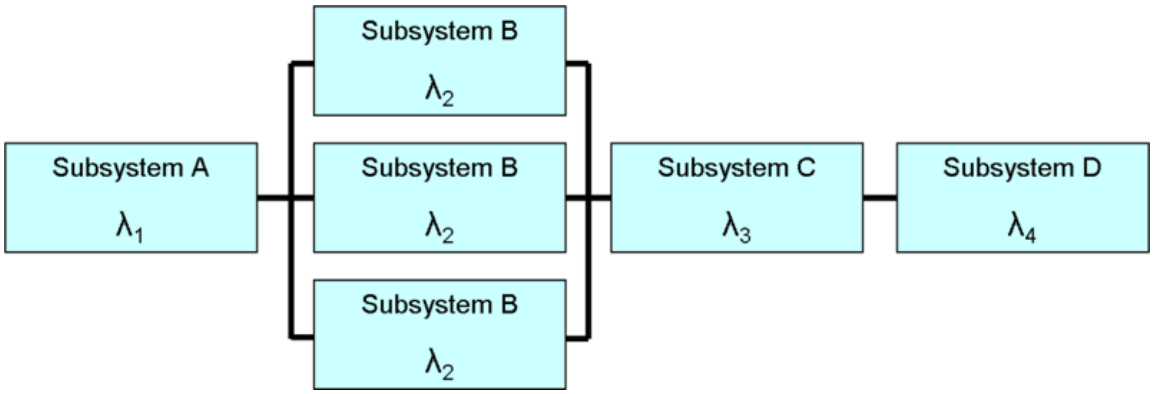
Table of Contents

Chapter 1

Reliability Engineering

Reliability engineering is an engineering field, that deals with the study of reliability: the ability of a system or component to perform its required functions under stated conditions for a specified period of time. It is often reported as a probability.

Overview



A Reliability Block Diagram

Reliability may be defined in several ways:

- The idea that something is fit for a purpose with respect to time;
- The capacity of a device or system to perform as designed;
- The resistance to failure of a device or system;
- The ability of a device or system to perform a required function under stated conditions for a specified period of time;
- The probability that a functional unit will perform its required function for a specified interval under stated conditions.
- The ability of something to "fail well" (fail without catastrophic consequences)

Reliability engineers rely heavily on statistics, probability theory, and reliability theory. Many engineering techniques are used in reliability engineering, such as reliability prediction, Weibull analysis, thermal management, reliability testing and accelerated life

testing. Because of the large number of reliability techniques, their expense, and the varying degrees of reliability required for different situations, most projects develop a reliability program plan to specify the reliability tasks that will be performed for that specific system.

The function of reliability engineering is to develop the reliability requirements for the product, establish an adequate reliability program, and perform appropriate analyses and tasks to ensure the product will meet its requirements. These tasks are managed by a reliability engineer, who usually holds an accredited engineering degree and has additional reliability-specific education and training. Reliability engineering is closely associated with maintainability engineering and logistics engineering, e.g. Integrated Logistics Support (ILS). Many problems from other fields, such as security engineering, can also be approached using reliability engineering techniques. We provides an overview of some of the most common reliability engineering tasks.

Many types of engineering employ reliability engineers and use the tools and methodology of reliability engineering. For example:

- System engineers design complex systems having a specified reliability
- Mechanical engineers may have to design a machine or system with a specified reliability
- Automotive engineers have reliability requirements for the automobiles (and components) which they design
- Electronics engineers must design and test their products for reliability requirements.
- In software engineering and systems engineering the **reliability engineering** is the subdiscipline of ensuring that a system (or a device in general) will perform its intended function(s) when operated in a specified manner for a specified length of time. Reliability engineering is performed throughout the entire life cycle of a system, including development, test, production and operation.

Reliability theory

Reliability theory is the foundation of reliability engineering. For engineering purposes, reliability is defined as:

> **the probability that a device will perform its intended function during a specified period of time under stated conditions.**

Mathematically, this may be expressed as,

$$R(t) = Pr\{T > t\} = \int_t^\infty f(x)\,dx$$

,

where $f(x)$ is the failure probability density function and *t* is the length of the period of time (which is assumed to start from time zero).

Reliability engineering is concerned with four key elements of this definition:

- First, reliability is a probability. This means that failure is regarded as a random phenomenon: it is a recurring event, and we do not express any information on individual failures, the causes of failures, or relationships between failures, except that the likelihood for failures to occur varies over time according to the given probability function. Reliability engineering is concerned with meeting the specified probability of success, at a specified statistical confidence level.
- Second, reliability is predicated on "intended function:" Generally, this is taken to mean operation without failure. However, even if no individual part of the system fails, but the system as a whole does not do what was intended, then it is still charged against the system reliability. The system requirements specification is the criterion against which reliability is measured.
- Third, reliability applies to a specified period of time. In practical terms, this means that a system has a specified chance that it will operate without failure before time t. Reliability engineering ensures that components and materials will meet the requirements during the specified time. Units other than time may sometimes be used. The automotive industry might specify reliability in terms of miles, the military might specify reliability of a gun for a certain number of rounds fired. A piece of mechanical equipment may have a reliability rating value in terms of cycles of use.
- Fourth, reliability is restricted to operation under stated conditions. This constraint is necessary because it is impossible to design a system for unlimited conditions. A Mars Rover will have different specified conditions than the family car. The operating environment must be addressed during design and testing. Also, that same rover, may be required to operate in varying conditions requiring additional scrutiny.

Reliability program plan

Many tasks, methods, and tools can be used to achieve reliability. Every system requires a different level of reliability. A commercial airliner must operate under a wide range of conditions. The consequences of failure are grave, but there is a correspondingly higher budget. A pencil sharpener may be more reliable than an airliner, but has a much different set of operational conditions, insignificant consequences of failure, and a much lower budget.

A reliability program plan is used to document exactly what tasks, methods, tools, analyses, and tests are required for a particular system. For complex systems, the reliability program plan is a separate document. For simple systems, it may be combined with the systems engineering management plan or integrated Logistics Support management plan. The reliability program plan is essential for a successful reliability program and is developed early during system development. It specifies not only what the

reliability engineer does, but also the tasks performed by others. The reliability program plan is approved by top program management.

Reliability requirements

For any system, one of the first tasks of reliability engineering is to adequately specify the reliability requirements. Reliability requirements address the system itself, test and assessment requirements, and associated tasks and documentation. Reliability requirements are included in the appropriate system/subsystem requirements specifications, test plans, and contract statements.

System reliability parameters

Requirements are specified using reliability parameters. The most common reliability parameter is the mean time between failures (MTBF), which can also be specified as the failure rate or the number of failures during a given period. These parameters are very useful for systems that are operated frequently, such as most vehicles, machinery, and electronic equipment. Reliability increases as the MTBF increases. The MTBF is usually specified in hours, but can also be used with other units of measurement such as miles or cycles.

In other cases, reliability is specified as the probability of mission success. For example, reliability of a scheduled aircraft flight can be specified as a dimensionless probability or a percentage refer to system safety engineering.

A special case of mission success is the single-shot device or system. These are devices or systems that remain relatively dormant and only operate once. Examples include automobile airbags, thermal batteries and missiles. Single-shot reliability is specified as a probability of success, or is subsumed into a related parameter. Single-shot missile reliability may be incorporated into a requirement for the probability of hit.

For such systems, the probability of failure on demand (PFD) is the reliability measure. This PFD is derived from failure rate and mission time for non-repairable systems. For repairable systems, it is obtained from failure rate and mean-time-to-repair (MTTR) and test interval. This measure may not be unique for a given system as this measure depends on the kind of demand. In addition to system level requirements, reliability requirements may be specified for critical subsystems. In all cases, reliability parameters are specified with appropriate statistical confidence intervals.

Reliability modelling

Reliability modelling is the process of predicting or understanding the reliability of a component or system. Two separate fields of investigation are common: The physics of failure approach uses an understanding of the failure mechanisms involved, such as crack propagation or chemical corrosion; The parts stress modelling approach is an empirical

method for prediction based on counting the number and type of components of the system, and the stress they undergo during operation.

For systems with a clearly defined failure time (which is sometimes not given for systems with a drifting parameter), the empirical distribution function of these failure times can be determined. This is done in general in an accelerated experiment with increased stress. These experiments can be divided into two main categories:

Early failure rate studies determine the distribution with a decreasing failure rate over the first part of the bathtub curve. Here in general only moderate stress is necessary. The stress is applied for a limited period of time in what is called a censored test. Therefore, only the part of the distribution with early failures can be determined.

In so-called zero defect experiments, only limited information about the failure distribution is acquired. Here the stress, stress time, or the sample size is so low that not a single failure occurs. Due to the insufficient sample size, only an upper limit of the early failure rate can be determined. At any rate, it looks good for the customer if there are no failures.

In a study of the intrinsic failure distribution, which is often a material property, higher stresses are necessary to get failure in a reasonable period of time. Several degrees of stress have to be applied to determine an acceleration model. The empirical failure distribution is often parametrised with a Weibull or a log-normal model.

It is a general praxis to model the early failure rate with an exponential distribution. This less complex model for the failure distribution has only one parameter: the constant failure rate. In such cases, the Chi-square distribution can be used to find the goodness of fit for the estimated failure rate. Compared to a model with a decreasing failure rate, this is quite pessimistic (important remark: this is not the case if less hours / load cycles are tested than service life in a wear-out type of test, in this case the opposite is true and assuming a more constant failure rate than it is in reality can be dangerous). Sensitivity analysis should be conducted in this case.

Reliability test requirements

Because reliability is a probability, even highly reliable systems have some chance of failure. However, testing reliability requirements is problematic for several reasons. A single test is insufficient to generate enough statistical data. Multiple tests or long-duration tests are usually very expensive. Some tests are simply impractical. Reliability engineering is used to design a realistic and affordable test program that provides enough evidence that the system meets its requirement. Statistical confidence levels are used to address some of these concerns. A certain parameter is expressed along with a corresponding confidence level: for example, an MTBF of 1000 hours at 90% confidence level. From this specification, the reliability engineer can design a test with explicit criteria for the number of hours and number of failures until the requirement is met or failed.

The combination of reliability parameter value and confidence level greatly affects the development cost and the risk to both the customer and producer. Care is needed to select the best combination of requirements. Reliability testing may be performed at various levels, such as component, subsystem, and system. Also, many factors must be addressed during testing, such as extreme temperature and humidity, shock, vibration, and heat. Reliability engineering determines an effective test strategy so that all parts are exercised in relevant environments. For systems that must last many years, reliability engineering may be used to design an accelerated life test as well.

Reliability prediction

A prediction of reliability is an important element in the process of selecting equipment for use by telecommunications service providers and other buyers of electronic equipment. Reliability is a measure of the frequency of equipment failures as a function of time. Reliability has a major impact on maintenance and repair costs and on the continuity of service. Reliability predictions:

- Help assess the effect of product reliability on the maintenance activity and on the quantity of spare units required for acceptable field performance of any particular system. For example, predictions of the frequency of unit level maintenance actions can be obtained. Reliability prediction can be used to size spare populations.
- Provide necessary input to system-level reliability models. System-level reliability models can subsequently be used to predict, for example, frequency of system outages in steady-state, frequency of system outages during early life, expected downtime per year, and system availability.
- Provide necessary input to unit and system-level Life Cycle Cost Analyses. Life cycle cost studies determine the cost of a product over its entire life. Therefore, how often a unit will have to be replaced needs to be known. Inputs to this process include unit and system failure rates. This includes how often units and systems fail during the first year of operation as well as in later years.
- Assist in deciding which product to purchase from a list of competing products. As a result, it is essential that reliability predictions be based on a common procedure.
- Can be used to set factory test standards for products requiring a reliability test. Reliability predictions help determine how often the system should fail.
- Are needed as input to the analysis of complex systems such as switching systems and digital cross-connect systems. It is necessary to know how often different parts of the system are going to fail even for redundant components.
- Can be used in design trade-off studies. For example, a supplier could look at a design with many simple devices and compare it to a design with fewer devices that are newer but more complex. The unit with fewer devices is usually more reliable.
- Can be used to set achievable in-service performance standards against which to judge actual performance and stimulate action.

The telecommunications industry has devoted much time over the years to concentrate on developing reliability models for electronic equipment. One such tool is the Automated Reliability Prediction Procedure (ARPP), which is an Excel-spreadsheet software tool that automates the reliability prediction procedures in SR-332, Reliability Prediction Procedure for Electronic Equipment. FD-ARPP-01 provides suppliers and manufacturers with a tool for making Reliability Prediction Procedure (RPP) calculations. It also provides a means for understanding RPP calculations through the capability of interactive examples provided by the user.

The RPP views electronic systems as hierarchical assemblies. Systems are constructed from units that, in turn, are constructed from devices. The methods presented predict reliability at these three hierarchical levels:

1. *Device*: A basic component (or part)
2. *Unit*: Any assembly of devices. This may include, but is not limited to, circuit packs, modules, plug-in units, racks, power supplies, and ancillary equipment. Unless otherwise dictated by maintenance considerations, a unit will usually be the lowest level of replaceable assemblies/devices. The RPP is aimed primarily at reliability prediction of units.
3. *Serial System*: Any assembly of units for which the failure of any single unit will cause a failure of the system.

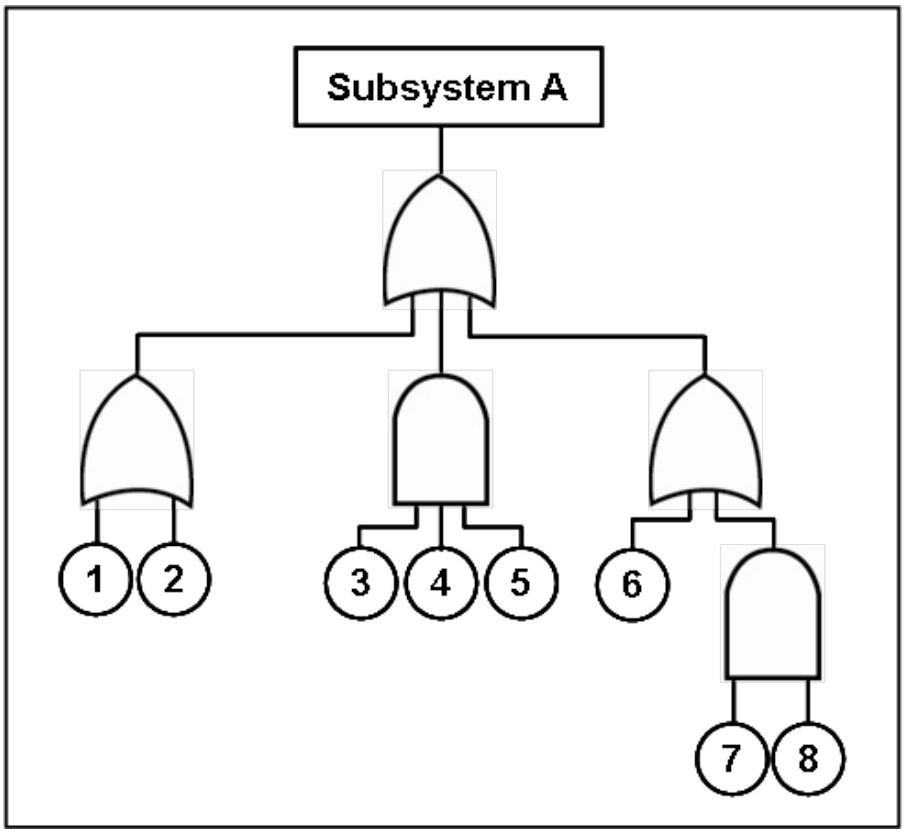
Requirements for reliability tasks

Reliability engineering must also address requirements for various reliability tasks and documentation during system development, test, production, and operation. These requirements are generally specified in the contract statement of work and depend on how much leeway the customer wishes to provide to the contractor. Reliability tasks include various analyses, planning, and failure reporting. Task selection depends on the criticality of the system as well as cost. A critical system may require a formal failure reporting and review process throughout development, whereas a non-critical system may rely on final test reports. The most common reliability program tasks are documented in reliability program standards, such as MIL-STD-785 and IEEE 1332. Failure reporting analysis and corrective action systems are a common approach for product/process reliability monitoring.

Design for reliability

Design For Reliability (DFR), is an emerging discipline that refers to the process of designing reliability into products. This process encompasses several tools and practices and describes the order of their deployment that an organization needs to have in place to drive reliability into their products. Typically, the first step in the DFR process is to set the system’s reliability requirements. Reliability must be "designed in" to the system. During system design, the top-level reliability requirements are then allocated to subsystems by design engineers and reliability engineers working together.

Reliability design begins with the development of a model. Reliability models use **block diagrams** and **fault trees** to provide a graphical means of evaluating the relationships between different parts of the system. These models incorporate predictions based on parts-count failure rates taken from historical data. While the predictions are often not accurate in an absolute sense, they are valuable to assess relative differences in design alternatives.



A Fault Tree Diagram

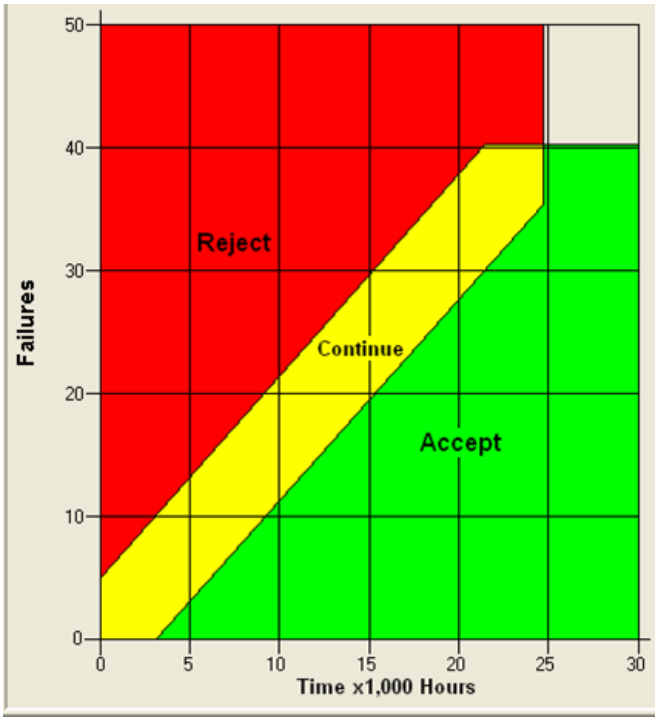
One of the most important design techniques is **redundancy**. This means that if one part of the system fails, there is an alternate success path, such as a backup system. An automobile brake light might use two light bulbs. If one bulb fails, the brake light still operates using the other bulb. Redundancy significantly increases system reliability, and is often the only viable means of doing so. However, redundancy is difficult and expensive, and is therefore limited to critical parts of the system. Another design technique, **physics of failure**, relies on understanding the physical processes of stress, strength and failure at a very detailed level. Then the material or component can be re-designed to reduce the probability of failure. Another common design technique is **component derating**: Selecting components whose tolerance significantly exceeds the expected stress, as using a heavier gauge wire that exceeds the normal specification for the expected electrical current.

Many tasks, techniques and analyses are specific to particular industries and applications. Commonly these include:

- Built-in test (BIT)
- Failure mode and effects analysis (FMEA)
- Reliability simulation modeling
- Thermal analysis
- Reliability Block Diagram analysis
- Fault tree analysis
- Root cause analysis
- Sneak circuit analysis
- Accelerated Testing
- Reliability Growth analysis
- Weibull analysis
- Electromagnetic analysis
- Statistical interference
- Avoid Single Point of Failure

Results are presented during the system design reviews and logistics reviews. Reliability is just one requirement among many system requirements. Engineering trade studies are used to determine the optimum balance between reliability and other requirements and constraints.

Reliability testing



A Reliability Sequential Test Plan

The purpose of reliability testing is to discover potential problems with the design as early as possible and, ultimately, provide confidence that the system meets its reliability requirements.

Reliability testing may be performed at several levels. Complex systems may be tested at component, circuit board, unit, assembly, subsystem and system levels. (The test level nomenclature varies among applications.) For example, performing environmental stress screening tests at lower levels, such as piece parts or small assemblies, catches problems before they cause failures at higher levels. Testing proceeds during each level of integration through full-up system testing, developmental testing, and operational testing, thereby reducing program risk. System reliability is calculated at each test level. Reliability growth techniques and failure reporting, analysis and corrective active systems (FRACAS) are often employed to improve reliability as testing progresses. The drawbacks to such extensive testing are time and expense. Customers may choose to accept more risk by eliminating some or all lower levels of testing.

Another type of tests are called Sequental Probability Ratio type of tests. These tests use both a statistical type 1 and type 2 error, combined with a discrimination ratio as main input (together with the R requirement). This test sets - Independently - before the start of the test both the risk of incorrectly accepting a bad design (Type 2 error) and the risk of incorrectly rejecting a good design (type 1 error) together with the discrimination ratio and the required minimum reliability parameter. The test is therefore more controllable and provides more information for a quality and business point of view. The number of test samples is not fixed, but it is said that this test is in general more efficient (requires less samples) and provides more information than for example zero failure testing.

It is not always feasible to test all system requirements. Some systems are prohibitively expensive to test; some failure modes may take years to observe; some complex interactions result in a huge number of possible test cases; and some tests require the use of limited test ranges or other resources. In such cases, different approaches to testing can be used, such as accelerated life testing, design of experiments, and simulations.

The desired level of statistical confidence also plays an important role in reliability testing. Statistical confidence is increased by increasing either the test time or the number of items tested. Reliability test plans are designed to achieve the specified reliability at the specified confidence level with the minimum number of test units and test time. Different test plans result in different levels of risk to the producer and consumer. The desired reliability, statistical confidence, and risk levels for each side influence the ultimate test plan. Good test requirements ensure that the customer and developer agree in advance on how reliability requirements will be tested.

A key aspect of reliability testing is to define "failure". Although this may seem obvious, there are many situations where it is not clear whether a failure is really the fault of the system. Variations in test conditions, operator differences, weather, and unexpected situations create differences between the customer and the system developer. One strategy to address this issue is to use a **scoring conference** process. A scoring

conference includes representatives from the customer, the developer, the test organization, the reliability organization, and sometimes independent observers. The scoring conference process is defined in the statement of work. Each test case is considered by the group and "scored" as a success or failure. This scoring is the official result used by the reliability engineer.

As part of the requirements phase, the reliability engineer develops a test strategy with the customer. The test strategy makes trade-offs between the needs of the reliability organization, which wants as much data as possible, and constraints such as cost, schedule, and available resources. Test plans and procedures are developed for each reliability test, and results are documented in official reports.

Accelerated testing

The purpose of accelerated life testing is to induce field failure in the laboratory at a much faster rate by providing a harsher, but nonetheless representative, environment. In such a test the product is expected to fail in the lab just as it would have failed in the field—but in much less time. The main objective of an accelerated test is either of the following:

- To discover failure modes
- To predict the normal field life from the high stress lab life

An **Accelerated testing** program can be broken down into the following steps:

- Define objective and scope of the test
- Collect required information about the product
- Identify the stress(es)
- Determine level of stress(es)
- Conduct the Accelerated test and analyse the accelerated data.

Common way to determine a life stress relationship are

- Arrhenius Model
- Eyring Model
- Inverse Power Law Model
- Temperature-Humidity Model
- Temperature Non-thermal Model

Software reliability

Software reliability is a special aspect of reliability engineering. System reliability, by definition, includes all parts of the system, including hardware, software, operators and procedures. Traditionally, reliability engineering focuses on critical hardware parts of the system. Since the widespread use of digital integrated circuit technology, software has become an increasingly critical part of most electronics and, hence, nearly all present day

systems. There are significant differences, however, in how software and hardware behave. Most hardware unreliability is the result of a component or material failure that results in the system not performing its intended function. Repairing or replacing the hardware component restores the system to its original unfailed state. However, software does not fail in the same sense that hardware fails. Instead, software unreliability is the result of unanticipated results of software operations. Even relatively small software programs can have astronomically large combinations of inputs and states that are infeasible to exhaustively test. Restoring software to its original state only works until the same combination of inputs and states results in the same unintended result. Software reliability engineering must take this into account.

Despite this difference in the source of failure between software and hardware — software does not wear out — some in the software reliability engineering community believe statistical models used in hardware reliability are nevertheless useful as a measure of software reliability, describing what we experience with software: the longer you run software, the higher the probability you will eventually use it in an untested manner and find a latent defect that results in a failure (Shooman 1987), (Musa 2005), (Denney 2005).

As with hardware, software reliability depends on good requirements, design and implementation. Software reliability engineering relies heavily on a disciplined software engineering process to anticipate and design against unintended consequences. There is more overlap between software quality engineering and software reliability engineering than between hardware quality and reliability. A good software development plan is a key aspect of the software reliability program. The software development plan describes the design and coding standards, peer reviews, unit tests, configuration management, software metrics and software models to be used during software development.

A common reliability metric is the number of software faults, usually expressed as faults per thousand lines of code. This metric, along with software execution time, is key to most software reliability models and estimates. The theory is that the software reliability increases as the number of faults (or fault density) goes down. Establishing a direct connection between fault density and mean-time-between-failure is difficult, however, because of the way software faults are distributed in the code, their severity, and the probability of the combination of inputs necessary to encounter the fault. Nevertheless, fault density serves as a useful indicator for the reliability engineer. Other software metrics, such as complexity, are also used.

Testing is even more important for software than hardware. Even the best software development process results in some software faults that are nearly undetectable until tested. As with hardware, software is tested at several levels, starting with individual units, through integration and full-up system testing. Unlike hardware, it is inadvisable to skip levels of software testing. During all phases of testing, software faults are discovered, corrected, and re-tested. Reliability estimates are updated based on the fault density and other metrics. At system level, mean-time-between-failure data are collected and used to estimate reliability. Unlike hardware, performing exactly the same test on

exactly the same software configuration does not provide increased statistical confidence. Instead, software reliability uses different metrics such as test coverage.

Eventually, the software is integrated with the hardware in the top-level system, and software reliability is subsumed by system reliability. The Software Engineering Institute's Capability Maturity Model is a common means of assessing the overall software development process for reliability and quality purposes. However, actual software reliability is served through SAE standards JA1002 and JA1003.

Reliability Operational Assessment

After a system is produced, reliability engineering monitors, assesses, and corrects deficiencies. Monitoring includes electronic and visual surveillance of critical parameters identified during the fault tree analysis design stage. The data are constantly analyzed using statistical techniques, such as Weibull analysis and linear regression, to ensure the system reliability meets requirements. Reliability data and estimates are also key inputs for system logistics. Data collection is highly dependent on the nature of the system. Most large organizations have quality control groups that collect failure data on vehicles, equipment, and machinery. Consumer product failures are often tracked by the number of returns. For systems in dormant storage or on standby, it is necessary to establish a formal surveillance program to inspect and test random samples. Any changes to the system, such as field upgrades or recall repairs, require additional reliability testing to ensure the reliability of the modification. Since it is not possible to anticipate all the failure modes of a given system, especially ones with a human element, failures will occur. The reliability program also includes a systematic root cause analysis that identifies the causal relationships involved in the failure such that effective corrective actions may be implemented. When possible, system failures and corrective actions are reported to the reliability engineering organization.

One of the most common methods to apply a Reliability Operational Assessment are Failure Reporting, Analysis and Corrective Action Systems (FRACAS). This systematic approach develops a reliability, safety and logistics assessment based on Failure / Incident reporting, management, analysis and corrective/preventive actions. Organizations today are adopting this method and utilize commercial systems such as a Web based FRACAS application enabling and organization to create a failure/incident data repository from which statistics can be derived to view accurate and genuine reliability, safety and quality performances.

Some of the common outputs from a FRACAS system includes: Field MTBF, MTTR, Spares Consumption, Reliability Growth, Failure/Incidents distribution by type, location, part no., serial no, symptom etc.

Reliability organizations

Systems of any significant complexity are developed by organizations of people, such as a commercial company or a government agency. The reliability engineering organization

must be consistent with the company's organizational structure. For small, non-critical systems, reliability engineering may be informal. As complexity grows, the need arises for a formal reliability function. Because reliability is important to the customer, the customer may even specify certain aspects of the reliability organization.

There are several common types of reliability organizations. The project manager or chief engineer may employ one or more reliability engineers directly. In larger organizations, there is usually a product assurance or specialty engineering organization, which may include reliability, maintainability, quality, safety, human factors, logistics, etc. In such case, the reliability engineer reports to the product assurance manager or specialty engineering manager.

In some cases, a company may wish to establish an independent reliability organization. This is desirable to ensure that the system reliability, which is often expensive and time consuming, is not unduly slighted due to budget and schedule pressures. In such cases, the reliability engineer works for the project day-to-day, but is actually employed and paid by a separate organization within the company.

Because reliability engineering is critical to early system design, it has become common for reliability engineers, however the organization is structured, to work as part of an integrated product team.

Certification

The American Society for Quality has a program to become a Certified Reliability Engineer, CRE. Certification is based on education, experience, and a certification test: periodic recertification is required. The body of knowledge for the test includes: reliability management, design evaluation, product safety, statistical tools, design and development, modeling, reliability testing, collecting and using data, etc.

Another highly respected certification program is the CRP (Certified Reliability Professional). To achieve certification, candidates must complete a series of courses focused on important Reliability Engineering topics, successfully apply the learned body of knowledge in the workplace and publicly present this expertise in an industry conference or journal.

Reliability engineering education

Some Universities offer graduate degrees in Reliability Engineering. Other reliability engineers typically have an engineering degree, which can be in any field of engineering, from an accredited university or college program. Many engineering programs offer reliability courses, and some universities have entire reliability engineering programs. A reliability engineer may be registered as a Professional Engineer by the state, but this is not required by most employers. There are many professional conferences and industry training programs available for reliability engineers. Several professional organizations

exist for reliability engineers, including the IEEE Reliability Society, the American Society for Quality (ASQ), and the Society of Reliability Engineers (SRE).

Chapter 2

Failure Rate

Failure rate is the frequency with which an engineered system or component fails, expressed for example in failures per hour. It is often denoted by the Greek letter λ (lambda) and is important in reliability engineering.

The failure rate of a system usually depends on time, with the rate varying over the life cycle of the system. For example, an automobile's failure rate in its fifth year of service may be many times greater than its failure rate during its first year of service. One does not expect to replace an exhaust pipe, overhaul the brakes, or have major transmission problems in a new vehicle.

In practice, the mean time between failures (MTBF, $1/\lambda$) is often used instead of the failure rate. The MTBF is an important system parameter in systems where failure rate needs to be managed, in particular for safety systems. The MTBF appears frequently in the engineering design requirements, and governs frequency of required system maintenance and inspections. In special processes called renewal processes, where the time to recover from failure can be neglected and the likelihood of failure remains constant with respect to time, the failure rate is simply the multiplicative inverse of the MTBF ($1/\lambda$).

A similar ratio used in the transport industries, especially in railways and trucking is '**mean distance between failures'**, a variation which attempts to correlate actual loaded distances to similar reliability needs and practices.

Failure rates are important factors in the insurance, finance, commerce and regulatory industries and fundamental to the design of safe systems in a wide variety of applications.

Failure rate in the discrete sense

The failure rate can be defined as the following:

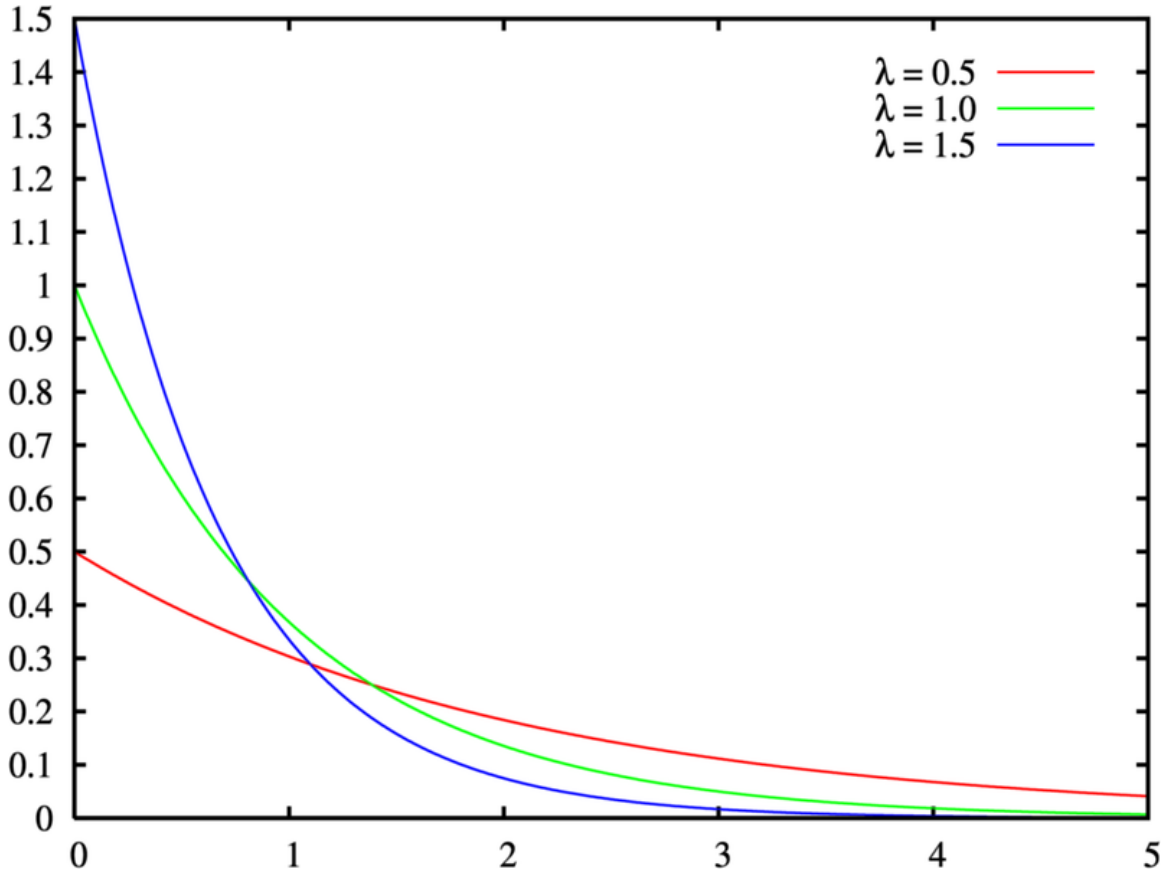
> The total number of failures within an item population, divided by the total time expended by that population, during a particular measurement interval under stated conditions. (MacDiarmid, *et al.*)

Although the failure rate, $\lambda(t)$, is often thought of as the probability that a failure occurs in a specified interval given no failure before time t, it is not actually a probability because it can exceed 1. It can be defined with the aid of the reliability function or survival function $R(t)$, the probability of no failure before time t, as:

$$\lambda = \frac{R(t_1) - R(t_2)}{(t_2 - t_1) \cdot R(t_1)} = \frac{R(t) - R(t + \triangle t)}{\triangle t \cdot R(t)}$$

over a time interval $(t_2 - t_1)$ from t_1 (or t) to t_2 and Δt is defined as $(t_2 - t_1)$. Note that this is a conditional probability, hence the $R(t)$ in the denominator.

Failure rate in the continuous sense



Exponential failure density functions

Calculating the failure rate for ever smaller intervals of time, results in the **hazard function** (or **hazard rate**), *h*(*t*). This becomes the *instantaneous* failure rate as Δttends to zero:

$$h(t) = \lim_{\Delta t \to 0} \frac{R(t) - R(t + \Delta t)}{\Delta t \cdot R(t)}.$$

A continuous failure rate depends on the existence of a **failure distribution**, $F(t)$, which is a cumulative distribution function that describes the probability of failure (at least) up to and including time *t*,

$$\Pr(T \leq t) = F(t) = 1 - R(t), \quad t \geq 0.$$

where *T* is the failure time. The failure distribution function is the integral of the failure *density* function, *f*(*t*),

$$F(t) = \int_0^t f(x)\,dx.$$

The hazard function can be defined now as

$$h(t) = \frac{f(t)}{R(t)}.$$

Many probability distributions can be used to model the failure distribution. A common model is the **exponential failure distribution**,

$$F(t) = \int_0^t \lambda e^{-\lambda x}\,dx = 1 - e^{-\lambda t},$$

which is based on the exponential density function. The hazard rate function for this is:

$$h(t) = \frac{f(t)}{R(t)} = \frac{\lambda e^{-\lambda t}}{e^{-\lambda t}} = \lambda.$$

Thus, for an exponential failure distribution, the hazard rate is a constant with respect to time (that is, the distribution is "memoryless"). For other distributions, such as a Weibull distribution or a log-normal distribution, the hazard function may not be constant with respect to time. For some such as the deterministic distribution it is monotonic increasing (analogous to "wearing out"), for others such as the Pareto distribution it is monotonic decreasing (analogous to "burning in"), while for many it is not monotonic.

Failure rate data

Failure rate data can be obtained in several ways. The most common means are:

- Historical data about the device or system under consideration.

  Many organizations maintain internal databases of failure information on the devices or systems that they produce, which can be used to calculate failure rates for those devices or systems. For new devices or systems, the historical data for similar devices or systems can serve as a useful estimate.

- Government and commercial failure rate data.

  Handbooks of failure rate data for various components are available from government and commercial sources. MIL-HDBK-217F, *Reliability Prediction of Electronic Equipment*, is a military standard that provides failure rate data for many military electronic components. Several failure rate data sources are available commercially that focus on commercial components, including some non-electronic components.

- Testing.

  The most accurate source of data is to test samples of the actual devices or systems in order to generate failure data. This is often prohibitively expensive or impractical, so that the previous data sources are often used instead.

Units

Failure rates can be expressed using any measure of time, but **hours** is the most common unit in practice. Other units, such as miles, revolutions, etc., can also be used in place of "time" units.

Failure rates are often expressed in engineering notation as failures per million, or 10^{-6}, especially for individual components, since their failure rates are often very low.

The **Failures In Time** (**FIT**) rate of a device is the number of failures that can be expected in one billion (10^9) device-hours of operation. (E.g. 1000 devices for 1 million hours, or 1 million devices for 1000 hours each, or some other combination.) This term is used particularly by the semiconductor industry.

Additivity

Under certain engineering assumptions (e.g. besides the above assumptions for a constant failure rate, the assumption that the considered system has no relevant redundancies), the failure rate for a complex system is simply the sum of the individual failure rates of its components, as long as the units are consistent, e.g. failures per million hours. This

permits testing of individual components or subsystems, whose failure rates are then added to obtain the total system failure rate.

Example

Suppose it is desired to estimate the failure rate of a certain component. A test can be performed to estimate its failure rate. Ten identical components are each tested until they either fail or reach 1000 hours, at which time the test is terminated for that component. (The level of statistical confidence is not considered in this example.) The results are as follows:

Estimated failure rate is

$$\frac{6 \text{ failures}}{7502 \text{ hours}} = 0.0007998 \frac{\text{failures}}{\text{hour}} = 799.8 \times 10^{-6} \frac{\text{failures}}{\text{hour}},$$

or 799.8 failures for every million hours of operation.

Estimation

The Nelson–Aalen estimator can be used to estimate the cumulative hazard rate function.

Chapter 3

Safety Engineering

Safety engineering is an applied science strongly related to systems engineering and the subset System Safety Engineering. Safety engineering assures that a life-critical system behaves as needed even when pieces fail.

Overview

Ideally, safety-engineers take an early design of a system, analyze it to find what faults can occur, and then propose safety requirements in design specifications up front and changes to existing systems to make the system safer. In an early design stage, often a fail-safe system can be made acceptably safe with a few sensors and some software to read them. Probabilistic fault-tolerant systems can often be made by using more, but smaller and less-expensive pieces of equipment.

Far too often, rather than actually influencing the design, safety engineers are assigned to prove that an existing, completed design is safe. If a safety engineer then discovers significant safety problems late in the design process, correcting them can be very expensive. This type of error has the potential to waste large sums of money.

The exception to this conventional approach is the way some large government agencies approach safety engineering from a more proactive and proven process perspective, known as "system safety". The system safety philosophy is to be applied to complex and critical systems, such as commercial airliners, complex weapon systems, spacecraft, rail and transportation systems, air traffic control system and other complex and safety-critical industrial systems. The proven system safety methods and techniques are to prevent, eliminate and control hazards and risks through designed influences by a collaboration of key engineering disciplines and product teams. Software safety is a fast growing field since modern systems functionality are increasingly being put under control of software. The whole concept of system safety and software safety, as a subset of systems engineering, is to influence safety-critical systems designs by conducting several types of hazard analyses to identify risks and to specify design safety features and procedures to strategically mitigate risk to acceptable levels before the system is certified.

Additionally, failure mitigation can go beyond design recommendations, particularly in the area of maintenance. There is an entire realm of safety and reliability engineering known as Reliability Centered Maintenance (RCM), which is a discipline that is a direct result of analyzing potential failures within a system and determining maintenance actions that can mitigate the risk of failure. This methodology is used extensively on aircraft and involves understanding the failure modes of the serviceable replaceable assemblies in addition to the means to detect or predict an impending failure. Every automobile owner is familiar with this concept when they take in their car to have the oil changed or brakes checked. Even filling up one's car with fuel is a simple example of a failure mode (failure due to fuel exhaustion), a means of detection (fuel gauge), and a maintenance action (filling the car's fuel tank).

For large scale complex systems, hundreds if not thousands of maintenance actions can result from the failure analysis. These maintenance actions are based on conditions (e.g., gauge reading or leaky valve), hard conditions (e.g., a component is known to fail after 100 hrs of operation with 95% certainty), or require inspection to determine the maintenance action (e.g., metal fatigue). The RCM concept then analyzes each individual maintenance item for its risk contribution to safety, mission, operational readiness, or cost to repair if a failure does occur. Then the sum total of all the maintenance actions are bundled into maintenance intervals so that maintenance is not occurring around the clock, but rather, at regular intervals. This bundling process introduces further complexity, as it might stretch some maintenance cycles, thereby increasing risk, but reduce others, thereby potentially reducing risk, with the end result being a comprehensive maintenance schedule, purpose built to reduce operational risk and ensure acceptable levels of operational readiness and availability.

Analysis techniques

The two most common fault modeling techniques are called failure mode and effects analysis and fault tree analysis. These techniques are just ways of finding problems and of making plans to cope with failures, as in probabilistic risk assessment. One of the earliest complete studies using this technique on a commercial nuclear plant was the WASH-1400 study, also known as the Reactor Safety Study or the Rasmussen Report.

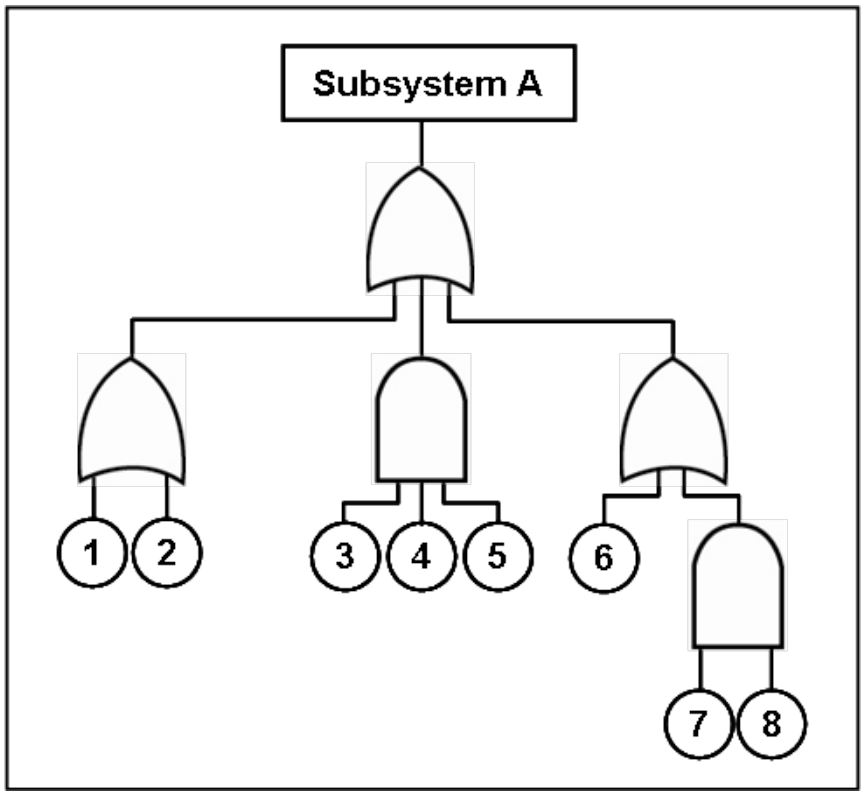
Failure modes and effects analysis

Failure Mode and Effects Analysis (FMEA) is a bottom-up, inductive analytical method which may be performed at either the functional or piece-part level. For functional FMEA, failure modes are identified for each function in a system or equipment item, usually with the help of a functional block diagram. For piece-part FMEA, failure modes are identified for each piece-part component (such as a valve, connector, resistor, or diode). The effects of the failure mode are described, and assigned a probability based on the failure rate and failure mode ratio of the function or component.

Failure modes with identical effects can be combined and summarized in a Failure Mode Effects Summary. When combined with criticality analysis, FMEA is known as Failure Mode, Effects, and Criticality Analysis or FMECA, pronounced "fuh-MEE-kuh".

Fault tree analysis

Fault tree analysis (FTA) is a top-down, deductive analytical method. In FTA, initiating primary events such as component failures, human errors, and external events are traced through Boolean logic gates to an undesired top event such as an aircraft crash or nuclear reactor core melt. The intent is to identify ways to make top events less probable, and verify that safety goals have been achieved.



A fault tree diagram

Fault trees are a logical inverse of success trees, and may be obtained by applying de Morgan's theorem to success trees (which are directly related to reliability block diagrams).

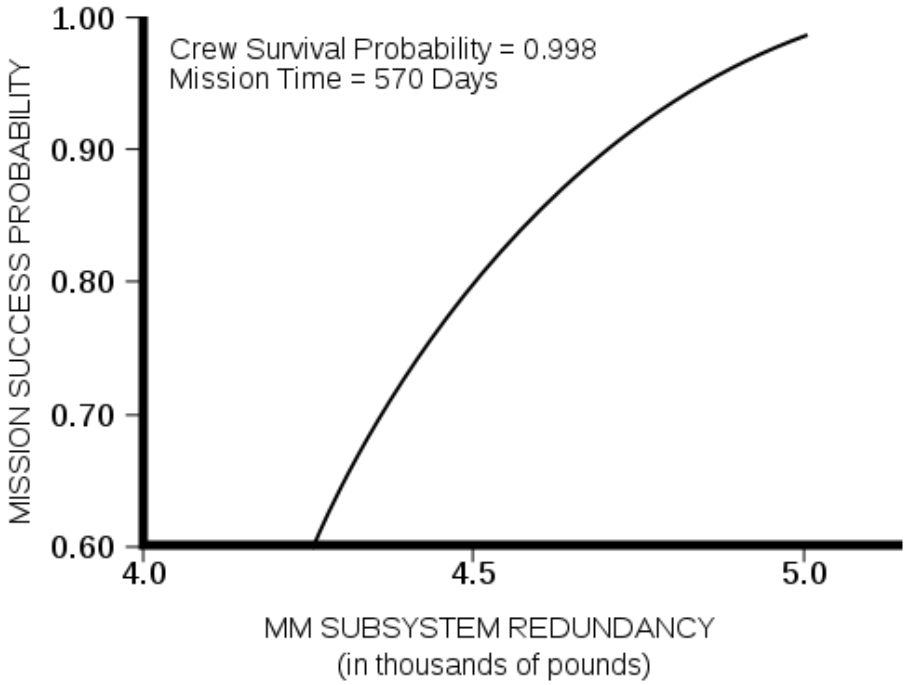
FTA may be qualitative or quantative. When failure and event probabilites are unknown, qualitative fault trees may be analyzed for minimal cut sets. For example, if any minimal cut set contains a single base event, then the top event may be caused by a single failure. Quantitative FTA is used to compute top event probability, and usually requires computer software such as CAFTA from the Electric Power Research Institute or SAPHIRE from the Idaho National Laboratory.

Some industries use both fault trees and event trees. An event tree starts from an undesired initiator (loss of critical supply, component failure etc.) and follows possible further system events through to a series of final consequences. As each new event is considered, a new node on the tree is added with a split of probabilities of taking either branch. The probabilities of a range of "top events" arising from the initial event can then be seen.

Safety certification

Usually a failure in safety-certified systems is acceptable if, on average, less than one life per 10^9 hours of continuous operation is lost to failure. Most Western nuclear reactors, medical equipment, and commercial aircraft are certified to this level. The cost versus loss of lives has been considered appropriate at this level (by FAA for aircraft under Federal Aviation Regulations).

Preventing failure



A NASA graph shows the relationship between the survival of a crew of astronauts and the amount of redundant equipment in their spacecraft (the "MM", Mission Module).

Probabilistic fault tolerance: adding redundancy to equipment and systems

Once a failure mode is identified, it can usually be prevented entirely by adding extra equipment to the system. For example, nuclear reactors contain dangerous radiation, and

nuclear reactions can cause so much heat that no substance might contain them. Therefore reactors have emergency core cooling systems to keep the temperature down, shielding to contain the radiation, and engineered barriers (usually several, nested, surmounted by a containment building) to prevent accidental leakage.

Most biological organisms have a certain amount of redundancy: multiple organs, multiple limbs, etc.

For any given failure, a fail-over or redundancy can almost always be designed and incorporated into a system.

When does safety stop, where does reliability begin?

Inherent fail-safe design

When adding equipment is impractical (usually because of expense), then the least expensive form of design is often "inherently fail-safe". The typical approach is to arrange the system so that ordinary single failures cause the mechanism to shut down in a safe way (for nuclear power plants, this is termed a passively safe design, although more than ordinary failures are covered).

One of the most common fail-safe systems is the overflow tube in baths and kitchen sinks. If the valve sticks open, rather than causing an overflow and damage, the tank spills into an overflow.

Another common example is that in an elevator the cable supporting the car keeps spring-loaded brakes open. If the cable breaks, the brakes grab rails, and the elevator cabin does not fall.

Inherent fail-safes are common in medical equipment, traffic and railway signals, communications equipment, and safety equipment.

Containing failure

It is also common practice to plan for the failure of safety systems through containment and isolation methods. The use of isolating valves, also known as the block and bleed manifold, is very common in isolating pumps, tanks, and control valves that may fail or need routine maintenance. In addition, nearly all tanks containing oil or other hazardous chemicals are required to have containment barriers set up around them to contain 100% of the volume of the tank in the event of a catastrophic tank failure. Similarly, long pipelines have remote-closing valves periodically installed in the line so that in the event of failure, the entire pipeline is not lost. The goal of all such containment systems is to provide means of limiting the damage done by a failure to a small localized area.

Chapter 4

Root Cause Analysis and Fault Tree Analysis

Root Cause Analysis

Root cause analysis (**RCA**) is a class of problem solving methods aimed at identifying the root causes of problems or events. The practice of RCA is predicated on the belief that problems are best solved by attempting to address, correct or eliminate root causes, as opposed to merely addressing the immediately obvious symptoms. By directing corrective measures at root causes, it is more probable that problem recurrence will be prevented. However, it is recognized that complete prevention of recurrence by one corrective action is not always possible. Conversely, there may be several effective measures (methods) that address the root cause of a problem. Thus, RCA is often considered to be an iterative process, and is frequently viewed as a tool of continuous improvement.

RCA, is typically used as a reactive method of identifying event(s) causes, revealing problems and solving them. Analysis is done *after* an event has occurred. Insights in RCA may make it useful as a pro-active method. In that event, RCA can be used to *forecast* or predict probable events even *before* they occur. While one follows the other, RCA is a completely separate process to Incident Management.

Root cause analysis is not a single, sharply defined methodology; there are many different tools, processes, and philosophies for performing RCA analysis. However, several very-broadly defined approaches or "schools" can be identified by their basic approach or field of origin: safety-based, production-based, process-based, failure-based, and systems-based.

- Safety-based RCA descends from the fields of accident analysis and occupational safety and health.
- Production-based RCA has its origins in the field of quality control for industrial manufacturing.
- Process-based RCA is basically a follow-on to production-based RCA, but with a scope that has been expanded to include business processes.

- Failure-based RCA is rooted in the practice of failure analysis as employed in engineering and maintenance.
- Systems-based RCA has emerged as an amalgamation of the preceding schools, along with ideas taken from fields such as change management, risk management, and systems analysis.

Despite the different approaches among the various schools of root cause analysis, there are some common principles. It is also possible to define several general processes for performing RCA.

General principles of root cause analysis

1. The primary aim of RCA is to identify the root cause(s) of a problem in order to create effective corrective actions that will prevent that problem from ever re-occurring, otherwise addressing the problem with virtual certainty of success. ("Success" is defined as the near-certain prevention of recurrence.)
2. To be effective, RCA must be performed systematically, usually as part of an investigation, with conclusions and root causes identified backed up by documented evidence. Usually a team effort is required.
3. There may be more than one root cause for an event or a problem, the difficult part is demonstrating the persistence and sustaining the effort required to develop them.
4. The purpose of identifying all solutions to a problem is to prevent recurrence at lowest cost in the simplest way. If there are alternatives that are equally effective, then the simplest or lowest cost approach is preferred.
5. Root causes identified depend on the way in which the problem or event is defined. Effective problem statements and event descriptions (as failures, for example) are helpful, or even required.
6. To be effective the analysis should establish a sequence of events or timeline to understand the relationships between contributory (causal) factors, root cause(s) and the defined problem or event to prevent in the future.
7. Root cause analysis can help to transform an reactive culture (that reacts to problems) into a forward-looking culture that solves problems before they occur or escalate. More importantly, it reduces the frequency of problems occurring over time within the environment where the RCA process is used.
8. RCA is a threat to many cultures and environments. Threats to cultures often meet with resistance. There may be other forms of management support required to achieve RCA effectiveness and success. For example, and "non-punitory" policy towards problem identifiers may be required.

General process for performing and documenting an RCA-based Corrective Action

Notice that RCA (in steps 3, 4 and 5) forms the most critical part of successful corrective action, because it directs the corrective action at the true root cause of the problem. The

root cause is secondary to the goal of prevention, but without knowing the root cause, we cannot determine what an effective corrective action for the defined problem will be.

1. Define the problem or describe the event factually
2. Gather data and evidence, classifying that along a timeline of events to the final failure or crisis.
3. Ask "why" and identify the causes associated with each step in the sequence towards the defined problem or event.
4. Classify causes into causal factors that relate to an event in the sequence, and root causes, that if applied can be agreed to have interrputed that step of the sequence chain.
5. If there are multiple root causes, which is often the case, reveal those clearly for later optimum selection.
6. Identify corrective action(s) that will prevent absolutely with certainty prevent recurrence of the problem or event. These can be used to select the best correction action, later
7. Identify solutions that effective, prevent recurrence with reasonable certainty with consensus agreement of the group, are within your control, meet your goals and objectives and do not cause introduce other new, unforeseen problems.
8. Implement the recommended root cause correction(s).
9. Ensure effectiveness by observing the implemented recommendation solutions.
10. Other methodologies for problem solving and problem avoidance may be useful.

Root cause analysis techniques

- Barrier analysis - a technique often used in process industries. It is based on tracing energy flows, with a focus on barriers to those flows, to identify how and why the barriers did not prevent the energy flows from causing harm.
- Bayesian inference
- Causal factor tree analysis - a technique based on displaying causal factors in a tree-structure such that cause-effect dependencies are clearly identified.
- Change analysis - an investigation technique often used for problems or accidents. It is based on comparing a situation that does not exhibit the problem to one that does, in order to identify the changes or differences that might explain why the problem occurred.
- Current Reality Tree - A method developed by Eliahu M. Goldratt in his theory of constraints that guides an investigator to identify and relate all root causes using a cause-effect tree whose elements are bound by rules of logic (Categories of Legitimate Reservation). The CRT begins with a brief list of the undesirables things we see around us, and then guides us towards one or more root causes. This method is particularly powerful when the system is complex, there is no obvious link between the observed undesirable things, and a deep understanding of the root cause(s) is desired.
- Failure modes and effects analysis
- Fault tree analysis
- 5 Whys ask why why why why over until exhausted

- Ishikawa diagram, also known as the fishbone diagram or cause-and-effect diagram. The Ishikawa diagram is the one method for project managers for conducting RCA. It's effective due to its simplicity, ability to resolve inexact information, ability to allow group participation and the complexity of the rest of the methods.
- Pareto analysis "80/20 rule"
- RPR Problem Diagnosis - An ITIL-aligned method for diagnosing IT problems.
- Kepner-Tregoe Approach

Common cause analysis (CCA) common modes analysis (CMA) are evolving engineering techniques for complex technical systems to determine if common root causes in hardware, software or highly integrated systems interaction may contribute to human error or improper operation of a system. Systems are analyzed for root causes and causal factors to determine probability of failure modes, fault modes, or common mode software faults due to escaped requirements. Also ensuring complete testing and verification are methods used for ensuring complex systems are designed with no common causes that cause severe hazards. Common cause analysis are sometimes required as part of the safety engineering tasks for theme parks, commercial/military aircraft, spacecraft, complex control systems, large electrical utility grids, nuclear power plants, automated industrial controls, medical devices or other safety safety-critical systems with complex functionality.

Basic elements of root cause using Management Oversight Risk Tree (MORT) Approach Classification

- Materials
  - Defective raw material
  - Wrong type for job
  - Lack of raw material
- Man Power
  - Inadequate capability
  - Lack of Knowledge
  - Lack of skill
  - Stress
  - Improper motivation
- Machine / Equipment
  - Incorrect tool selection
  - Poor maintenance or design
  - Poor equipment or tool placement
  - Defective equipment or tool
- Environment
  - Orderly workplace
  - Job design or layout of work
  - Surfaces poorly maintained
  - Physical demands of the task
  - Forces of nature

- Management
  - No or poor management involvement
  - Inattention to task
  - Task hazards not guarded properly
  - Other (horseplay, inattention....)
  - Stress demands
  - Lack of Process
  - Lack of Communication
- Methods
  - No or poor procedures
  - Practices are not the same as written procedures
  - Poor communication
- Management system
  - Training or education lacking
  - Poor employee involvement
  - Poor recognition of hazard
  - Previously identified hazards were not eliminated

Fault Tree Analysis

Fault tree analysis (FTA) is a failure analysis in which an undesired state of a system is analyzed using boolean logic to combine a series of lower-level events. This analysis method is mainly used in the field of safety engineering to quantitatively determine the probability of a safety hazard.

History

Fault Tree Analysis (FTA) was originally developed in 1962 at Bell Laboratories by H.A. Watson, under a U.S. Air Force Ballistics Systems Division contract to evaluate the Minuteman I Intercontinental Ballistic Missile (ICBM) Launch Control System. Following the first published use of FTA in the 1962 Minuteman I Launch Control Safety Study, Boeing and AVCO expanded use of FTA to the entire Minuteman II system in 1963-1964. FTA received extensive coverage at a 1965 System Safety Symposium in Seattle sponsored by Boeing and the University of Washington. Boeing began using FTA for civil aircraft design around 1966. In 1970, the U.S. Federal Aviation Administration (FAA) published a change to 14 CFR 25.1309 airworthiness regulations for transport aircraft in the Federal Register at 35 FR 5665 (1970-04-08). This change adopted failure probability criteria for aircraft systems and equipment and led to widespread use of FTA in civil aviation.

Within the nuclear power industry, the U.S. Nuclear Regulatory Commission began using probabilistic risk assessment (PRA) methods including FTA in 1975, and significantly

expanded PRA research following the 1979 incident at Three Mile Island. This eventually led to the 1981 publication of the NRC Fault Tree Handbook NUREG–0492, and mandatory use of PRA under the NRC's regulatory authority.

Fault Tree Analysis (FTA) attempts to model and analyze failure processes of engineering and biological systems. FTA is basically composed of logic diagrams that display the state of the system and is constructed using graphical design techniques. Originally, engineers were responsible for the development of Fault Tree Analysis, as a deep knowledge of the system under analysis is required.

Often, FTA is defined as another part, or technique, of reliability engineering. Although both model the same major aspect, they have arisen from two different perspectives. Reliability engineering was, for the most part, developed by mathematicians, while FTA, as stated above, was developed by engineers.

Fault Tree Analysis usually involves events from hardware wear out, material failure or malfunctions or combinations of deterministic contributions to the event stemming from assigning a hardware/system failure rate to branches or cut sets. Typically failure rates are carefully derived from substantiated historical data such as mean time between failure of the components, unit, subsystem or function. Predictor data may be assigned. Assigning a software failure rate is elusive and not possible. Since software is a vital contributor and inclusive of the system operation it is assumed the software will function normally as intended. There is no such thing as a software fault tree unless considered in the system context. Software is an instruction set to the hardware or overall system for correct operation. Since basic software events do not fail in the physical sense, attempting to predict manifestation of software faults or coding errors with any reliability or accuracy is impossible, unless assumptions are made. Predicting and assigning human error rates is not the primary intent of a fault tree analysis, but may be attempted to gain some knowledge of what happens with improper human input or intervention at the wrong time.

FTA can be used as a valuable design tool, can identify potential accidents, and can eliminate costly design changes. It can also be used as a diagnostic tool, predicting the most likely system failure in a system breakdown. FTA is used in safety engineering and in all major fields of engineering.

Methodology

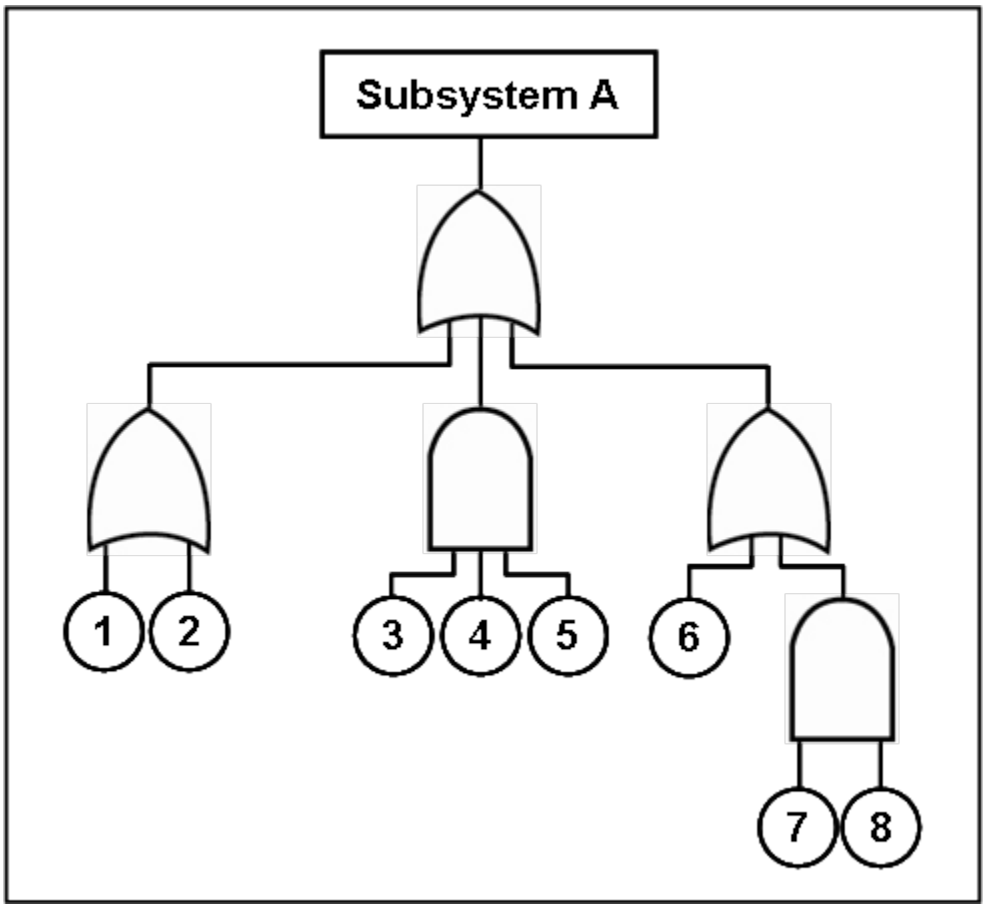
FTA methodology is described in several industry and government standards, including NRC NUREG–0492 for the nuclear power industry, an aerospace-oriented revision to NUREG–0492 for use by NASA, SAE ARP4761 for civil aerospace, MIL–HDBK–338 for military systems for military systems. IEC standard IEC 61025 is intended for cross-industry use and has been adopted as European Norme EN 61025.

Since no system is perfect, dealing with a subsystem fault is a necessity, and any working system eventually will have a fault in some place. However, the probability for a

complete or partial success is greater than the probability of a complete failure or partial failure. Assembling a FTA is thus not as tedious as assembling a success tree which can turn out to be very time consuming.

Because assembling a FTA can be a costly and cumbersome experience, the perfect method is to consider subsystems. In this way dealing with smaller systems can assure less error work probability, less system analysis. Afterward, the subsystems integrate to form the well analyzed big system.

An undesired effect is taken as the root ('top event') of a tree of logic. There should be only one Top Event and all concerns must tree down from it. Then, each situation that could cause that effect is added to the tree as a series of logic expressions. When fault trees are labeled with actual numbers about failure probabilities (which are often in practice unavailable because of the expense of testing), computer programs can calculate failure probabilities from fault trees.



A fault tree diagram

The Tree is usually written out using conventional logic gate symbols. The route through a tree between an event and an initiator in the tree is called a Cut Set. The shortest credible way through the tree from fault to initiating event is called a Minimal Cut Set.

Some industries use both Fault Trees and Event Trees. An Event Tree starts from an undesired initiator (loss of critical supply, component failure etc.) and follows possible further system events through to a series of final consequences. As each new event is considered, a new node on the tree is added with a split of probabilities of taking either branch. The probabilities of a range of 'top events' arising from the initial event can then be seen.

Classic programs include the Electric Power Research Institute's (EPRI) CAFTA software, which is used by many of the US nuclear power plants and by a majority of US and international aerospace manufacturers, and the Idaho National Laboratory's SAPHIRE, which is used by the U.S. Government to evaluate the safety and reliability of nuclear reactors, the Space Shuttle, and the International Space Station. Outside the US, the software RiskSpectrum is a popular tool for Fault Tree and Event Tree analysis and is licensed for use at almost half of the worlds nuclear power plants for Probabilistic Safety Assessment.

Analysis

Many different approaches can be used to model a FTA, but the most common and popular way can be summarized in a few steps. Remember that a fault tree is used to analyze a single fault event, and that one and only one event can be analyzed during a single fault tree. Even though the “fault” may vary dramatically, a FTA follows the same procedure for an event, be it a delay of 0.25 msec for the generation of electrical power, or the random, unintended launch of an ICBM.

FTA analysis involves five steps:

1. Define the undesired event to study
    - Definition of the undesired event can be very hard to catch, although some of the events are very easy and obvious to observe. An engineer with a wide knowledge of the design of the system or a system analyst with an engineering background is the best person who can help define and number the undesired events. Undesired events are used then to make the FTA, one event for one FTA; no two events will be used to make one FTA.
2. Obtain an understanding of the system
    - Once the undesired event is selected, all causes with probabilities of affecting the undesired event of 0 or more are studied and analyzed. Getting exact numbers for the probabilities leading to the event is usually impossible for the reason that it may be very costly and time consuming to do so. Computer software is used to study probabilities; this may lead to less costly system analysis.
      System analysts can help with understanding the overall system. System designers have full knowledge of the system and this knowledge is very important for not missing any cause affecting the undesired event. For the selected event all causes are then numbered and sequenced in the order of

occurrence and then are used for the next step which is drawing or constructing the fault tree.

3. Construct the fault tree
    - After selecting the undesired event and having analyzed the system so that we know all the causing effects (and if possible their probabilities) we can now construct the fault tree. Fault tree is based on AND and OR gates which define the major characteristics of the fault tree.
4. Evaluate the fault tree
    - After the fault tree has been assembled for a specific undesired event, it is evaluated and analyzed for any possible improvement or in other words study the risk management and find ways for system improvement. This step is as an introduction for the final step which will be to control the hazards identified. In short, in this step we identify all possible hazards affecting in a direct or indirect way the system.
5. Control the hazards identified
    - This step is very specific and differs largely from one system to another, but the main point will always be that after identifying the hazards all possible methods are pursued to decrease the probability of occurrence.

Comparison with other analytical methods

FTA is a deductive, top-down method aimed at analyzing the effects of initiating faults and events on a complex system. This contrasts with failure mode and effects analysis (FMEA), which is an inductive, bottom-up analysis method aimed at analyzing the effects of single component or function failures on equipment or subysystems. FTA is very good at showing how resistant a system is to single or multiple initiating faults. It is not good at finding all possible initiating faults. FMEA is good at exhaustively cataloging initiating faults, and identifying their local effects. It is not good at examining multiple failures or their effects at a system level. FTA considers external events, FMEA does not. In civil aerospace the usual practice is to perform both FTA and FMEA, with a failure mode effects summary (FMES) as the interface between FMEA and FTA.

Alternatives to FTA include dependence diagram (DD), also known as reliability block diagram (RBD) and Markov analysis. A dependence diagram is equivalent to a success tree analysis (STA), the logical inverse of an FTA, and depicts the system using paths instead of gates. DD and STA produce probability of success (i.e., avoiding a top event) rather than probability of a top event.

Chapter 5

Fault-tolerant Design

In engineering, **fault-tolerant design**, also known as **fail-safe design**, is a design that enables a system to continue operation, possibly at a reduced level (also known as graceful degradation), rather than failing completely, when some part of the system fails. The term is most commonly used to describe computer-based systems designed to continue more or less fully operational with, perhaps, a reduction in throughput or an increase in response time in the event of some partial failure. That is, the system as a whole is not stopped due to problems either in the hardware or the software. An example in another field is a motor vehicle designed so it will continue to be drivable if one of the tires is punctured. A structure is able to retain its integrity in the presence of damage due to causes such as fatigue, corrosion, manufacturing flaws, or impact.

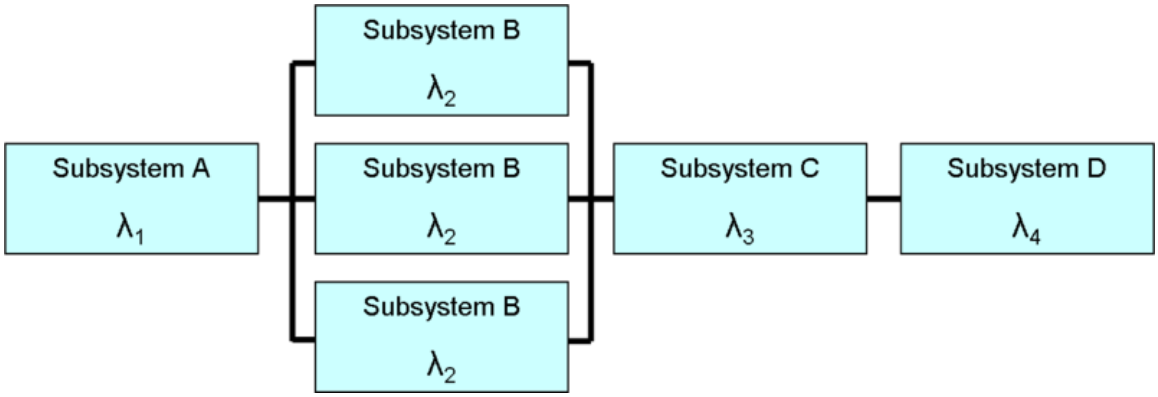
Components

If each component, in turn, can continue to function when one of its subcomponents fails, this will allow the total system to continue to operate, as well. Using a passenger vehicle as an example, a car can have "run-flat" tires, which each contain a solid rubber core, allowing them to be used even if a tire is punctured. The punctured "run-flat" tire may be used for a limited time at a reduced speed.

Redundancy

This means having backup components which automatically "kick in" should one component fail. For example, large cargo trucks can lose a tire without any major consequences. They have many tires, and no one tire is critical (with the exception of the front tires, which are used to steer).

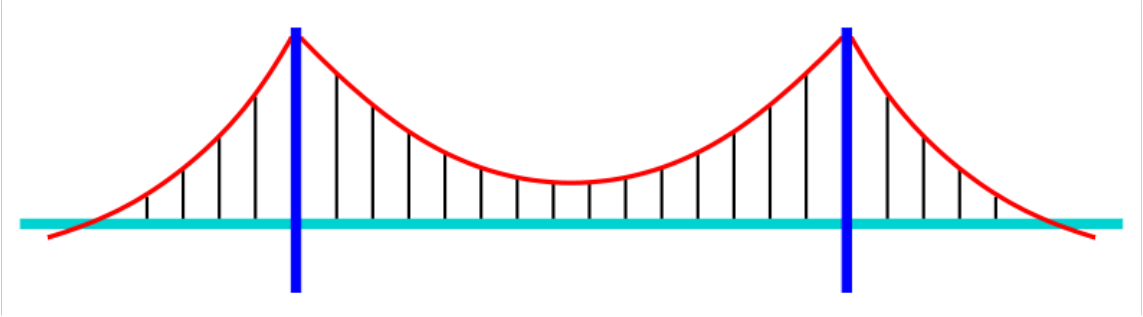
Redundant power supply



Redundant subsystem "B"

In engineering, **redundancy** is the duplication of critical components of a system with the intention of increasing reliability of the system, usually in the case of a backup or fail-safe.

In many safety-critical systems, such as fly-by-wire and hydraulic systems in aircraft, some parts of the control system may be triplicated, which is formally termed triple modular redundancy (TMR). An error in one component may then be out-voted by the other two. In a triply redundant system, the system has three sub components, all three of which must fail before the system fails. Since each one rarely fails, and the sub components are expected to fail independently, the probability of all three failing is calculated to be extremely small. Redundancy may also be known by the terms "**majority voting systems**" or "**voting logic**".



A Suspension Bridge's numerous cables are a form of redundancy

Forms of redundancy

There are four major forms of redundancy, these are:

- Hardware redundancy, such as DMR and TMR
- Information redundancy, such as Error detection and correction methods
- Time redundancy, including transient fault detection methods such as **Alternate Logic**
- Software redundancy such as N-version programming

Function of redundancy

The two functions of redundancy are passive redundancy and active redundancy. Both functions prevent performance decline from exceeding specification limits without human intervention using extra capacity.

Passive redundancy uses excess capacity to reduce the impact of component failures. One common form of passive redundancy is the extra strength of cabling and struts used in bridges. This extra strength allows some structural components to fail without bridge collapse. The extra strength used in the design is called the margin of safety.

Eyes and ears provide working examples of passive redundancy. Vision loss in one eye does not cause blindness but depth depth perception is impaired. Hearing loss in one ear does not cause deafness but directionality is impaired. Performance decline is commonly associated with passive redundancy when a limited number of failures occur.

Active redundancy eliminates performance decline by monitoring performance of individual device, and this monitoring is used in voting logic. The voting logic is linked to switching that automatically reconfigures components. Error detection and correction and the Global Positioning System (GPS) are two examples of active redundancy.

Electrical power distribution provides an example of active redundancy. Several power lines connect each generation facility with customers. Each power line include monitors that detect overload. Each power line also includes circuit breakers. The combination of power lines provides excess capacity. Circuit breakers disconnect a power line when monitors detect an overload. Power is redistributed across the remaining lines.

Voting Logic

Voting logic uses performance monitoring to determine how to reconfigure individual components so that operation continues without violating specification limitations of the overall system. Voting logic often involve computers, but systems composed of items other than computers may be reconfigured using voting logic. Circuit breakers are an example of a form of non-computer voting logic.

Electrical power systems use power scheduling to reconfigure active redundancy. Computing systems adjust the production output of each generating facility when other generating facilities are suddenly lost. This prevents blackout conditions during major events like earthquake.

The simplest voting logic in computing systems involves two components: primary and alternate. They both run similar software, but the output from the alternate remains inactive during normal operation. The primary monitors itself and periodically sends an activity message to the alternate as long as everything is OK. All outputs from the primary stop, including the activity message, when the primary detects a fault. The alternate activates its output and takes over from the primary after a brief delay when the activity message ceases. Errors in voting logic can cause both to have all outputs active at the same time, can cause both to have all outputs inactive at the same time, or outputs can flutter on and off.

A more reliable form of voting logic involves an odd number of 3 devices or more. All perform identical functions and the outputs are compared by the voting logic. The voting logic establishes a majority when there is a disagreement, and the majority will act to deactivate the output from other device(s) that disagree. A single fault will not interrupt normal operation. This technique is used with avionics systems, such as those responsible for operation of the space shuttle.

Calculating the probability of system failure

Each duplicate component added to the system decreases the probability of system failure according to the formula:

$$p = \prod_{i=1}^{n} p_i$$

where:

- n - number of components
- p_i - probability of component i failing
- p - the probability of all components failing (system failure)

This formula assumes independence of failure events. That means that the probability of a component B failing given that a component A has already failed is the same as that of B failing when A has not failed. There are situations where this is unreasonable, such as using two power supplies connected to the same socket, whereby if one socket failed, the other would too.

It also assumes that at only one component is needed to keep the system running. If m components are needed for the system to survive, out of n, the probability of failure is

$P = 1 - ((1-p)^{(n-m)} C_n^m)$, Assuming all components have equal probability, p, of failure

This model is probably unrealistic in that it assumes that components are not replaced in time when they fail.

When to use

Providing fault-tolerant design for every component is normally not an option. In such cases the following criteria may be used to determine which components should be fault-tolerant:

- **How critical is the component?** In a car, the radio is not critical, so this component has less need for fault-tolerance.
- **How likely is the component to fail?** Some components, like the drive shaft in a car, are not likely to fail, so no fault-tolerance is needed.
- **How expensive is it to make the component fault-tolerant?** Requiring a redundant car engine, for example, would likely be too expensive both economically and in terms of weight and space, to be considered.

An example of a component that passes all the tests is a car's occupant restraint system. While we do not normally think of the *primary* occupant restraint system, it is gravity. If the vehicle rolls over or undergoes severe g-forces, then this primary method of occupant restraint may fail. Restraining the occupants during such an accident is absolutely critical to safety, so we pass the first test. Accidents causing occupant ejection were quite common before seat belts, so we pass the second test. The cost of a redundant restraint method like seat belts is quite low, both economically and in terms or weight and space, so we pass the third test. Therefore, adding seat belts to all vehicles is an excellent idea. Other "supplemental restraint systems", such as airbags, are more expensive and so pass that test by a smaller margin.

Examples

Hardware fault-tolerance sometimes requires that broken parts can be swapped out with new ones while the system is still operational (in computing known as *hot swapping*). Such a system implemented with a single backup is known as **single point tolerant**, and represents the vast majority of fault-tolerant systems. In such systems the mean time between failures should be long enough for the operators to have time to fix the broken devices (mean time to repair) before the backup also fails. It helps if the time between failures is as long as possible, but this is not specifically required in a fault-tolerant system.

Fault-tolerance is notably successful in computer applications. Tandem Computers built their entire business on such machines, which used single point tolerance to create their **NonStop** systems with uptimes measured in years.

Fail-safe architectures may encompass also the computer software, for example by process replication (computer science).

Disadvantages

Fault-tolerant design's advantages are obvious, while many of its disadvantages are not:

- **Interference with fault detection in the same component.** To continue the above passenger vehicle example, it may not be obvious to the driver when a tire has been punctured, with either of the fault-tolerant systems. This is usually handled with a separate "automated fault detection system". In the case of the tire, an air pressure monitor detects the loss of pressure and notifies the driver. The alternative is a "manual fault detection system", such as manually inspecting all tires at each stop.

- **Interference with fault detection in another component.** Another variation of this problem is when fault-tolerance in one component prevents fault detection in a different component. For example, if component B performs some operation based on the output from component A, then fault-tolerance in B can hide a problem with A. If component B is later changed (to a less fault-tolerant design)

the system may fail suddenly, making it appear that the new component B is the problem. Only after the system has been carefully scrutinized will it become clear that the root problem is actually with component A.

- **Reduction of priority of fault correction.** Even if the operator is aware of the fault, having a fault-tolerant system is likely to reduce the importance of repairing the fault. If the faults are not corrected, this will eventually lead to system failure, when the fault-tolerant component fails completely or when all redundant components have also failed.

- **Test difficulty.** For certain critical fault-tolerant systems, such as a nuclear reactor, there is no easy way to verify that the backup components are functional. The most infamous example of this is Chernobyl, where operators tested the emergency backup cooling by disabling primary and secondary cooling. The backup failed, resulting in a core meltdown and massive release of radiation.

- **Cost.** Both fault-tolerant components and redundant components tend to increase cost. This can be a purely economic cost or can include other measures, such as weight. Manned spaceships, for example, have so many redundant and fault-tolerant components that their weight is increased dramatically over unmanned systems, which don't require the same level of safety.

- **Inferior components.** A fault-tolerant design may allow for the use of inferior components, which would have otherwise made the system inoperable. While this practice has the potential to mitigate the cost increase, use of multiple inferior components may lower the reliability of the system to a level equal to, or even worse than, a comparable non-fault-tolerant system.

Related terms

There is a difference between fault-tolerance and systems that rarely have problems. For instance, the Western Electric crossbar systems had failure rates of two hours per forty years, and therefore were highly *fault resistant*. But when a fault did occur they still stopped operating completely, and therefore were not *fault-tolerant*.

Chapter 6

Fault-tolerant System

Fault-tolerance or **graceful degradation** is the property that enables a system (often computer-based) to continue operating properly in the event of the failure of (or one or more faults within) some of its components. If its operating quality decreases at all, the decrease is proportional to the severity of the failure, as compared to a naïvely-designed system in which even a small failure can cause total breakdown. Fault-tolerance is particularly sought-after in high-availability or life-critical systems.

Fault-tolerance is not just a property of individual machines; it may also characterise the rules by which they interact. For example, the Transmission Control Protocol (TCP) is designed to allow reliable two-way communication in a packet-switched network, even in the presence of communications links which are imperfect or overloaded. It does this by requiring the endpoints of the communication to *expect* packet loss, duplication, reordering and corruption, so that these conditions do not damage data integrity, and only reduce throughput by a proportional amount.

An example of graceful degradation by design in an image with transparency. The top two images are each the result of viewing the composite image in a viewer that recognises transparency. The bottom two images are the result in a viewer with no

support for transparency. Because the transparency mask (centre bottom) is discarded, only the overlay (centre top) remains; the image on the left has been designed to degrade gracefully, hence is still meaningful without its transparency information.

Data formats may also be designed to degrade gracefully. HTML for example, is designed to be forward compatible, allowing new HTML entities to be ignored by Web browsers which do not understand them without causing the document to be unusable.

Recovery from errors in fault-tolerant systems can be characterised as either **roll-forward** or **roll-back**. When the system detects that it has made an error, roll-forward recovery takes the system state at that time and corrects it, to be able to move forward. Roll-back recovery reverts the system state back to some earlier, correct version, for example using checkpointing, and moves forward from there. Roll-back recovery requires that the operations between the checkpoint and the detected erroneous state can be made idempotent. Some systems make use of both roll-forward and roll-back recovery for different errors or different parts of one error.

Within the scope of an *individual* system, fault-tolerance can be achieved by anticipating exceptional conditions and building the system to cope with them, and, in general, aiming for self-stabilization so that the system converges towards an error-free state. However, if the consequences of a system failure are catastrophic, or the cost of making it sufficiently reliable is very high, a better solution may be to use some form of duplication. In any case, if the consequence of a system failure is catastrophic, the system must be able to use reversion to fall back to a safe mode. This is similar to roll-back recovery but can be a human action if humans are present in the loop.

Fault tolerance requirements

The basic characteristics of fault tolerance require:

1. No single point of repair
2. Fault isolation to the failing component
3. Fault containment to prevent propagation of the failure
4. Availability of reversion modes

In addition, fault tolerant systems are characterized in terms of both planned service outages and unplanned service outages. These are usually measured at the application level and not just at a hardware level. The figure of merit is called availability and is expressed as a percentage. For example, a five nines system would statistically provide 99.999% availability.

Fault-tolerant systems are typically based on the concept of redundancy.

Fault-tolerance by replication

Spare components addresses the first fundamental characteristic of fault-tolerance in three ways:

- Replication: Providing multiple identical instances of the same system or subsystem, directing tasks or requests to all of them in parallel, and choosing the correct result on the basis of a quorum;
- Redundancy: Providing multiple identical instances of the same system and switching to one of the remaining instances in case of a failure (failover);
- Diversity: Providing multiple *different* implementations of the same specification, and using them like replicated systems to cope with errors in a specific implementation.

All implementations of RAID, redundant array of independent disks, except RAID 0 are examples of a fault-tolerant storage device that uses data redundancy.

A lockstep fault-tolerant machine uses replicated elements operating in parallel. At any time, all the replications of each element should be in the same state. The same inputs are provided to each replication, and the same outputs are expected. The outputs of the replications are compared using a voting circuit. A machine with two replications of each element is termed Dual Modular Redundant (DMR). The voting circuit can then only detect a mismatch and recovery relies on other methods. A machine with three replications of each element is termed Triple Modular Redundancy (TMR). The voting circuit can determine which replication is in error when a two-to-one vote is observed. In this case, the voting circuit can output the correct result, and discard the erroneous version. After this, the internal state of the erroneous replication is assumed to be different from that of the other two, and the voting circuit can switch to a DMR mode. This model can be applied to any larger number of replications.

Lockstep fault tolerant machines are most easily made fully synchronous, with each gate of each replication making the same state transition on the same edge of the clock, and the clocks to the replications being exactly in phase. However, it is possible to build lockstep systems without this requirement.

Bringing the replications into synchrony requires making their internal stored states the same. They can be started from a fixed initial state, such as the reset state. Alternatively, the internal state of one replica can be copied to another replica.

One variant of DMR is **pair-and-spare**. Two replicated elements operate in lockstep as a pair, with a voting circuit that detects any mismatch between their operations and outputs a signal indicating that there is an error. Another pair operates exactly the same way. A final circuit selects the output of the pair that does not proclaim that it is in error. Pair-and-spare requires four replicas rather than the three of TMR, but has been used commercially.

No single point of repair

If a system experiences a failure, it must continue to operate without interruption during the repair process.

Fault isolation to the failing component

When a failure occurs, the system must be able to isolate the failure to the offending component. This requires the addition of dedicated failure detection mechanisms that exist only for the purpose of fault isolation.

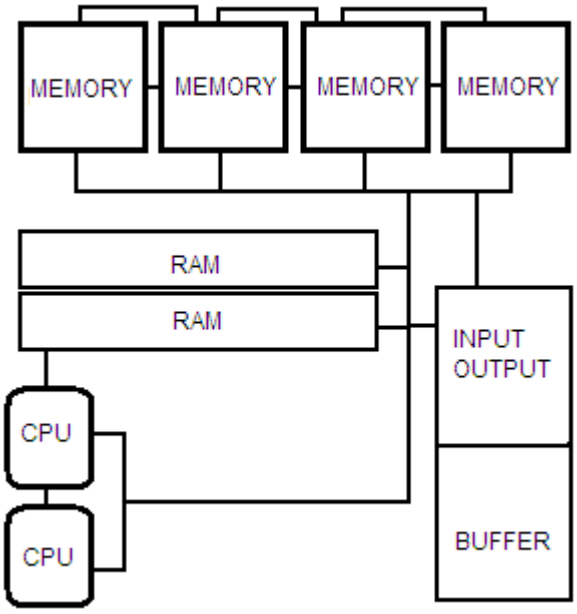
Recovery from a fault condition requires classifying the fault or failing component. The National Institute of Standards and Technology (NIST) categorizes faults based on Locality, Cause, Duration and Effect.

Fault containment

Some failure mechanisms can cause a system to fail by propagating the failure to the rest of the system. An example of this kind of failure is the "Rogue transmitter" which can swamp legitimate communication in a system and cause overall system failure. Mechanisms that isolate a rogue transmitter or failing component to protect the system are required.

Specimen of Fault –tolerant system

Fault-tolerant computer system



A conceptual design of a segregated-component fault-tolerant computer design

Fault-tolerant computer systems are systems designed around the concepts of fault tolerance. In essence, they have to be able to keep working to a level of satisfaction in the presence of faults.

Types of fault tolerance

Most fault-tolerant computer systems are designed to be able to handle several possible failures, including hardware-related faults such as hard disk failures, input or output device failures, or other temporary or permanent failures; software bugs and errors; interface errors between the hardware and software, including driver failures; operator errors, such as erroneous keystrokes, bad command sequences, or installing unexpected software; and physical damage or other flaws introduced to the system from an outside source.

Hardware fault-tolerance is the most common application of these systems, designed to prevent failures due to hardware components. Typically, components have multiple backups and are separated into smaller "segments" that act to contain a fault, and extra redundancy is built into all physical connectors, power supplies, fans, etc. There are special software and instrumentation packages designed to detect failures, such as fault masking, which is a way to ignore faults by seamlessly preparing a backup component to

execute something as soon as the instruction is sent, using a sort of voting protocol where if the main and backups don't give the same results, the flawed output is ignored.

Software fault-tolerance is based more around nullifying programming errors using real-time redundancy, or static "emergency" subprograms to fill in for programs that crash. There are many ways to conduct such fault-regulation, depending on the application and the available hardware.

History

The first known fault-tolerant computer was SAPO, built in 1951 in Czechoslovakia by Antonin Svoboda. Its basic design was magnetic drums connected via relays, with a voting method of memory error detection. Several other machines were developed along this line, mostly for military use. Eventually, they separated into three distinct categories: machines that would last a long time without any maintenance, such as the ones used on NASA space probes and satellites; computers that were very dependable but required constant monitoring, such as those used to monitor and control nuclear power plants or supercollider experiments; and finally, computers with a high amount of runtime which would be under heavy use, such as many of the supercomputers used by insurance companies for their probability monitoring.

Most of the development in the so called LLNM (Long Life, No Maintenance) computing was done by NASA during the 1960s, in preparation for Project Apollo and other research aspects. NASA's first machine went into a space observatory, and their second attempt, the JSTAR computer, was used in Voyager. This computer had a backup of memory arrays to use memory recovery methods and thus it was called the JPL Self-Testing-And-Repairing computer. It could detect its own errors and fix them or bring up redundant modules as needed. The computer is still working today.

Hyper-dependable computers were pioneered mostly by aircraft manufacturers, nuclear power companies, and the railroad industry in the USA. These needed computers with massive amounts of uptime that would fail gracefully enough with a fault to allow continued operation, while relying on the fact that the computer output would be constantly monitored by humans to detect faults. Again, IBM developed the first computer of this kind for NASA for guidance of Saturn V rockets, but later on BNSF, Unisys, and General Electric built their own.

In general, the early efforts at fault-tolerant designs were focused mainly on internal diagnosis, where a fault would indicate something was failing and a worker could replace it. SAPO, for instance, had a method by which faulty memory drums would emit a noise before failure. Later efforts showed that, to be fully effective, the system had to be self-repairing and diagnosing – isolating a fault and then implementing a redundant backup while alerting a need for repair. This is known as N-model redundancy, where faults cause automatic fail safes and a warning to the operator, and it is still the most common form of level one fault-tolerant design in use today.

Voting was another initial method, as discussed above, with multiple redundant backups operating constantly and checking each other's results, with the outcome that if, for example, four components reported an answer of 5 and one component reported an answer of 6, the other four would "vote" that the fifth component was faulty and have it taken out of service. This is called M out of N majority voting.

Historically, motion has always been to move further from N-model and more to M out of N due to the fact that the complexity of systems and the difficulty of ensuring the transitive state from fault-negative to fault-positive did not disrupt operations.

Fault tolerance verification and validation

The most important requirement of design in a fault tolerant computer system is making sure it actually meets its requirements for reliability. This is done by using various failure models to simulate various failures, and analyzing how well the system reacts. These statistical models are very complex, involving probability curves and specific fault rates, latency curves, error rates, and the like. The most commonly used models are HARP, SAVE, and SHARPE in the USA, and SURF or LASS in Europe.

Fault tolerance research

Research into the kinds of tolerances needed for critical systems involves a large amount of interdisciplinary work. The more complex the system, the more carefully all possible interactions have to be considered and prepared for. Considering the importance of high-value systems in transport, utilities and the military, the field of topics that touch on research is very wide: it can include such obvious subjects as software modeling and reliability, or hardware design, to arcane elements such as stochastic models, graph theory, formal or exclusionary logic, parallel processing, remote data transmission, and more.

Chapter 7

Inherent Safety

Inherent safety is a concept particularly used in the chemical and process industries. An inherently safe process has a low level of danger even if things go wrong. It is used in contrast to safe systems where a high degree of hazard is controlled by protective systems. It should not be confused with intrinsic safety which is a particular technology for electrical systems in potentially flammable atmospheres. As perfect safety cannot be achieved, common practice is to talk about *inherently safer design*. "An inherently safer design is one that avoids hazards instead of controlling them, particularly by reducing the amount of hazardous material and the number of hazardous operations in the plant."

Origins

The concept of reducing rather than controlling hazards comes from Trevor Kletz in an article entitled "What You Don't Have, Can't Leak" on lessons from the Flixborough Disaster, and the name 'inherent safety' from a book which was an expanded version of the article. A greatly revised and retitled 1991 version gave the techniques which are generally quoted.

Principles

The terminology of inherent safety has developed since 1991, with some slightly different words but the same intentions as Kletz. The 4 main methods for achieving inherently safer design are:

- **Minimize**: Reducing the amount of hazardous material present at any one time. Example: The old batch process for the manufacture of nitro-glycerin, The reaction was carried out using 1 tonne of material, the reason is because the time scale was long 92 hours). however the chemical reaction is not slow, but the mixing process in the batch reactor was not good.

The problem was solved by the design of a small, continuous-flow, well-mixed reactor with residence time reduced from 2 hours to 2 minutes. The advantages of continuous process over a batch process is:

- Heat evolution is uniform therefore easier to control
- Batch processes change with time and affected by mixing quality.
- **Substitute**: Replacing one material with another of less hazard, e.g. cleaning with water and detergent rather than a flammable solvent
- **Moderate**: Reducing the strength of an effect, e.g. having a cold liquid instead of a gas at high pressure, or using material in a dilute rather than concentrated form
- **Simplify**: Designing out problems rather than adding additional equipment or features to deal with them. Only fitting options and using complex procedures if they are really necessary.

2 further principles are used by some:

- **Error Tolerance**: Equipment and processes can be designed to be capable of withstanding possible faults or deviations from design. A very simple example is making piping and joints capable of withstanding the maximum possible pressure if outlets are closed.
- **Limit Effects**: Designing and locating equipment so that the worst possible condition gives less danger, e.g. gravity will take a leak to a safe place, the use of bunds.

In terms of making plants more user-friendly Kletz also added the following:

- Avoiding Knock-on Effects;
- Making Incorrect Assembly Impossible;
- Making Status Clear;
- Ease of Control;
- Software and management procedures.

Official Status

Inherent safety has been recognised as a desirable principle by a number of national authorities, including the US Nuclear Regulatory Commission and the UK Health and Safety Executive (HSE). In assessing COMAH sites the HSE states “Major accident hazards should be avoided or reduced at source through the application of principles of inherent safety”. The European Commission in its Guidance Document on the Seveso II Directive states “Hazards should be possibly avoided or reduced at source through the application of inherently safe practices.” In the USA, Contra Costa County requires chemical plants and petroleum refineries to implement inherent safety reviews and make changes based on these reviews.

Quantification

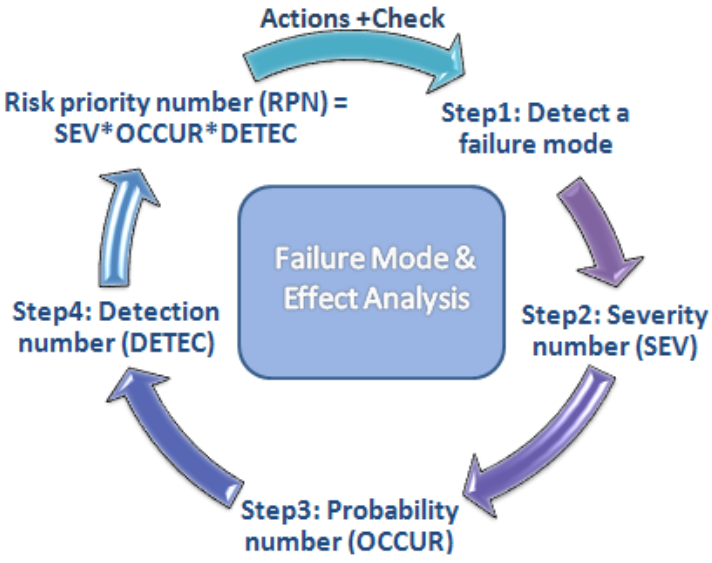
The Dow Fire and Explosion Index is essentially a measure of inherent danger and is the most widely used quantification of inherent safety. A more specific index of inherently safe design has been proposed by Heikkilä, and variations of this have been published. However all of these are much more complex than the Dow F & E Index.

Chapter 8

Failure Mode and Effects Analysis

A **failure modes and effects analysis (FMEA)**, is a procedure in product development and operations management for analysis of potential failure modes within a system for classification by the severity and likelihood of the failures. A successful FMEA activity helps a team to identify potential failure modes based on past experience with similar products or processes, enabling the team to design those failures out of the system with the minimum of effort and resource expenditure, thereby reducing development time and costs. It is widely used in manufacturing industries in various phases of the product life cycle and is now increasingly finding use in the service industry. *Failure modes* are any errors or defects in a process, design, or item, especially those that affect the customer, and can be potential or actual. *Effects analysis* refers to studying the consequences of those failures.

Basic terms



FMEA cycle.

Failure
: "The LOSS of an intended function of a device under stated conditions."

Failure mode
: "The manner by which a failure is observed; it generally describes the way the failure occurs."

Failure effect
: Immediate consequences of a failure on operation, function or functionality, or status of some item

Indenture levels
: An identifier for item complexity. Complexity increases as levels are closer to one.

Local effect
: The Failure effect as it applies to the item under analysis.

Next higher level effect
: The Failure effect as it applies at the next higher indenture level.

End effect
: The failure effect at the highest indenture level or total system.

Failure cause
: Defects in design, process, quality, or part application, which are the underlying cause of the failure or which initiate a process which leads to failure.

Severity
: "The consequences of a failure mode. Severity considers the worst potential consequence of a failure, determined by the degree of injury, property damage, or system damage that could ultimately occur."

History

Learning from each failure is both costly and time consuming, and FMEA is a more systematic method of studying failure. As such, it is considered better to first conduct some thought experiments.

FMEA was formally introduced in the late 1940s for military usage by the US Armed Forces. Later it was used for aerospace/rocket development to avoid errors in small sample sizes of costly rocket technology. An example of this is the Apollo Space program. It was also used as application for HACCP for the Apollo Space Program, and later the food industry in general. The primary push came during the 1960s, while developing the means to put a man on the moon and return him safely to earth. In the late 1970s the Ford Motor Company introduced FMEA to the automotive industry for safety and regulatory consideration after the Pinto affair. They applied the same approach to processes (PFMEA) to consider potential process induced failures prior to launching production.

Although initially developed by the military, FMEA methodology is now extensively used in a variety of industries including semiconductor processing, food service, plastics, software, and healthcare. It is integrated into the Automotive Industry Action Group's (AIAG) Advanced Product Quality Planning (APQP) process to provide risk mitigation,

in both product and process development phases. Each potential cause must be considered for its effect on the product or process and, based on the risk, actions are determined and risks revisited after actions are complete. Toyota has taken this one step further with its Design Review Based on Failure Mode (DRBFM) approach. The method is now supported by the American Society for Quality which provides detailed guides on applying the method.

Implementation

In FMEA, failures are prioritized according to how serious their consequences are, how frequently they occur and how easily they can be detected. An FMEA also documents current knowledge and actions about the risks of failures for use in continuous improvement. FMEA is used during the design stage with an aim to avoid future failures (sometimes called DFMEA in that case). Later it is used for process control, before and during ongoing operation of the process. Ideally, FMEA begins during the earliest conceptual stages of design and continues throughout the life of the product or service.

The outcome of an FMEA development is actions to prevent or reduce the severity or likelihood of failures, starting with the highest-priority ones. It may be used to evaluate risk management priorities for mitigating known threat vulnerabilities. FMEA helps select remedial actions that reduce cumulative impacts of life-cycle consequences (risks) from a systems failure (fault).

It is used in many formal quality systems such as QS-9000 or ISO/TS 16949.

Using FMEA when designing

FMEA can provide an analytical approach, when dealing with potential failure modes and their associated causes. When considering possible failures in a design – like safety, cost, performance, quality and reliability – an engineer can get a lot of information about how to alter the development/manufacturing process, in order to avoid these failures. FMEA provides an easy tool to determine which risk has the greatest concern, and therefore an action is needed to prevent a problem before it arises. The development of these specifications will ensure the product will meet the defined requirements and customer needs.

The pre-work

The process for conducting an FMEA is straightforward. It is developed in three main phases, in which appropriate actions need to be defined. But before starting with an FMEA, it is important to complete some pre-work to confirm that robustness and past history are included in the analysis.

A robustness analysis can be obtained from interface matrices, boundary diagrams, and parameter diagrams. Many failures are due to noise factors and shared interfaces with other parts and/or systems, because engineers tend to focus on what they control directly.

To start it is necessary to describe the system and its function. A good understanding simplifies further analysis. This way an engineer can see which uses of the system are desirable and which are not. It is important to consider both intentional and unintentional uses. Unintentional uses are a form of hostile environment.

Then, a block diagram of the system needs to be created. This diagram gives an overview of the major components or process steps and how they are related. These are called logical relations around which the FMEA can be developed. It is useful to create a coding system to identify the different system elements. The block diagram should always be included with the FMEA.

Before starting the actual FMEA, a worksheet needs to be created, which contains the important information about the system, such as the revision date or the names of the components. On this worksheet all the items or functions of the subject should be listed in a logical manner, based on the block diagram.

Example FMEA Worksheet

Function	Failure mode	Effects	S (severity rating)	Cause(s)	O (occurrence rating)	Current controls	D (detection rating)	CRIT (critical characteristic	RPN (risk priority number)	Recommended actions	Responsibility and target completion date	Action taken
Fill tub	High level sensor never trips	Liquid spills on customer floor	8	level sensor failed level sensor disconnected	2	Fill timeout based on time to fill to low level sensor	5	N	80	Perform cost analysis of adding additional sensor halfway between low and high level sensors	Jane Doe 10-Oct-2010	

Step 1: Occurrence

In this step it is necessary to look at the cause of a failure mode and how many times it occurs. This can be done by looking at similar products or processes and the failure modes that have been documented for them. A failure cause is looked upon as a design weakness. All the potential causes for a failure mode should be identified and documented. Again this should be in technical terms. Examples of causes are: erroneous algorithms, excessive voltage or improper operating conditions. A failure mode is given an *occurrence ranking (O)*, again 1–10. Actions need to be determined if the occurrence is high (meaning > 4 for non-safety failure modes and > 1 when the severity-number from step 1 is 9 or 10). This step is called the detailed development section of the FMEA process. Occurrence also can be defined as %. If a non-safety issue happened less than 1%, we can give 1 to it. It is based on your product and customer specification.

Rating	Meaning
1	No effect
2/3	Low (relatively few failures)
4/5/6	Moderate (occasional failures)
7/8	High (repeated failures)
9/10	Very high (failure is almost inevitable)

Step 2: Severity

Determine all failure modes based on the functional requirements and their effects. Examples of failure modes are: Electrical short-circuiting, corrosion or deformation. A failure mode in one component can lead to a failure mode in another component, therefore each failure mode should be listed in technical terms and for function. Hereafter the ultimate effect of each failure mode needs to be considered. A failure effect is defined as the result of a failure mode on the function of the system as perceived by the user. In this way it is convenient to write these effects down in terms of what the user might see or experience. Examples of failure effects are: degraded performance, noise or even injury to a user. Each effect is given a *severity number (S)* from 1 (no danger) to 10 (critical). These numbers help an engineer to prioritize the failure modes and their effects. If the severity of an effect has a number 9 or 10, actions are considered to change the design by eliminating the failure mode, if possible, or protecting the user from the effect. A severity rating of 9 or 10 is generally reserved for those effects which would cause injury to a user or otherwise result in litigation.

Rating	Meaning
1	No effect
2	Very minor (only noticed by discriminating customers)
3	Minor (affects very little of the system, noticed by average customer)
4/5/6	Moderate (most customers are annoyed)
7/8	High (causes a loss of primary function; customers are dissatisfied)
9/10	Very high and hazardous (product becomes inoperative; customers angered; the failure may result unsafe operation and possible injury)

Step 3: Detection

When appropriate actions are determined, it is necessary to test their efficiency. In addition, design verification is needed. The proper inspection methods need to be chosen. First, an engineer should look at the current controls of the system, that prevent failure modes from occurring or which detect the failure before it reaches the customer. Hereafter one should identify testing, analysis, monitoring and other techniques that can be or have been used on similar systems to detect failures. From these controls an engineer can learn how likely it is for a failure to be identified or detected. Each combination from the previous 2 steps receives a *detection number (D)*. This ranks the

ability of planned tests and inspections to remove defects or detect failure modes in time. The assigned detection number measures the risk that the failure will *escape detection*. A high detection number indicates that the chances are high that the failure will escape detection, or in other words, that the chances of detection are low.

Rating	Meaning
1	Almost certain
2	High
3	Moderate
4/5/6	Moderate - most customers are annoyed
7/8	Low
9/10	Very remote to absolute uncertainty

After these three basic steps, risk priority numbers (RPN) are calculated

Risk priority numbers

RPN play an important part in the choice of an action against failure modes. They are threshold values in the evaluation of these actions.

After ranking the severity, occurrence and detectability the RPN can be easily calculated by multiplying these three numbers: RPN = S × O × D

This has to be done for the entire process and/or design. Once this is done it is easy to determine the areas of greatest concern. The failure modes that have the highest RPN should be given the highest priority for corrective action. This means it is not always the failure modes with the highest severity numbers that should be treated first. There could be less severe failures, but which occur more often and are less detectable.

After these values are allocated, recommended actions with targets, responsibility and dates of implementation are noted. These actions can include specific inspection, testing or quality procedures, redesign (such as selection of new components), adding more redundancy and limiting environmental stresses or operating range. Once the actions have been implemented in the design/process, the new RPN should be checked, to confirm the improvements. These tests are often put in graphs, for easy visualization. Whenever a design or a process changes, an FMEA should be updated.

A few logical but important thoughts come in mind:

- Try to eliminate the failure mode (some failures are more preventable than others)
- Minimize the severity of the failure
- Reduce the occurrence of the failure mode
- Improve the detection

Timing of FMEA

The FMEA should be updated whenever:

- At the beginning of a cycle (new product/process)
- Changes are made to the operating conditions
- A change is made in the design
- New regulations are instituted
- Customer feedback indicates a problem

Uses of FMEA

- Development of system requirements that minimize the likelihood of failures.
- Development of methods to design and test systems to ensure that the failures have been eliminated.
- Evaluation of the requirements of the customer to ensure that those do not give rise to potential failures.
- Identification of certain design characteristics that contribute to failures, and minimize or eliminate those effects.
- Tracking and managing potential risks in the design. This helps avoid the same failures in future projects.
- Ensuring that any failure that could occur will not injure the customer or seriously impact a system.
- To produce world class quality products

Advantages

- Improve the quality, reliability and safety of a product/process
- Improve company image and competitiveness
- Increase user satisfaction
- Reduce system development timing and cost
- Collect information to reduce future failures, capture engineering knowledge
- Reduce the potential for warranty concerns
- Early identification and elimination of potential failure modes
- Emphasize problem prevention
- Minimize late changes and associated cost
- Catalyst for teamwork and idea exchange between functions
- Reduce the possibility of same kind of failure in future
- Reduce impact of profit margin company
- Reduce possible scrap in production

Limitations

Since FMEA is effectively dependent on the members of the committee which examines product failures, it is limited by their experience of previous failures. If a failure mode cannot be identified, then external help is needed from consultants who are aware of the

many different types of product failure. FMEA is thus part of a larger system of quality control, where documentation is vital to implementation. General texts and detailed publications are available in forensic engineering and failure analysis. It is a general requirement of many specific national and international standards that FMEA is used in evaluating product integrity. If used as a top-down tool, FMEA may only identify major failure modes in a system. Fault tree analysis (FTA) is better suited for "top-down" analysis. When used as a "bottom-up" tool FMEA can augment or complement FTA and identify many more causes and failure modes resulting in top-level symptoms. It is not able to discover complex failure modes involving multiple failures within a subsystem, or to report expected failure intervals of particular failure modes up to the upper level subsystem or system.

Additionally, the multiplication of the severity, occurrence and detection rankings may result in rank reversals, where a less serious failure mode receives a higher RPN than a more serious failure mode. The reason for this is that the rankings are ordinal scale numbers, and multiplication is not defined for ordinal numbers. The ordinal rankings only say that one ranking is better or worse than another, but not by how much. For instance, a ranking of "2" may not be twice as bad as a ranking of "1," or an "8" may not be twice as bad as a "4," but multiplication treats them as though they are.

Software

Most FMEAs are created as a spreadsheet. Specialized FMEA software packages exist that offer some advantages over spreadsheets.

Types of FMEA

- Process: analysis of manufacturing and assembly processes
- Design: analysis of products prior to production
- Concept: analysis of systems or subsystems in the early design concept stages
- Equipment: analysis of machinery and equipment design before purchase
- Service: analysis of service industry processes before they are released to impact the customer
- System: analysis of the global system functions
- Software: analysis of the software functions

Chapter 9

Failure Mode, Effects, and Criticality Analysis

Failure mode, effects, and criticality analysis (FMECA) is an extension of failure mode and effects analysis (FMEA). FMEA is a bottom-up, inductive analytical method which may be performed at either the functional or piece-part level. FMECA extends FMEA by including a *criticality analysis*, which is used to chart the probability of failure modes against the severity of their consequences. The result highlights failure modes with relatively high probability and severity of consequences, allowing remedial effort to be directed where it will produce the greatest value. FMECA tends to be preferred over FMEA in space and North Atlantic Treaty Organization (NATO) military applications, while various forms of FMEA predominate in other industries.

History

FMECA was originally developed in the 1940's by the U.S military, which published MIL–P–1629 in 1949. By the early 1960s, contractors for the U.S. National Aeronautics and Space Administration (NASA) were using variations of FMECA under a variety of names. In 1966 NASA released its FMECA procedure for use on the Apollo program. FMECA was subsequently used on other NASA programs including Viking, Voyager, Magellan, and Galileo. Possibly because MIL–P–1629 was replaced by MIL–STD–1629 (SHIPS) in 1974, development of FMECA is sometimes incorrectly attributed to NASA. At the same time as the space program developments, use of FMEA and FMECA was already spreading to civil aviation. In 1967 the Society for Automotive Engineers released the first civil publication to address FMECA. The civil aviation industry now tends to use a combination of FMEA and Fault Tree Analysis in accordance with SAE ARP4761 instead of FMECA, though some helicopter manufacturers continue to use FMECA for civil rotorcraft.

Ford Motor Company began using FMEA in the 1970s after problems experienced with its Pinto model, and by the 1980's FMEA was gaining broad use in the automotive industry. In Europe, the International Electrotechnical Commission published IEC 812 (now IEC 60812) in 1985, addressing both FMEA and FMECA for general use. The British Standards Institute published BS 5760–5 in 1991 for the same purpose.

In 1980, MIL–STD–1629A replaced both MIL–STD–1629 and the 1977 aeronautical FMECA standard MIL–STD–2070. MIL–STD–1629A was canceled without replacement in 1998, but nonetheless remains in wide use for military and space applications today.

Methodology

Slight differences are found between the various FMECA standards. By RAC CRTA–FMECA, the FMECA analysis procedure typically consists of the following logical steps:

- Define the system
- Define ground rules and assumptions in order to help drive the design
- Construct system block diagrams
- Identify failure modes (piece part level or functional)
- Analyze failure effects/causes
- Feed results back into design process
- Classify the failure effects by severity
- Perform criticality calculations
- Rank failure mode criticality
- Determine critical items
- Feed results back into design process
- Identify the means of failure detection, isolation and compensation
- Perform maintainability analysis
- Document the analysis, summarize uncorrectable design areas, identify special controls necessary to reduce failure risk
- Make recommendations
- Follow up on corrective action implementation/effectiveness

FMECA may be performed at the functional or piece part level. Functional FMECA considers the effects of failure at the functional block level, such as a power supply or an amplifier. Piece part FMECA considers the effects of individual component failures, such as resistors, transistors, microcircuits, or valves. A piece part FMECA requires far more effort, but is sometimes preferred because it relies more on quantitative data and less an engineering judgment than a functional FMECA.

The criticality analysis may be quantitative or qualitative, depending on the availability of supporting part failure data.

System definition

In this step, the major system to be analyzed is defined and partitioned into an indentured hierarchy such as systems, subsystems or equipment, units or subassemblies, and piece parts. Functional descriptions are created for the systems and allocated to the subsystems, covering all operational modes and mission phases.

Ground rules and assumptions

Before detailed analysis takes place, ground rules and assumptions are usually defined and agreed to. This might include, for example:

- Standardized mission profile with specific fixed duration mission phases
- Sources for failure rate and failure mode data
- Fault detection coverage that system built-in test will realize
- Whether the analysis will be functional or piece part
- Criteria to be considered (mission abort, safety, maintenance, etc.)
- System for uniquely identifying parts or functions
- Severity category definitions

Block diagrams

Next, the systems and subsystems are depicted in functional block diagrams. Reliability block diagrams or fault trees are usually constructed at the same time. These diagrams are used to trace information flow at different levels of system hierarchy, identify critical paths and interfaces, and identify the higher level effects of lower level failures.

Failure mode identification

For each piece part or each function covered by the analysis, a complete list of failure modes is developed. For functional FMECA, typical failure modes include:

- Untimely operation
- Failure to operate when required
- Loss of output
- Intermittent output
- Erroneous output (given the current condition)
- Invalid output (for any condition)

For piece part FMECA, failure mode data may be obtained from databases such as RAC FMD–91 or RAC FMD–97. These databases provide not only the failure modes, but also the failure mode ratios. For example:

Device Failure Modes and Failure Mode Ratios (FMD–91)

Device Type	Failure Mode	Ratio (α)
Relay	Fails to trip	.55
	Spurious trip	.26
	Short	.19
Resistor, Composistion	Parameter change	.66
	Open	.31
	Short	.93

Each function or piece part is then listed in matrix form with one row for each failure mode. Because FMECA usually involves very large data sets, a unique identifier must be assigned to each item (function or piece part), and to each failure mode of each item.

Failure effects analysis

Failure effects are determined and entered for each row of the FMECA matrix, considering the criteria identified in the ground rules. Effects are separately described for the local, next higher, and end (system) levels. System level effects may include:

- System failure
- Degraded operation
- System status failure
- No immediate effect

The failure effect categories used at various hierarchical levels are tailored by the analyist using engineering judgment.

Severity classification

Severity classification is assigned for each failure mode of each unique item and entered on the FMECA matrix, based upon system level consequences. A small set of classifications, usually having 3 to 10 severity levels, is used. For example, When prepared using MIL–STD–1629A, failure or mishap severity classification normally follows MIL–STD–882.

Mishap Severity Categories (MIL–STD–882)

Category	Description	Criteria
I	Catastrophic	Could result in death, permanent total disability, loss exceeding $1M, or irreversible severe environmental damage that violates law or regulation.
II	Critical	Could result in permanent partial disability, injuries or occupational illness that may result in hospitalization of at least three personnel, loss exceeding $200K but less than $1M, or reversible environmental damage causing a violation of law or regulation.
III	Marginal	Could result in injury or occupational illness resulting in one or more lost work days(s), lossexceeding $10K but less than $200K, or mitigatible environmental damage without violation of law or regulation where restoration activities can be accomplished.
IV	Negligible	Could result in injury or illness not resulting in a lost work day, loss exceeding $2K but less than $10K, or minimal environmental damage not violating law or regulation.

Current FMECA severity categories for U.S. Federal Aviation Administration (FAA), NASA and European Space Agency space applications are derived from MIL–STD–882.

Failure detection methods

For each component and failure mode, the ability of the system to detect and report the failure in question is analyzed. One of the following will be entered on each row of the FMECA matrix:

- *Normal*: the system correctly indicates a safe condition to the crew
- *Abnormal*: the system correctly indicates a malfunction requiring crew action
- *Incorrect*: the system erroneously indicates a safe condition in the event of malfunction, or alerts the crew to a malfunction that does not exist (false alarm)

Criticality ranking

Failure mode criticality assessment may be qualitative or quantitative. For qualitative assessment, a mishap probability code or number is assigned and entered on the matrix. For example, MIL–STD–882 uses five probability levels:

Failure Probability Levels (MIL–STD–882)

Description	Level	Individual Item	Fleet
Frequent	A	Likely to occur in the life of the item	Continuously experienced
Probable	B	Will occur several times in the life of an item	Will occur frequently
Occasional	C	Likely to occur some time in the life of an item	Will occur several times
Remote	D	Unlikely but possible to occur in the life of an item	Unlikely, but can reasonably be expected to occur
Improbable	E	So unlikely, it can be assumed occurrence may not be experienced	Unlikely to occur, but possible

The failure mode may then be charted on a criticality matrix using severity code as one axis and probability level code as the other. For quantitative assessment, *modal criticality number* C_m is calculated for each failure mode of each item, and *item crticality number* C_r is calculated for each item. The criticality numbers are computed using the following values:

- Basic failure rate λ_p
- Failure mode ratio α
- Conditional probability β

- Mission phase duration t

The criticality numbers are computed as $C_m = \lambda_p \alpha \beta t$ and $C_r = \sum_{n=1}^{N} (C_m)_n$. The basic failure rate λ_p is usually fed into the FMECA from a failure rate prediction based on MIL–HDBK–217, PRISM, RIAC 217Plus, or a similar model. The failure mode ratio may be taken from a database source such as RAC FMD–97. For functional level FMECA, engineering judgment may be required to assign failure mode ratio. The conditional probability number β represents the conditional probability that the failure effect will result in the identified severity classification, given that the failure mode occurs. It represents the analyst's best judgment as to the likelihood that the loss will occur. For graphical analysis, a criticality matrix may be charted using either C_m or C_r on one axis and severity code on the other.

Critical item/failure mode list

Once the criticality assessment is completed for each failure mode of each item, the FMECA matrix may be sorted by severity and qualitative probability level or quantiative criticality number. This enables the analysis to identify critical items and critical failure modes for which design mitigation is desired.

Recommendations

After performing FMECA, recommendations are made to design to reduce the consequences of critical failures. This may include selecting components with higher reliability, reducing the stress level at which a critical item operates, or adding redundancy or monitoring to the system.

Maintainability analysis

FMECA usually feeds into both Maintainability Analysis and Logistics Support Analysis, which both require data from the FMECA.

FMECA report

A FMECA report consists of system description, ground rules and assumptions, conclusions and recommendations, corrective actions to be tracked, and the attached FMECA matrix which may be in spreadsheet, worksheet, or database form.

Risk priority calculation

RAC CRTA–FMECA and MIL–HDBK–338 both identify Risk Priorty Number (RPN) calculation is an alternate method to criticality analysis. The *RPN* is a result of a multiplication of detectability (*D*) x severity (*S*) x occurrence (*O*). Each on a scale from 1 to 10. The highest *RPN* is 10x10x10 = 1000. This means that this failure is not detectable

by inspection, very severe and the occurrence is almost sure. If the occurrence is very sparse, this would be 1 and the *RPN* would decrease to 100. So, criticality analysis enables to focus on the highest risks.

Advantages and disadvantages

Strengths of FMECA include its comprehensiveness, the systematic establishment of relationships between failure causes and effects, and its ability to point out individual failure modes for corrective action in design. Weaknesses include the extensive labor required, the large number of trivial cases considered, and inability to deal with multiple-failure scenarios or unplanned cross-system effects such as sneak circuits.

According to an FAA research report for commercial space transportation,

Failure Modes, effects, and Criticality Analysis is an excellent hazard analysis and risk assessment tool, but it suffers from other limitations. This alternative does not consider combined failures or typically include software and human interaction considerations. It also usually provides an optimistic estimate of reliability. Therefore, FMECA should be used in conjunction with other analytical tools when developing reliability estimates.

Chapter 10

Effective Safety Training

The Occupational Safety and Health Administration (OSHA) has written voluminous workplace safety standards and regulations that affect employers and employees in the United States. It is the employer's legal responsibility to educate employees on all workplace safety standards and the hazards that their employees may face while on the job.

Introduction

Employers must have an overall safety program including relative site specific safety information where applicable. The safety training program should cover topics such as:

- Accident Prevention and Safety Promotion
- Safety Compliance
- Accident and Emergency Response
- Personal Protective Equipment
- Safety Practices
- Equipment and Machinery
- Chemical and Hazardous Materials Safety
- Workplace Hazards
- Employee Involvement and

Employers must document all training. Creating a training matrix will help keep track of who has been trained, when they were trained, the training topic, and when it is time for refresher training. Employees must also sign an official sign-in sheet provided by the employer that can serve as proof that employees received proper training. The sign in sheet must have a broad description of what is being covered in the training. Tests or quizzes on the presented material can help gauge employee understanding of the material and highlight topics that need to be reviewed.

The non-English speaking population is consistently growing in many industries and it is important that employers provide bilingual training for those workers, as OSHA requires that **all employees** be properly trained.

Most employees display attitudes of disinterest and dread at the thought of attending a safety training, which can leave the trainer feeling frustrated and unappreciated. It is the trainer's duty to make safety training fun and educational, which will help the trainees to retain the information, enjoy the course, and apply the learning to their work and lives.

Benefits of a training program

An effective training program can reduce the number of injuries and deaths, property damage, legal liability, illnesses, workers' compensation claims, and missed time from work. A safety training program can also help a trainer keep the required OSHA-mandated safety training courses organized and up-to-date.

Safety training classes help establish a safety culture in which employees themselves help promote proper safety procedures while on the job. It is important that new employees be properly trained and embrace the importance of workplace safety as it is easy for seasoned workers to negatively influence the new hires. That negative influence however, can be purged with the establishment of new, hands-on, innovative effective safety training which will ultimately lead to an effective safety culture. A 1998 NIOSH study concluded that the role of training in developing and maintaining effective hazard control activities is a proven and successful method of intervention.

OSHA's voluntary training guidelines

OSHA issued voluntary training guidelines in 1992. These guidelines serve as a model for trainers to use in developing, organizing, evaluating, and editing their safety training programs. It is important for trainers to tailor the OSHA guidelines to their specific work site so that the training is relevant to the specific working conditions and not just a long generalized informational session.

Many standards promulgated by OSHA explicitly require the employer to train employees in the safety and health aspects of their jobs. Other OSHA standards make it the employer’s responsibility to limit certain job assignments to employees who are “certified,” “competent,” or “qualified”—meaning that they have had special previous training, in or out of the workplace. The term “designated” personnel means selected or assigned by the employer or the employer’s representative as being qualified to perform specific duties. These requirements reflect OSHA’s belief that training is an essential part of every employer’s safety and health program for protecting workers from injuries and illnesses.

OSHA’s training guidelines follow a model that consists of:

- A. Determining if Training is Needed
- B. Identifying Training Needs
- C. Identifying Goals and Objectives
- D. Developing Learning Activities
- E. Conducting the Training

- F. Evaluating Program Effectiveness
- G. Improving the Program
- H. Training must align with job tasks.

A. Determining if Training is Needed

You first have to determine if a situation can be solved using training. Training, or retraining as the case may be, could be required by an OSHA standard. Training is an effective solution to problems such as employee lack of understanding, unfamiliarity with equipment, incorrect execution of a task, lack of attention, or lack of motivation. Sometimes, however, the situation cannot be mitigated through the use of training and other methods, such as the establishment of engineering controls, may be needed to ensure worker safety.

B. Identifying Training Needs

A Job Safety Analysis and/or a job hazard analysis should be conducted with every employee so that it is understood what is needed to do the job safely and what hazards are associated with the job. A safety trainer may observe the worker in his/her environment to adequately assess the worker's training needs. Certain employees may need extra training due to the hazards associated with their particular job. These employees should be trained not only on how to perform their job safely but also on how to operate within a hazardous environment.

C. Identifying Goals and Objectives

It is important for the Trainer to identify necessary training material. It is equally important that the trainer identify training material that is not needed to avoid unnecessary training and frustration from their trainees.

At the beginning of every safety training session the trainer should clearly iterate the objectives of the class. The objectives should be delivered using action oriented words like: the employee... "will be able to demonstrate" or "will know when to"... which will help the audience understand what he/she should know by the end of the class or what to information to assimilate during the class. Clearly established objectives also help focus the evaluation process on those skill sets and knowledge requirements necessaary to perform the job safely.

- **D. Developing Learning Activities**

Training should be hands-on and simulate the job as closely as possible. Trainers can use instructional aids such as charts, manuals, PowerPoint presentations, and films. Trainers can also include role-playing, live demonstrations, and round-table group discussions to stimulate employee participation. Games like "what's wrong with this picture" (it is usually good to use pictures of situations found at their specific location)" or "safety jeopardy" can be useful ways to make the training fun yet educational.

- **E. Conducting the Training**

Trainers should provide employees with an overview of the material to be learned and relate the training to the employees' experiences. Employers should also reinforce what the employees have learned by summarizing the program's objectives and key points of training. At the beginning of the training program, the trainer should show the employees why the material is important and relevant to their jobs. Employees are more likely to pay attention and apply what they've learned if they know the benefits of the training.

- **F. Evaluating Program Effectiveness**

Evaluation will help employers or supervisors determine the amount of learning achieved and whether an employee’s performance has improved on the job. Among the methods of evaluating training are:

- (1) Student opinion. Questionnaires or informal discussions with employees can help employers determine the relevance and appropriateness of the training program
- (2) Supervisors’ observations. Supervisors are in good positions to observe an employee’s performance both before and after the training and note improvements or changes
- (3) Workplace improvements. The ultimate success of a training program may be changes throughout the workplace that result in reduced injury or accident rates
- (4) Formal assessments. Practical and written exams also assist in evaluating understanding of training material. For example, for a lift-truck operator, a written and a practical exam would identify areas of training that may need to be revisited. Furthermore administering a pre-test and post-test will establish a knowledge base line or reference point to measure training effectiveness.

- **G. Improving the Program**

As evaluations are reviewed, it may be evident the training was not adequate and that the employees did not reach the expected level of knowledge and skill. As the program is evaluated, the trainer should ask:

- (1) If a job analysis was conducted, was it accurate?
- (2) Was any critical feature of the job overlooked?
- (3) Were the important gaps in knowledge and skill included?
- (4) Was material already known by the employees intentionally omitted?
- (5) Were the instructional objectives presented clearly and concretely?
- (6) Did the objectives state the level of acceptable performance that was expected of employees?
- (7) Did the learning activity simulate the actual job?
- (8) Was the learning activity appropriate for the kinds of knowledge and skills required on the job?
- (9) When the training was presented, was the organization of the material and its meaning made clear?
- (10) Were the employees motivated to learn?
- (11) Were the employees allowed to participate actively in the training process?
- (12) Was the employer's evaluation of the program thorough?

Computer and video training

Computers and videos can be a great addition to a company's safety training program. As stand alone resources, they may not be adequate in meeting OSHA's training requirements as they are not site specific. Computer-based training can help meet the following training challenges

- Training employees in remote sites
- Employees who become bored with the same safety training
- Safety managers lack of time and resources to effectively train employees
- Providing a means of documenting and tracking student progress
- Lowering trainer fees or travel costs
- A self-paced, relaxed learning environment

OSHA Medical Safety

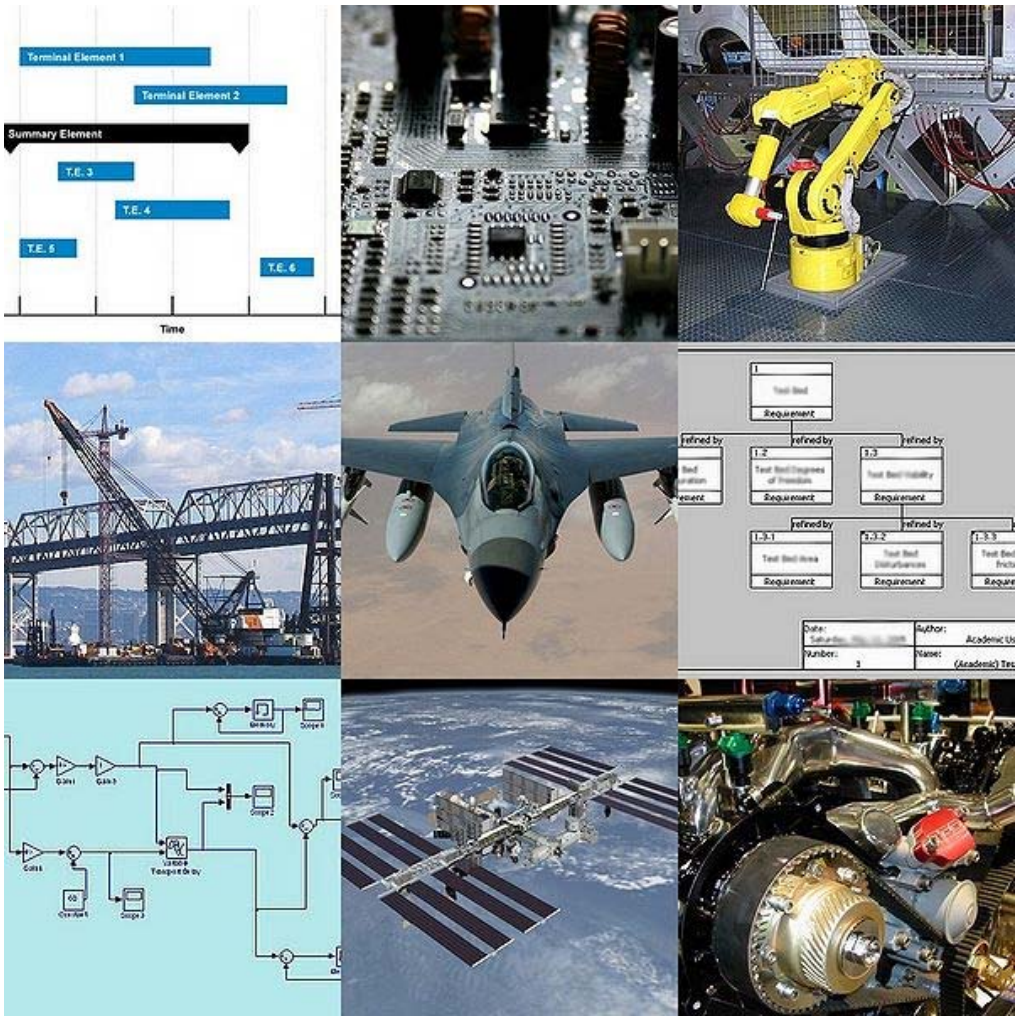
There is no more important places to regard the positive impact the OSHA regulations than in the healthcare and clinical settings. OSHA has been revolutionary in the medical

field due to its ability to prevent the spread of diseases. Every clinical facility on US land, civilian or military is governed by OSHA's directives. To remain in accordance to the Federal regulations enacted by OSHA healthcare administrators must maintain an OSHA safety program and train their employees on an annual basis. Some of the topics that employees must be trained on include:

- Bloodborne Pathogen Standard
- Chemical Hazard Communications
- Tuberculosis Exposure Control
- Mercury Exposure
- Ionizing Radiation Exposure
- Fire Escape Plan
- Emergency Action Plan
- Electricity Safety
- Fire Safety Standard

Chapter 11

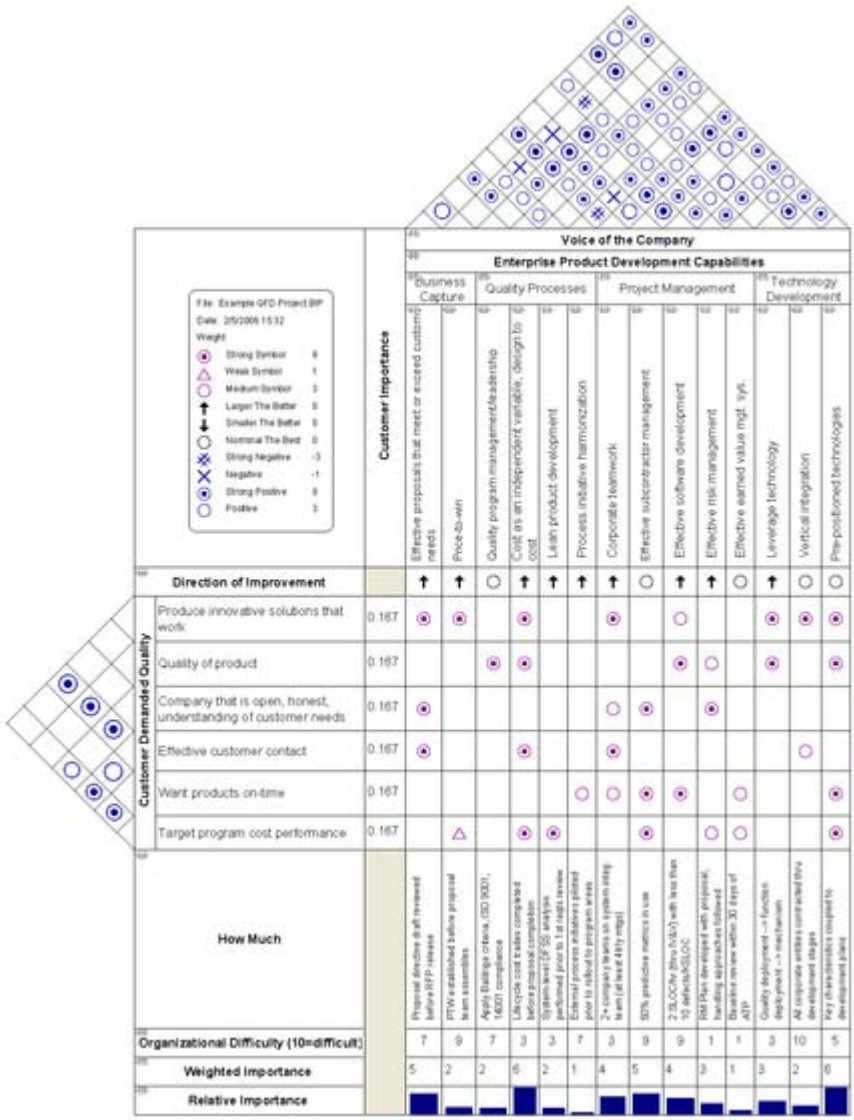
Systems Engineering



Systems engineering techniques are used in complex projects: spacecraft design, computer chip design, robotics, software integration, and bridge building. Systems engineering uses a host of tools that include modeling and simulation, requirements analysis and scheduling to manage complexity.

Systems engineering is an interdisciplinary field of engineering that focuses on how complex engineering projects should be designed and managed over the life cycle of the project. Issues such as logistics, the coordination of different teams, and automatic control of machinery become more difficult when dealing with large, complex projects. Systems engineering deals with work-processes and tools to handle such projects, and it overlaps with both technical and human-centered disciplines such as control engineering, industrial engineering, organizational studies, and project management.

History



QFD House of Quality for Enterprise Product Development Processes

The term *systems engineering* can be traced back to Bell Telephone Laboratories in the 1940s. The need to identify and manipulate the properties of a system as a whole, which

in complex engineering projects may greatly differ from the sum of the parts' properties, motivated the Department of Defense, NASA, and other industries to apply the discipline.

When it was no longer possible to rely on design evolution to improve upon a system and the existing tools were not sufficient to meet growing demands, new methods began to be developed that addressed the complexity directly. The evolution of systems engineering, which continues to this day, comprises the development and identification of new methods and modeling techniques. These methods aid in better comprehension of engineering systems as they grow more complex. Popular tools that are often used in the systems engineering context were developed during these times, including USL, UML, QFD, and IDEF0.

In 1990, a professional society for systems engineering, the *National Council on Systems Engineering* (NCOSE), was founded by representatives from a number of U.S. corporations and organizations. NCOSE was created to address the need for improvements in systems engineering practices and education. As a result of growing involvement from systems engineers outside of the U.S., the name of the organization was changed to the International Council on Systems Engineering (INCOSE) in 1995. Schools in several countries offer graduate programs in systems engineering, and continuing education options are also available for practicing engineers.

Concept

Systems engineering signifies both an approach and, more recently, as a discipline in engineering. The aim of education in systems engineering is to simply formalize the approach and in doing so, identify new methods and research opportunities similar to the way it occurs in other fields of engineering. As an approach, systems engineering is holistic and interdisciplinary in flavour.

Origins and traditional scope

The traditional scope of engineering embraces the design, development, production and operation of physical systems, and systems engineering, as originally conceived, falls within this scope. "Systems engineering", in this sense of the term, refers to the distinctive set of concepts, methodologies, organizational structures (and so on) that have been developed to meet the challenges of engineering functional physical systems of unprecedented complexity. The Apollo program is a leading example of a systems engineering project.

The use of the term " system engineer " has evolved over time to embrace a wider, more holistic concept of "systems" and of engineering processes. This evolution of the definition has been a subject of ongoing controversy , and the term continues to be applied to both the narrower and broader scope.

Holistic view

Systems engineering focuses on analyzing and eliciting customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem, the system lifecycle. Oliver *et al.* claim that the systems engineering process can be decomposed into

- a *Systems Engineering Technical Process*, and
- a *Systems Engineering Management Process*.

Within Oliver's model, the goal of the Management Process is to organize the technical effort in the lifecycle, while the Technical Process includes *assessing available information*, *defining effectiveness measures*, to *create a behavior model*, *create a structure model*, *perform trade-off analysis*, and *create sequential build & test plan*.

Depending on their application, although there are several models that are used in the industry, all of them aim to identify the relation between the various stages mentioned above and incorporate feedback. Examples of such models include the Waterfall model and the VEE model.

Interdisciplinary field

System development often requires contribution from diverse technical disciplines. By providing a systems (holistic) view of the development effort, systems engineering helps mold all the technical contributors into a unified team effort, forming a structured development process that proceeds from concept to production to operation and, in some cases, to termination and disposal.

This perspective is often replicated in educational programs in that systems engineering courses are taught by faculty from other engineering departments which, in effect, helps create an interdisciplinary environment.

Managing complexity

The need for systems engineering arose with the increase in complexity of systems and projects, in turn exponentially increasing the possibility of component friction, and therefore the reliability of the design. When speaking in this context, complexity incorporates not only engineering systems, but also the logical human organization of data. At the same time, a system can become more complex due to an increase in size as well as with an increase in the amount of data, variables, or the number of fields that are involved in the design. The International Space Station is an example of such a system.

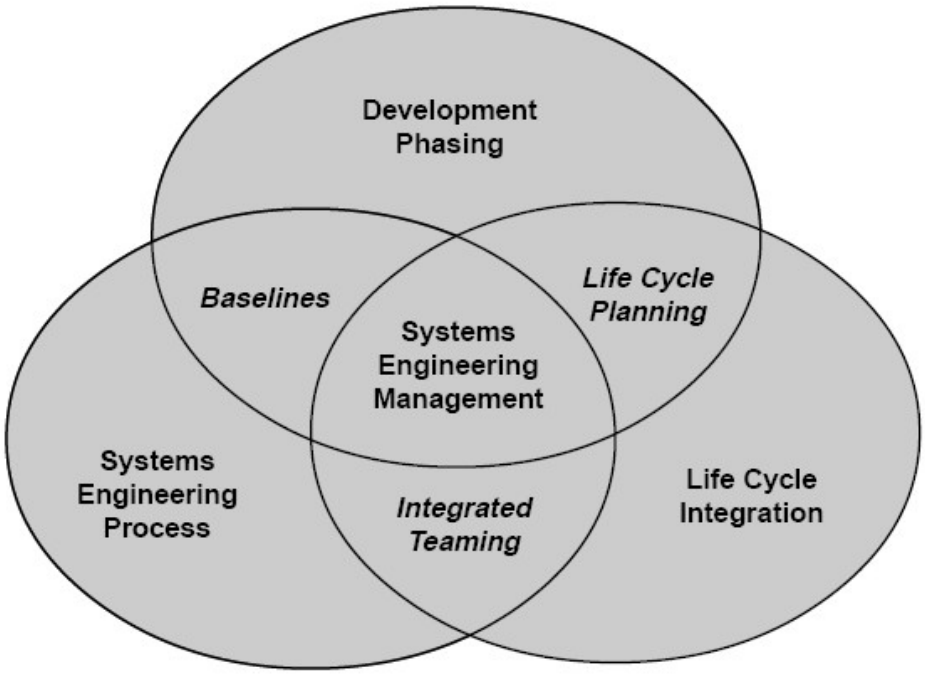
The development of smarter control algorithms, microprocessor design, and analysis of environmental systems also come within the purview of systems engineering. Systems

engineering encourages the use of tools and methods to better comprehend and manage complexity in systems. Some examples of these tools can be seen here:

- *System model, Modeling, and Simulation*,
- System architecture,
- *Optimization*,
- *System dynamics*,
- *Systems analysis*,
- *Statistical analysis*,
- *Reliability analysis*, and
- *Decision making*

Taking an interdisciplinary approach to engineering systems is inherently complex since the behavior of and interaction among system components is not always immediately well defined or understood. Defining and characterizing such systems and subsystems and the interactions among them is one of the goals of systems engineering. In doing so, the gap that exists between informal requirements from users, operators, marketing organizations, and technical specifications is successfully bridged.

Scope



The scope of systems engineering activities

One way to understand the motivation behind systems engineering is to see it as a method, or practice, to identify and improve common rules that exist within a wide variety of systems. Keeping this in mind, the principles of systems engineering — holism, emergent behavior, boundary, et al. — can be applied to any system, complex or otherwise, provided systems thinking is employed at all levels. Besides defense and aerospace, many information and technology based companies, software development firms, and industries in the field of electronics & communications require systems engineers as part of their team.

An analysis by the INCOSE Systems Engineering center of excellence (SECOE) indicates that optimal effort spent on systems engineering is about 15-20% of the total project effort. At the same time, studies have shown that systems engineering essentially leads to reduction in costs among other benefits. However, no quantitative survey at a larger scale encompassing a wide variety of industries has been conducted until recently. Such studies are underway to determine the effectiveness and quantify the benefits of systems engineering.

Systems engineering encourages the use of modeling and simulation to validate assumptions or theories on systems and the interactions within them.

Use of methods that allow early detection of possible failures, in safety engineering, are integrated into the design process. At the same time, decisions made at the beginning of a project whose consequences are not clearly understood can have enormous implications later in the life of a system, and it is the task of the modern systems engineer to explore these issues and make critical decisions. There is no method which guarantees that decisions made today will still be valid when a system goes into service years or decades after it is first conceived but there are techniques to support the process of systems engineering. Examples include the use of soft systems methodology, Jay Wright Forrester's System dynamics method and the Unified Modeling Language (UML), each of which are currently being explored, evaluated and developed to support the engineering decision making process.

Education

Education in systems engineering is often seen as an extension to the regular engineering courses, reflecting the industry attitude that engineering students need a foundational background in one of the traditional engineering disciplines (e.g. mechanical engineering, industrial engineering, computer engineering, electrical engineering) plus practical, real-world experience in order to be effective as systems engineers. Undergraduate university programs in systems engineering are rare.

INCOSE maintains a continuously updated Directory of Systems Engineering Academic Programs worldwide. As of 2006, there are about 75 institutions in United States that offer 130 undergraduate and graduate programs in systems engineering. Education in systems engineering can be taken as *SE-centric* or *Domain-centric*.

- *SE-centric* programs treat systems engineering as a separate discipline and all the courses are taught focusing on systems engineering practice and techniques.
- *Domain-centric* programs offer systems engineering as an option that can be exercised with another major field in engineering.

Both these patterns cater to educate the systems engineer who is able to oversee interdisciplinary projects with the depth required of a core-engineer.

Systems engineering topics

Systems engineering tools are strategies, procedures, and techniques that aid in performing systems engineering on a project or product. The purpose of these tools vary from database management, graphical browsing, simulation, and reasoning, to document production, neutral import/export and more.

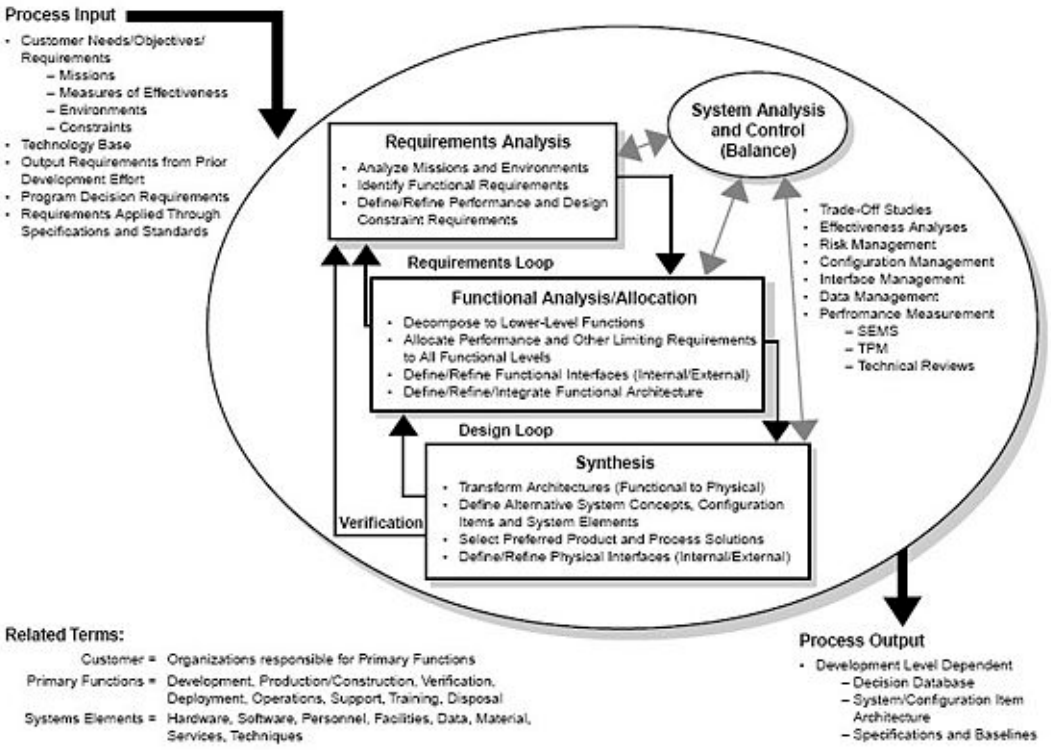
System

There are many definitions of what a system is in the field of systems engineering. Below are a few authoritative definitions:

- ANSI/EIA-632-1999: "An aggregation of end products and enabling products to achieve a given purpose."
- IEEE Std 1220-1998: "A set or arrangement of elements and processes that are related and whose behavior satisfies customer/operational needs and provides for life cycle sustainment of the products."
- ISO/IEC 15288:2008: "A combination of interacting elements organized to achieve one or more stated purposes."
- NASA Systems Engineering Handbook: "(1) The combination of elements that function together to produce the capability to meet a need. The elements include all hardware, software, equipment, facilities, personnel, processes, and procedures needed for this purpose. (2) The end product (which performs operational functions) and enabling products (which provide life-cycle support services to the operational end products) that make up a system."
- INCOSE Systems Engineering Handbook: "homogeneous entity that exhibits predefined behavior in the real world and is composed of heterogeneous parts that do not individually exhibit that behavior and an integrated configuration of components and/or subsystems."
- INCOSE: "A system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce systems-level results. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected."

The systems engineering process

Depending on their application, tools are used for various stages of the systems engineering process:



Using models

Models play important and diverse roles in systems engineering. A model can be defined in several ways, including:

- An abstraction of reality designed to answer specific questions about the real world
- An imitation, analogue, or representation of a real world process or structure; or
- A conceptual, mathematical, or physical tool to assist a decision maker.

Together, these definitions are broad enough to encompass physical engineering models used in the verification of a system design, as well as schematic models like a functional flow block diagram and mathematical (i.e., quantitative) models used in the trade study process.

The main reason for using mathematical models and diagrams in trade studies is to provide estimates of system effectiveness, performance or technical attributes, and cost from a set of known or estimable quantities. Typically, a collection of separate models is needed to provide all of these outcome variables. The heart of any mathematical model is

a set of meaningful quantitative relationships among its inputs and outputs. These relationships can be as simple as adding up constituent quantities to obtain a total, or as complex as a set of differential equations describing the trajectory of a spacecraft in a gravitational field. Ideally, the relationships express causality, not just correlation.

Tools for graphic representations

Initially, when the primary purpose of a systems engineer is to comprehend a complex problem, graphic representations of a system are used to communicate a system's functional and data requirements. Common graphical representations include:

- Functional Flow Block Diagram (FFBD)
- VisSim
- Data Flow Diagram (DFD)
- N2 (N-Squared) Chart
- IDEF0 Diagram
- UML Use case diagram
- UML Sequence diagram
- USL Function Maps and Type Maps.
- Enterprise Architecture frameworks, like TOGAF, MODAF, Zachman Frameworks etc.

A graphical representation relates the various subsystems or parts of a system through functions, data, or interfaces. Any or each of the above methods are used in an industry based on its requirements. For instance, the N2 chart may be used where interfaces between systems is important. Part of the design phase is to create structural and behavioral models of the system.

Once the requirements are understood, it is now the responsibility of a systems engineer to refine them, and to determine, along with other engineers, the best technology for a job. At this point starting with a trade study, systems engineering encourages the use of weighted choices to determine the best option. A decision matrix, or Pugh method, is one way (QFD is another) to make this choice while considering all criteria that are important. The trade study in turn informs the design which again affects the graphic representations of the system (without changing the requirements). In an SE process, this stage represents the iterative step that is carried out until a feasible solution is found. A decision matrix is often populated using techniques such as statistical analysis, reliability analysis, system dynamics (feedback control), and optimization methods.

At times a systems engineer must assess the existence of feasible solutions, and rarely will customer inputs arrive at only one. Some customer requirements will produce no feasible solution. Constraints must be traded to find one or more feasible solutions. The customers' wants become the most valuable input to such a trade and cannot be assumed. Those wants/desires may only be discovered by the customer once the customer finds that he has overconstrained the problem. Most commonly, many feasible solutions can be found, and a sufficient set of constraints must be defined to produce an optimal solution.

This situation is at times advantageous because one can present an opportunity to improve the design towards one or many ends, such as cost or schedule. Various modeling methods can be used to solve the problem including constraints and a cost function.

Systems Modeling Language (SysML), a modeling language used for systems engineering applications, supports the specification, analysis, design, verification and validation of a broad range of complex systems.

Universal Systems Language (USL) is a systems oriented object modeling language with executable (computer independent) semantics for defining complex systems, including software.

Related Fields and Sub-fields

Many related fields may be considered tightly coupled to systems engineering. These areas have contributed to the development of systems engineering as a distinct entity.

Cognitive systems engineering

Cognitive systems engineering (CSE) is a specific approach to the description and analysis of human-machine systems or sociotechnical systems. The three main themes of CSE are how humans cope with complexity, how work is accomplished by the use of artefacts, and how human-machine systems and socio-technical systems can be described as joint cognitive systems. CSE has since its beginning become a recognised scientific discipline, sometimes also referred to as Cognitive Engineering. The concept of a Joint Cognitive System (JCS) has in particular become widely used as a way of understanding how complex socio-technical systems can be described with varying degrees of resolution. The more than 20 years of experience with CSE has been described extensively .

Configuration Management

Like systems engineering, Configuration Management as practiced in the defence and aerospace industry is a broad systems-level practice. The field parallels the taskings of systems engineering; where systems engineering deals with requirements development, allocation to development items and verification, Configuration Management deals with requirements capture, traceability to the development item, and audit of development item to ensure that it has achieved the desired functionality that systems engineering and/or Test and Verification Engineering have proven out through objective testing.

Control engineering

Control engineering and its design and implementation of control systems, used extensively in nearly every industry, is a large sub-field of systems engineering. The cruise control on an automobile and the guidance system for a ballistic missile are two examples. Control systems theory is an active field of applied mathematics involving the investigation of solution spaces and the development of new methods for the analysis of the control process.

Industrial engineering

Industrial engineering is a branch of engineering that concerns the development, improvement, implementation and evaluation of integrated systems of people, money, knowledge, information, equipment, energy, material and process. Industrial engineering draws upon the principles and methods of engineering analysis and synthesis, as well as mathematical, physical and social sciences together with the principles and methods of engineering analysis and design to specify, predict and evaluate the results to be obtained from such systems.

Interface design

Interface design and its specification are concerned with assuring that the pieces of a system connect and inter-operate with other parts of the system and with external systems as necessary. Interface design also includes assuring that system interfaces be able to accept new features, including mechanical, electrical and logical interfaces, including reserved wires, plug-space, command codes and bits in communication protocols. This is known as extensibility. Human-Computer Interaction (HCI) or Human-Machine Interface (HMI) is another aspect of interface design, and is a critical aspect of modern systems engineering. Systems engineering principles are applied in the design of network protocols for local-area networks and wide-area networks.

Mechatronic engineering

Mechatronic engineering, like Systems engineering, is a multidisciplinary field of engineering that uses dynamical systems modeling to express tangible constructs. In that regards it is almost indistinguishable from Systems Engineering, but what sets it apart is the focus on smaller details rather than larger generalizations and relationships. As such, both fields are distinguished by the scope of their projects rather than the methodology of their practice.

Operations research

Operations research supports systems engineering. The tools of operations research are used in systems analysis, decision making, and trade studies. Several schools teach SE courses within the operations research or industrial engineering department, highlighting the role systems engineering plays in complex projects. Operations research, briefly, is concerned with the optimization of a process under multiple constraints.

Performance engineering

Performance engineering is the discipline of ensuring a system will meet the customer's expectations for performance throughout its life. Performance is usually defined as the speed with which a certain operation is executed or the capability of executing a number of such operations in a unit of time. Performance may be degraded when an operations queue to be executed is throttled when the capacity is of the system is limited. For example, the performance of a packet-switched network would be characterised by the end-to-end packet transit delay or the number of packets switched within an hour. The design of high-performance systems makes use of analytical or simulation modeling, whereas the delivery of high-performance implementation involves thorough performance testing. Performance engineering relies heavily on statistics, queueing theory and probability theory for its tools and processes.

Program management and project management.

Program management (or programme management) has many similarities with systems engineering, but has broader-based origins than the engineering ones of systems engineering. Project management is also closely related to both program management and systems engineering.

Proposal engineering

Proposal engineering is the application of scientific and mathematical principles to design, construct, and operate a cost-effective proposal development system. Basically, proposal engineering uses the "systems engineering process" to create a cost effective proposal and increase the odds of a successful proposal.

Reliability engineering

Reliability engineering is the discipline of ensuring a system will meet the customer's expectations for reliability throughout its life; i.e. it will not fail more frequently than expected. Reliability engineering applies to all aspects of the system. It is closely associated with maintainability, availability and logistics engineering. Reliability engineering is always a critical component of safety engineering, as in failure modes and effects analysis (FMEA) and hazard fault tree analysis, and of security engineering. Reliability engineering relies heavily onstatistics, probability theory and reliability theory for its tools and processes.

Safety engineering

The techniques of safety engineering may be applied by non-specialist engineers in designing complex systems to minimize the probability of safety-critical failures. The "System Safety Engineering" function helps to identify "safety hazards" in emerging designs, and may assist with techniques to "mitigate" the effects of (potentially) hazardous conditions that cannot be designed out of systems.

Security engineering

Security engineering can be viewed as an interdisciplinary field that integrates the community of practice for control systems design, reliability, safety and systems engineering. It may involve such sub-specialties as authentication of system users, system targets and others: people, objects and processes.

Software engineering

From its beginnings, software engineering has helped shape modern systems engineering practice. The techniques used in the handling of complexes of large software-intensive systems has had a major effect on the shaping and reshaping of the tools, methods and processes of SE.

Chapter 12

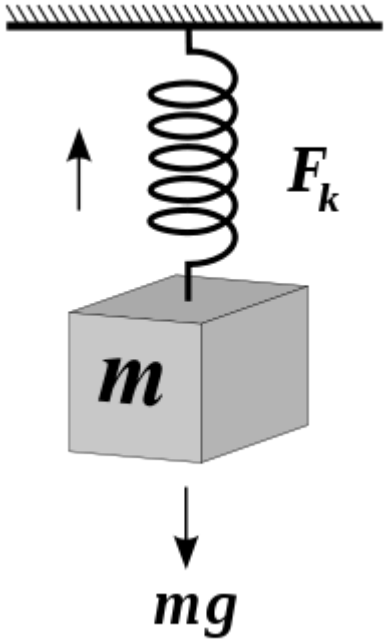
Shock Indicator

A shock indicator is a mechanical device that detects and records mechanical-shocks experienced by it. It is a common practice to attach shock indicators to expensive goods, that are to be shipped. This way, the magnitude and the time of shock is recorded and ensured that the delicate and expensive instruments are not mishandled during shipping. The shock indicator usually has a visual indicator when the shock-threshold is reached or exceeded, in the form of a broken spring or a digital display.

Introduction

The shock indicator was invented to ensure safe shipment of expensive goods and to hold the right people liable for any damage caused during the process. As any other invention, this one too has undergone various stages of development. The earliest shock indicators were simple mechanical devices with a spring-load assembly. The newer indicators, however, come integrated with a complex set of electronic circuits that detect, measure and record the shocks. There are many companies that make these indicators and these indicators are still a matter of active research.

Spring-Load assembly



Hooke's law describes how far the spring will stretch with a specific force

The spring-load shock indicators involved a simple extensible spring with a load attached to it at one end. The spring constant was carefully chosen, based on the rigidity of the shipment material and the forces involved during the shocks. When these forces exceeded the limits, as set by the spring constant, the spring would either be broken or extended beyond its elastic limit. The law governing this relation is called the Hooke's law of elasticity. Mathematically, Hooke's law states that

$$\mathbf{F} = -k\mathbf{x},$$

where

x is the displacement of the end of the spring from its equilibrium position (in SI units: "m");
F is the restoring force exerted by the material (in SI units: "N"); and
k is the *force constant* (or *spring constant*) (in SI units: "N·m^{-1}" or "kgs^{-2}").

When this holds, the behavior is said to be *linear*. If shown on a graph, the line should show a direct variation. There is a negative sign on the right hand side of the equation because the restoring force always acts in the opposite direction of the displacement (for example, when a spring is stretched to the left, it pulls back to the right).

Modern indicators

The modern indicators that rely heavily on their supporting electronics, are much more complex than their mechanical counterparts. They can not only detect and measure the shock, but also record the entire spectrum of shocks experienced by them over a period of several months. Their power consumption is rather low and are fueled by simple batteries.

Chapter 13

Sneak Circuit Analysis

Sneak Circuit Analysis is a vital part of the safety assurance of safety-critical electronic and electro-mechanical systems.

Sneak conditions are defined as latent hardware, software, or integrated conditions that may cause unwanted actions or may inhibit a desired function, and are not caused by component failure.

Sneak Circuit Analysis (SCA) is used in safety-critical systems to identify sneak (or hidden) paths in electronic circuits and electro-mechanical systems that may cause unwanted action or inhibit desired functions. The analysis is aimed at uncovering design flaws that allow for sneak conditions to develop. The sneak circuit analysis technique differs from other system analysis techniques in that it is based on identification of designed-in inadvertent modes of operation and is not based on failed equipment or software.

SCA is most applicable to circuits that can cause irreversible events. These include:

a. Systems that control or perform active tasks or functions
b. Systems that control electrical power and its distribution.
c. Embedded code which controls and times system functions.

Sneak conditions are classified into four basic types:

1. Sneak paths - unintended electrical (current) paths within a circuit and its external interfaces.
2. Sneak timing–unexpected interruption or enabling of a signal due to switch circuit timing problems which may cause or prevent the activation or inhibition of a function at an unexpected time.
3. Sneak indications–undesired activation or deactivation of an indicator which may cause an ambiguous or false display of system operating conditions.
4. Sneak labels–incorrect or ambiguous labeling of a switch which may cause operator error through inappropriate control activation.

Historical background

SCA is a detailed examination of switching circuitry that controls irreversible functions such as squibs and latches. The former Military Standard for Reliability Program (MIL-STD-785B) defines SCA (Task 205) as a task " … to identify latent paths which cause occurrence of unwanted functions or inhibit desired functions, assuming all components are functioning properly."

The Mercury-Redstone launch failure 1961 and a number of other mishaps caused by sneak circuits in missiles and torpedoes caused the military services and NASA to require formal procedures for prevention of these incidents. The first computer aided implementation of SCA was for the NASA Apollo program in 1967 by the Boeing Company . Among early publications in the field are a 1970 Boeing report "Sneak Circuit Analysis Handbook" by J. P. Rankin and C. F. White (NTIS N71-12487), and a 1977 AGARD report by J. L. Wilson and R. C. Clardy "Sneak Circuit Analysis Application to Control System Design" (AD A041042). By 1980 the requirements for SCA had become sufficiently common to lead to the Navy publication of a "Contract and Management Guide for Sneak Circuit Analysis" (NAVSEA-TE001-AA-GYD-010/SCA) . Subsequent efforts led to several Air Force reports in 1990: "Sneak Circuit Analysis for the Common Man", Rome Air Development Center Technical Report, RADC-TR-89-223, October, 1989 and "Intergration of Sneak Analysis with Design", Air Development Center Technical Report, RADC-TR-90-109, (June, 1990.) .

Current standards and guidelines include NASA's Sneak Circuit Analysis Guideline for Electromechanical Systems (PD-AP-1314) and AIAA's Performance-Based Sneak Circuit Analysis (SCA) Requirements (BSR/ANSI/AIAA S-102.2.5-2xxx) .

Sneak circuit example

Most sneak circuits reported from production systems are too complex to describe in an introductory discussion. However, the essential characteristics of a sneak circuit can be explained with a hypothetical example of an aircraft cargo door release latch as shown in Figure 1-1.

Figure 1-1 Sneak Circuit in Cargo Door Latching Function

To prevent unintended opening of the cargo door in flight, the normal cargo door control (CARGO OPEN) is powered in series with the GEAR DOWN switch. This permits routine opening on the ground. But there can be emergencies that require jettisoning cargo, and to be prepared for these there is an EMERGENCY CARGO OPEN switch that may be guarded with a safety wire to prevent its unintended operation. Now assume that an in-flight emergency exists that requires opening the cargo door. The flight personnel flips the normal CARGO OPEN switch and nothing happens (since the GEAR DOWN switch is open). It is realized that it is necessary to close the EMERGENCY CARGO OPEN switch, and when that action is taken the cargo door latch is indeed released, permitting the door to be opened. But at the same time the landing gear is lowered, not a

desired action and one that probably will aggravate the emergency. The condition that permits this undesired lowering of the landing gear to occur when both cargo door switches are closed is a sneak circuit.

Two observations about this sneak circuit apply generally:

1. Switches or other control elements are operated in an unusual or even prohibited manner

2. The unintended function (in this example the lowering of the landing gear) is associated with current flow through a circuit element that is opposite to the intended current flow.

The latter of these conditions permits elimination of the sneak circuit by inserting a diode as shown in Figure 1-2.

Figure 1-2. Corrected Cargo Door Latching Circuit

Conventional SCA Techniques

The original SCA techniques depended on recognition of circuit patterns or “clues” for the detection of potential sneak circuits. The most common of these circuit patterns are shown in Figure 1-3.

Figure 1-3 Circuit Patterns for Sneak Circuit Analysis

The box symbols represent arbitrary circuit elements; in many cases the individual legs of the patterns include switches. It will be recognized that the leg containing the normal CARGO OPEN switch in Figure 1-1 constitutes the middle horizontal leg of an H-pattern. The inverted Y is also called a ground dome; note that the two bottom legs terminate in different ground levels, such as chassis ground and signal ground. The Y-pattern is also called a power dome. The two upper legs terminate at different power sources, such as V1 and V2.

To facilitate the recognition of these patterns or clues, the schematic diagrams were redrawn as “network trees”, with power sources at the top and grounds at the bottom. In sneak circuit analysis both positive and negative sources will be shown at the top of the figure. Because searching for the patterns is very labor intensive, computer programs were developed to recognize the common clues in the network trees. Main frame computers had to be utilized to perform the topological searches even on small designs. Despite the aid of computers, SCA remained a very expensive and lengthy activity, and it was usually conducted only after the circuit design was frozen to avoid having to repeat it after changes. This had a distinct disadvantage when a sneak circuit was detected: it became very expensive to fix it because usually the circuit card or cabling was already in production.

An effort of the Rome Air Development Center (now part of the USAF Research Laboratories) directed at finding techniques that would permit sneak circuit analysis to be conducted as part of the design activity led to the “bi-path” methodology, developed by SoHaR. The “bi-path” algorithm was implemented into an automated software tool that allowed large designs to be analyzed very quickly early in the design phase of a safety-critical circuit.

Editing

Editing is used to eliminate paths that cannot contribute to operation of sensitive elements (elements that can lead to critical actions). Circuits that control squibs or latches usually contain computational, instrumentation, and switching elements. An example of the integration of these functions for a hypothetical and simplified missile detonation system is shown in Figure 1-4. The computational elements at the top of the figure establish the conditions for operation of the pre-arm, arm, and detonate switches. The heavy lines constitute the switching elements. The instrumentation functions are shown in the lower part of the figure. Sneak circuit analysis encompasses only the switching functions; the computational and instrumentation elements are eliminated from the traced paths.

This editing is justified because the connection between the computational elements and the switches (shown as dashed lines in the figure) is non-conducting. In most cases the output of the computational element goes to the gate of a MOSFET while the switching function uses the source-drain path. The computational elements are typically quite complex and their failure probability is much higher than that of the switching path. Thus safeguards are provided to tolerate the worst failure modes of these devices and sneak circuit analysis of the computational elements is not required.

Figure 1-4. Hypothetical Missile Detonation System

The elimination of the instrumentation functions is justified by the isolation resistors at the connection with the switching function. The resistance values are typically of the order of 10k ohms. Since the switching voltage is in the 20V–30V range, the current flow through the isolation resistors cannot exceed a few milliamperes, while squibs fire only above 1 ampere. In addition to this editing of major blocks, individual elements connected to the switching circuit may have to be eliminated or modified by editing as shown in the examples of Figure 1-5. The feedback resistor Rf constitutes an intentional bi-path (not a sneak circuit). Its high resistance prevents significant current flow. In part b. of the figure a mechanical connection keeps switches S1 and S2 from being closed at the same time, preventing a power-to-power tie.

Chapter 14

Safety Instrumented System

A **Safety Instrumented System (SIS)** is a form of process control usually implemented in industrial processes, such as those of a factory or an oil refinery. The SIS performs specified functions to achieve or maintain a safe state of the process when unacceptable or dangerous process conditions are detected. Safety instrumented systems are separate and independent from regular control systems but are composed of similar elements, including sensors, logic solvers, actuators and support systems.

The specified functions, or *safety instrumented functions* (SIF) are implemented as part of an overall risk reduction strategy which is intended to reduce the likelihood of identified hazardous events involving a catastrophic release. The safe state is a state of the process operation where the hazardous event cannot occur. The safe state should be achieved within one-half of the process safety time. Most SIF are focused on preventing catastrophic incidents.

The correct operation of an SIS requires a series of equipment to function properly. It must have sensors capable of detecting abnormal operating conditions, such as high flow, low level, or incorrect valve positioning. A logic solver is required to receive the sensor input signal(s), make appropriate decisions based on the nature of the signal(s), and change its outputs according to user-defined logic. The logic solver may use electrical, electronic or programmable electronic equipment, such as relays, trip amplifiers, or programmable logic controllers. Next, the change of the logic solver output(s) results in the final element(s) taking action on the process (e.g. closing a valve) to bring it to a safe state. Support systems, such as power, instrument air, and communications, are generally required for SIS operation. The support systems should be designed to provide the required integrity and reliability.

International standard IEC 61511 was published in 2003 to provide guidance to end-users on the application of Safety Instrumented Systems in the process industries. This standard is based on IEC 61508, a generic standard for design, construction, and operation of electrical/electronic/programmable electronic systems. Other industry sectors may also have standards that are based on IEC 61508, such as IEC 62061 (machinery systems), IEC 62425 (for railway signaling systems), IEC 61513 (for nuclear systems), and ISO 26262 (for road vehicles, currently a draft international standard).

Other names

Other terms often used in conjunction with and/or to describe safety instrumented systems include:

- Critical control system
- Safety shutdown system
- Protective instrumented system
- Equipment protection system
- Emergency shutdown system
- Safety critical system
- Interlock (engineering)
- Interlocking (railway signalling)

SIS reliability

What a SIS shall do (*the functional requirements*) and how well it must perform (*the safety integrity requirements*) may be determined from Hazard and operability studies (HAZOP), layers of protection analysis (LOPA), risk graphs, and so on. All techniques are mentioned in IEC 61511 and IEC 61508. During SIS design, construction, installation, and operation, it is necessary to verify that these requirements are met. The functional requirements may be verified by design reviews, such as failure modes, effects, and criticality analysis (FMECA) and various types of testing, for example factory acceptance testing, site acceptance testing, and regular functional testing.

The safety integrity requirements may be verified by reliability analysis. For SIS that operates on demand, it is often the probability of failure on demand (PFD) that is calculated. In the design phase, the PFD may be calculated using generic reliability data, for example from OREDA. Later on, the initial PFD estimates may be updated with field experience from the specific plant in question.

It is not possible to address all factors that affect SIS reliability through reliability calculations. It is therefore also necessary to have adequate measures in place (e.g., procedures and competence) to avoid, reveal, and correct SIS related failures.

SIS examples

Safety instrumented systems are most often used in process (i.e., refineries, chemical, nuclear, etc.) facilities to provide protection such as:

- High fuel gas pressure initiates action to close the main fuel gas valve.
- High reactor temperature initiates action to open cooling media valve.
- High distillation column pressure initiates action to open a pressure vent valve.

Chapter 15

System Safety

The **system safety** concept calls for a risk management strategy based on identification, analysis of hazards and application of remedial controls using a systems-based approach. This is different from traditional safety strategies which rely on control of conditions and causes of an accident based either on the Epidemiological analysis or as a result of investigation of individual past accidents.. The concept of system safety is useful in demonstrating adequacy of technologies when difficulties are faced with probabilistic risk analysis. The underlying principle is one of synergy: a whole is more than sum of its parts. Systems-based approach to safety requires the application of scientific, technical and managerial skills to hazard identification, hazard analysis, and elimination, control, or management of hazards throughout the life-cycle of a system, program, project or an activity or a product. "Hazop" is one of several techniques available for identification of hazards.

System approach

A system is defined as a set or group of interacting, interrelated or interdependent elements or parts that are organized and integrated to form a collective unity or a unified whole to achieve a common objective. This definition lays emphasis on the interactions between the parts of a system and the external environment to perform a specific task or function in the context of an operational environment. This focus on interactions is to take a view on the expected or unexpected demands (inputs) that will be placed on the system and see whether necessary and sufficient resources are available to process the demands. These might take form of stresses. These stresses can be either expected, as part of normal operations, or unexpected, as part of unforeseen acts or conditions that produce beyond-normal (i.e., abnormal) stresses. This definition of a system, therefore, includes not only the product or the process but also the influences that the surrounding environment (including human interactions) may have on the product's or process's safety performance. Conversely, system safety also takes into account the effects of the system on its surrounding environment. Thus, a correct definition and management of interfaces becomes very important. Broader definitions of a system are the hardware, software, human systems integration, procedures and training. Therefore system safety as part of the systems engineering process should systematically address all of these

domains and areas in engineering and operations in a concerted fashion to prevent, eliminate and control hazards.

A “system", therefore, has implicit as well as explicit definition of boundaries to which the systematic process of hazard identification, hazard analysis and control is applied. The system can range in complexity from a manned spacecraft to an autonomous machine tool. The system safety concept helps the system designer(s) to model, analyse, gain awareness about, understand and eliminate the hazards, and apply controls to achieve an acceptable level of safety. Ineffective decision making in safety matters is regarded as the first step in the sequence of hazardous flow of events in the "Swiss Cheese" model of accident causation. Communications regarding system risk have an important role to play in correcting risk perceptions by creating, analysing and understanding information model to show what factors create and control the hazardous process. For almost any system, product, or service, the most effective means of limiting product liability and accident risks is to implement an organized system safety function, beginning in the conceptual design phase and continuing through to its development, fabrication, testing, production, use and ultimate disposal. The aim of the system safety concept is to gain assurance that a system and associated functionality behaves in a safe manner and is safe to operate. This assurance is necessary. Technological advances in the past have produced positive as well as negative effects.

Root cause analysis

A root cause analysis identifies the set of multiple causes that together might create a potential accident. Root cause techniques have been successfully borrowed from other disciplines and adapted to meet the needs of the system safety concept, most notably the tree structure from Fault Tree Analysis, which was originally an engineering technique. The root cause analysis techniques can be categorised into two groups: a) tree techniques, and b) check list methods. There are several root causal analysis techniques, e.g. Management Oversight and Risk Tree (MORT) analysis. Others are Event and Causal Factor Analysis (ECFA),Multilinear Events Sequencing, Sequentially Timed Events Plotting Procedure, Savannah River Plant Root Cause Analysis System.

Use in other fields

Safety engineering

Safety engineering describes some of the methods used in nuclear and other industries. Traditional safety engineering techniques are focused on the consequences of human error and do not investigate the causes or reasons for the occurrence of human error. System safety concept can be applied to this traditional field to help identify the set of conditions for safe operation of the system. Modern and more complex systems with computer application and controls require functional hazard analyses and a set of detailed specifications at all levels that address safety attributes to be inherent in the design.

Weapon system safety

Weapon System Safety is an important application of the system safety field, due to the potentially destructive effects of a system failure or malfunction. A healthy skeptical attitude towards the system, when it is at the requirements definition and drawing-board stage, by conducting functional hazard analyses, would help in learning about the factors that create hazards and mitigations that control the hazards. A rigorous process is usually formally implemented as part of systems engineering to influence the design and improve the situation before the errors and faults weaken the system defences and cause accidents.

Typically weapons systems pertaining to ships, land vehicles, guided missiles and aircraft differ in hazards and effects; some are inherent, such as explosives, and some are created due to the specific operating environments (as in, for example, aircraft sustaining flight). In the military aircraft industry safety-critical functions are identified and the overall design architecture of hardware, software and human systems integration are thoroughly analyzed and explicit safety requirements are derived and specified during proven hazard analysis process to establish safeguards to ensure essential functions are not lost or function correctly in a predictable manner. Prevention of mishaps is the objective.

Chapter 16

Partial Stroke Testing

Partial stroke testing (or PST) is a technique used in a safety instrumented system to allow the user to test a percentage of the possible failure modes of a shut down valve without the need to physically close the valve.

Standards

Partial stroke testing is an accepted petroleum industry standard technique and is also quantified in detail by regulatory bodies such as the International Electrotechnical Commission (or IEC) and the Instrument Society of America (or ISA). The following are the standards appropriate to these bodies.

- **IEC61508** - Functional safety of electrical/electronic/programmable electronic safety-related systems
- **IEC61511** - Functional safety - Safety instrumented systems for the process industry sector
- **ANSI/ISA-84.00.01** - Functional Safety: Safety instrumented systems for the process industry sector

Measuring safety performance

IEC61508 adapts a Safety life cycle approach to the management of plant safety. During the design phase of this life cycle of a safety system the required safety performance level is determined using techniques such as Markov analysis, FMEA, Fault tree analysis and Hazop. These techniques allow the user to determine the potential frequency and consequence of hazardous activities and to quantify the level of risk. A common method for this quantification is the Safety integrity level. This is quantified from 1 to 4 with level 4 being the most hazardous.

Once the SIL level is determined this specifies the required performance level of the safety systems during the operational phase of the plant. The metric for measuring the performance of a safety funciton is called the Average Probability of Failure on Demand (or PFD_{avg}) and this correlates to the SIL level as follows

SIL	PFD_{avg}
4	$\geq 10^{-5}$ to $<10^{-4}$
3	$\geq 10^{-4}$ to $<10^{-3}$
2	$\geq 10^{-3}$ to $<10^{-2}$
1	$\geq 10^{-2}$ to $<10^{-1}$

One method of calculating the PFD_{avg} for a basic safety function with no redundancy is using the formula

$$\mathbf{PFD_{avg} = [(1\text{-}DC) * \lambda_D * (TI_{FC}/2)] + [DC * \lambda_D * (TI_{PST}/2)]}$$

Where:

DC = Diagnostic Coverage of the partial stroke test.
λ_D = The dangerous failure rate of the safety function.
TI_{FC} = The full closure interval, i.e. how often the valve must be full closed for testing.
TI_{PST} = The partial stroke test interval.

The diagnostic coverage is a measure of how effective the partial stroke test is and the higher the DC the great the effect the test.

Benefits

The benefits of using PST are not limited to simply the safety performance but gains can also be made in the production performance of a plant and the capital cost of a plant.. These are summarised as follows

Safety benefits

Gains can be made in the following areas by the use of PST.

- Reducing the Probability of failure on Demand.
- Improving the Safe Failure Fraction (SFF).

Production benefits

There are a number of areas where production efficiency can be improved by the successful implementation of a PST system.

- Extension of the time between compulsory plant shutdowns.
- Predicting potential valve failures facilitating the pre-ordering of spare parts.
- Prioritisation of maintenance tasks.

Capital cost benefits

If the gains of the SFF are of an appropriate level the need for costly redundant valves may be eliminated

Techniques

There are a number of different techniques available for partial stroke testing available and the selection of the most appropriate technique depends on the main benefits the operator is trying to gain.

Mechanical jammers

Mechanical jammers are devices where a device is inserted into the valve and actuator assembly what physically prevents the valve from moving past a certain point. These are used in cases where accidentally shutting the valve would have severe consequences.

However these kind of devices are not completely suitable for functional safety systems as the safety funciton is off-line for the duration of the test. In addition it is not possible to automate this testing of these devices therefore these can introduce a potential element of human error onto the operation of the system.

Examples of this kind of device include snail cams fitted to the valve stem and also adjustable actuator end stops

Pneumatic valve positioners

The basic principle behind partial stroke testing is that the valve is moved to a predetermined position in order to determine the performance of the shut down valve. This led to the adaptation of pneumatic positioners used on flow control valve for use in partial stroke testing. These systems are often suitable for use on shutdown valves up to and including SIL3

These are however limited to use on pneumatically actuated valves

Electronic timer control systems

Timer control systems use a configurable electronic timer that connects between the supply from the ESD system and the solenoid valve. In order to perform a test the timer de-energises the solenoid valve to simulate a shutdown and re-energises the soleniod when the required degree of partial stroke is reached. These systems are fundamentally a miniature PLC dedicated to the testing of the valve.

Due to their nature these devices do not actually form part of the safety function and are therefore 100% fail safe. With the addition of a pressure sensor and/or a position sensor for feedback timer systems are also capable of providing intelligent diagnostics in order

to diagnose the performance of all components including the valve, actuator and solenoid valves.

In addition timers are capable of operating with any type of fluid power actuator and can also be used with subsea valves where the solenoid valve is located top-side

Chapter 17

Shut Down Valve & Safety Engineer

Shut Down Valve

A **shut down valve** (also referred to as **SDV** or **Emergency shutdown valve**, **ESV**, **ESD**, or **ESDV**) is an actuated valve designed to stop the flow of a hazarous fluid upon the detection of a dangerous event. This provides protection against possible harm to people, equipment or the environment. Shutdown valves form part of a Safety instrumented system. The process of providing automated safety protection upon the detection of a hazardous event is called Functional Safety

Shutdown valves are primarily associated with the petroleum industry although other industries may also require this type of protection system

Types of valve

Metal seated ball valves are used as shut-down valves (SDV's). Use of metal seated ball valves leads to overall lower costs when taking into account lost production and inventory, and valve repair costs resulting from the use of soft seated ball valves which have a lower initial cost.

Straight-through flow valves, such as rotary-shaft ball valves, are typically high-recovery valves. High recovery valves are valves that lose little energy due to little flow turbulence. Flow paths are straight through. Rotary control valves, butterfly valve and ball valves are good example

Types of actuation

Pneumatically actuated shut down valve

As shutdown valves are form part of a SIS it is necessary to operate the valve by means of an actuator. These actuators are normally fail safe fluid power type. Typical examples of these are:

- **Pneumatic cylinder**
- **Hydraulic cylinder**
- **Electro-hydraulic actuator**

In addition to the fluid type actuators also vary in the manner in which the energy is stored to operate the valve on demand as follows

- **single-acting cylinder** - Or spring return where the energy is stored by means of a compressed spring
- **double-acting cylinder** - Energy is stored using a volume of compressed fluid

The type of actuation required depends upon the application, site facilities and also the physical space available although the majority of actuators used for shutdown valves are of the spring return type due to the fail safe nature of spring return systems

Measuring performance

For shutdown valves used in safety instrumented systems it is essential to know that the valve is capable of providing the required level of safety performance and that the valve will operate on demand. The required level of performance is dicated by the Safety integrity level (SIL). In order to adhere to this level of performance it is necessary to test the valve. There are 2 types of testing methods available being

- **Proof test** - A manual test that that allows the operator to determine whether the valve is in the "as good as new" condition by testing for all possible failure modes and requires a plant shutdown
- **Diagnostic Teset** - An automated on-line test that will detect a percentage of the possible failrue modes of the shudown valve. An example of this for a shutdown valve would be a partial stroke test

Safety Engineer

Scope of a Safety Engineer

To perform their professional functions, safety engineering professionals must have education, training and experience in a common body of knowledge. They need to have a fundamental knowledge of physics, chemistry, biology, physiology, statistics, mathematics, computer science, engineering mechanics, industrial processes, business, communication and psychology. Professional safety studies include industrial hygiene and toxicology, design of engineering hazard controls, fire protection, ergonomics, system and process safety, system safety, safety and health program management, accident investigation and analysis, product safety, construction safety, education and training methods, measurement of safety performance, human behavior, environmental safety and health, and safety, health and environmental laws, regulations and standards. Many safety engineers have backgrounds or advanced study in other disciplines, such as management and business administration, engineering, system engineering, requirements engineering, reliability engineering, maintenance , human factors, operations, education, physical and social sciences and other fields. Others have advanced study in safety. This extends their expertise beyond the basics of the safety engineering profession.

Functions of a Safety Engineer

The major areas relating to the protection of people, property and the environment are:

- Anticipate, identify and evaluate hazardous conditions and practices.
- Develop hazard control designs, methods, procedures and programs.

- Implement, administer and advise others on hazard control programs.
- Measure, audit and evaluate the effectiveness of hazard control programs.
- Draft a future safety plan and statement based on real time experiences and facts.

Personality and role

Oddly enough, personality issues can be paramount in a safety engineer. They must be personally pleasant, intelligent, and ruthless with themselves and their organization. In particular, they have to be able to "sell" the failures that they discover, as well as the attendant expense and time needed to correct them. They can be the messengers of bad news.

Safety engineers have to be ruthless about getting facts from other engineers. It is common for a safety engineer to consider software, chemical, electrical, mechanical, procedural, and training problems in the same day. Often the facts can be very uncomfortable as many safety related issues point towards mediocre management systems or worse, questionable business ethics.

Teamwork

It's difficult and expensive to retrofit safety into an unsafe system. To prevent safety problems, an organization should therefore treat safety as an early design and "architecture" activity, using the principles of Inherent safety, rather than as a "paperwork requirement" to be cleaned up after the real design is done.

Safety engineers also must work in a team that includes other engineering specialties, quality assurance, quality improvement, regulatory compliance specialists, educators and lawyers.

Subpar safety and quality problems often indicate underlying deficiencies in an organization's goals, recruitment, succession, training, management systems and business culture.

Safety often works well in a true matrix-management organization, in which safety is a managed discipline integrated into a project plan.

Chapter 18

Functional Safety

Functional Safety is the part of the overall safety of a system or piece of equipment that depends on the system or equipment operating correctly in response to its inputs, including the safe management of likely operator errors, hardware failures and environmental changes.

Objective of Functional Safety

The objective of Functional Safety is freedom from unacceptable risk of physical injury or of damage to the health of people either directly or indirectly (through damage to property or to the environment).

Functional Safety is intrinsically end-to-end in scope in that it has to treat the function of a component or subsystem as part of the function of the whole system. This means that whilst Functional Safety standards focus on Electrical, Electronic and Programmable Systems (E/E/PS), the end-to-end scope means that in practice Functional Safety methods have to extend to the non-E/E/PS parts of the system that the E/E/PS actuates, controls or monitors.

Achieving Functional Safety

Functional Safety is achieved when every specified safety function is carried out and the level of performance required of each safety function is met. This is normally achieved by a process that includes the following steps as a minimum:

1. Identifying what the required safety functions are. This means the hazards and safety functions have to be known. A process of function reviews, formal HAZIDs, HAZOPs and Accident Reviews are applied to identify these.

2. Assessment of the risk-reduction required by the safety function. This will involve a Safety Integrity Level (SIL) Assessment. A Safety Integrity Level (SIL) applies to an

end-to-end safety function of the safety-related system, not just to a component or part of the system.

3. Ensuring the safety function performs to the design intent, including under conditions of incorrect operator input and failure modes. This will involve having the design and lifecycle managed by qualified and competent engineers carrying out processes to a recognised functional safety standard. In Europe, that standard is IEC EN 61508, or one of the industry specific standards derived from IEC EN 61508.

4. Verification that the system meets the assigned SIL, by determining the Mean Time Between Failures and the Safe Failure Fraction (SFF). The unsafe failure fraction is the probability of the system failing in a dangerous (or critical) state, derived from a Failure Mode and Effects Analysis or (Failure Mode, Effects, and Criticality Analysis) of the system (FMEA or FMECA).

5. Conduct functional safety management audit. The safety lifecycle management audit is a mechanism used to help reduce systematic problems from appearing in the design of a product....the functional safety lifecycle management audit looks at those elements of the manufacturer’s process that may impact the quality of the functional safety of the product being produced.

Neither safety nor Functional Safety can be determined without considering the system as a whole and the environment with which it interacts. Functional Safety is inherently end-to-end in scope.

Certifying Functional Safety

Any claim of Functional Safety for a component, subsystem or system should be independently certified to one of the recognised Functional Safety standards. A certified product can then be claimed to be Functionally Safe to a particular Safety Integrity Level or a Performance Level in a specific range of applications: the certificate is provided to the customers with a test report describing the scope and limits of performance.

An important element of functional safety certification is on-going surveillance by the certification agency. This follow-up surveillance ensures that that product, sub-system, or system is still being manufactured in accordance with the what was originally certified for functional safety. Follow-up surveillance may occur as various frequencies depending on the certification agency, but will typically look at the product's hardware, software, as well as the manufacturer's ongoing compliance of functional safety management systems.

The principles underpinning Functional Safety were developed in the military, nuclear and aerospace industries, and then taken up by rail transport, process and control industries developing sector specific standards. Functional Safety standards are applied across all industry sectors dealing with safety critical requirements. Thousands of products and processes meet the standards based on IEC EN 61508: from bathroom showers, automotive safety products, medical devices, sensors, actuators, Process

Controllers from ABB, Siemens , and their integration by companies such as Capula to ships, aircraft and major plant.

In Europe, Functional Safety certification is supported by a well developed infrastructure . The CASS Scheme is the primary method by which products are certified to IEC EN 61508 and related standards, through accredited quality auditors. It is possible to certify both products and processes that manage the lifecycle of the product, (in which case, the company certified would then issue a certificate of conformity to that certification in respect of its relevant products).

The US FAA have similar Functional Safety certification processes, in the form of US RTCA DO-178B for software and DO-254 for hardware , which is applied throughout the aerospace industry.

In the USA, NASA developed an infrastructure for safety critical systems adopted widely by industry, both in North America and elsewhere, with a standard , supported by guidelines . The NASA standard and guidelines are built on ISO 12207, which is a good software practice standard rather than a safety critical standard, hence the extensive nature of the documentation NASA has been obliged to add, compared to using a purpose designed standard such as EN 61508 with the CASS Templates. A certification process for systems developed in accord with the NASA guidelines exists .

Modern E/E/PS medical devices are being certified to 501(k) on the basis of the industry sector specific IEC EN 62304 standard, based on IEC EN 61508 concepts.

MISRA in the automotive industry are moving standards towards IEC EN 61508 in the development of industry specific standards.

Contemporary Functional Safety Standards

The primary Functional Safety standards in current use are listed below:

- IEC EN 61508 Parts 1 to 3 is a core Functional Safety standard, applied widely to all types of safety critical E/E/PS and to systems with a safety function incorporating E/E/PS.
- UK Defence Standard 00-56 Issue 2
- US RTCA DO-178B North American Avionics Software
- US RTCA DO-254 North American Avionics Hardware
- EUROCAE ED-12B European Airborne Flight Safety Systems
- IEC 62304 - Medical Device Software
- IEC 61513, Nuclear power plants – Instrumentation and control for systems important to safety – General requirements for systems, based on EN 61508
- IEC 61511-1, Functional safety – Safety instrumented systems for the process industry sector – Part 1: Framework, definitions, system, hardware and software requirements, , based on EN 61508

- IEC 61511-2, Functional safety – Safety instrumented systems for the process industry sector – Part 2: Guidelines for the application of IEC 61511-1, , based on EN 61508
- IEC 61511-3, Functional safety – Safety instrumented systems for the process industry sector – Part 3: Guidance for the determination of the required safety integrity levels, based on EN 61508
- IEC 62061, Safety of machinery - Functional safety of safety-related electrical, electronic and programmable electronic control systems, based on EN 61508
- EN 50128, Railway Industry Specific
- EN 50129, Railway Industry Specific
- NASA Safety Critical Guidelines
- UL 1998 Software in Programmable Components
- UL 991 Tests for Safety-Related Controls Employing Solid-State Devices
- ISO 13849 Safety of Machinery – Safety-related Parts of Control Systems

Chapter 19

Dependability

Dependability is a value showing the reliability of a person to others because of his/her integrity, truthfulness, and trustfulness, traits that can encourage someone to depend on him/her.

The wider use of this noun is in Systems engineering.

Dependability as applied to a computer system is defined by the IFIP 10.4 Working Group on Dependable Computing and Fault Tolerance as:

"*[..] the trustworthiness of a computing system which allows reliance to be justifiably placed on the service it delivers [..]*"

an alternative and broader definition is provided by IEC IEV 191-02-03:

"*dependability (is) the collective term used to describe the availability performance and its influencing factors : reliability performance, maintainability performance and maintenance support performance*"

This definition was developed by Technical Committee 56 *Dependability* of the International Electrotechnical Commission (IEC). The committee also develops and maintains International Standards in the field of dependability. The standards provide systematic methods and tools for dependability assessment and management of equipment, services and systems throughout their life cycles.

This concept can be further extended to encompass mechanisms to increase and maintain the Dependability of a system . Dependability can be thought of as being composed of three elements:

- **Attributes** - A way to assess the Dependability of a system
- **Threats** - An understanding of the things that can affect the Dependability of a system
- **Means** - Ways to increase the Dependability of a system

History

The field of **Dependability** grew out of previous related fields such as fault tolerance and system reliability in the 1960s. As interest in these fields increased during the 1970s and early part of the 1980s the term reliability began to be overloaded and was being used outside of its originally intended definition, as a measurement of failures in a system, to encompass more diverse measures which would now come under other classifications such as safety, integrity, etc. Jean-Claude Laprie thus coined the term Dependability to encompass these related disciplines in the early 1980.

The field of Dependability has evolved from these beginnings to be an internationally active field of research. This research is fostered by a number of prominent international conferences, notably the International Conference on Dependable Systems and Networks, the International Symposium on Reliable Distributed Systems and the International Symposium on Software Reliability Engineering.

The original definition of dependability for a computing system gathers the following attributes or non-functional requirements:

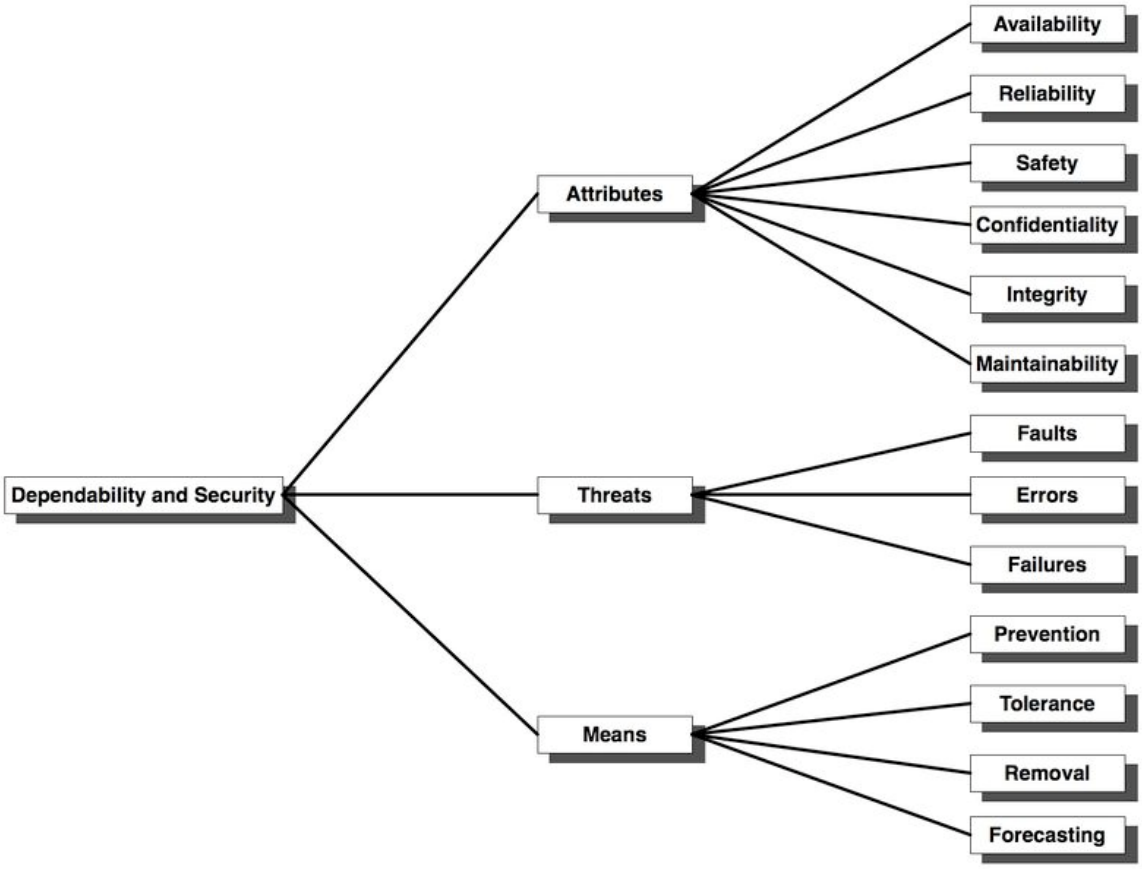
- Availability: readiness for correct service
- Reliability: continuity of correct service
- Maintainability: to undergo modifications and repairs

and combines them with the concepts of Threats and Failures to create Dependability.

This definition was further enhanced to incorporate Safety and Security.

Elements of dependability

Attributes



Taxonomy showing relationship between Dependability & Security and Attributes, Threats and Means (after Laprie et al.)

Attributes are qualities of a system. These can be assessed to determine its overall dependability using Qualitative or Quantitative measures. Avizienis et al. define the following Dependability Attributes:

- Availability - readiness for correct service
- Reliability - continuity of correct service
- Safety - absence of catastrophic consequences on the user(s) and the environment
- Integrity - absence of improper system alteration
- Maintainability - ability for a process to undergo modifications and repairs

As these definitions suggested, only Availability and Reliability are quantifiable by direct measurements whilst others are more subjective. For instance Safety cannot be measured directly via metrics but is a subjective assessment that requires judgmental information to be applied to give a level of confidence, whilst Reliability can be measured as failures over time.

Confidentiality, i.e. *the absence of unauthorized disclosure of information* is also used when addressing security. Security is a composite of Confidentiality, Integrity, and Availability. Security is sometimes classed as an attribute but the current view is to aggregate it together with dependability and treat Dependability as a composite term called Dependability and Security.

Practically, applying security measures to the appliances of a system generally improves the dependability by limiting the number of externally-originated errors.

Threats

Threats are things that can affect a system and cause a drop in Dependability. There are three main terms that must be clearly understood:

- Fault: A fault (which is usually referred to as a bug for historic reasons) is a defect in a system. The presence of a fault in a system may or may not lead to a failure, for instance although a system may contain a fault its input and state conditions may never cause this fault to be executed so that an error occurs and thus never exhibits as a failure.
- Error: An error is a discrepancy between the intended behaviour of a system and its actual behaviour inside the system boundary. Errors occur at runtime when some part of the system enters an unexpected state due to the activation of a fault. Since errors are generated from invalid states they are hard to observe without special mechanisms, such as debuggers or debug output to logs.
- Failure: A failure is an instance in time when a system displays behaviour that is contrary to its specification. An error may not necessarily cause a failure, for instance an exception may be thrown by a system but this may be caught and handled using fault tolerance techniques so the overall operation of the system will conform to the specification.

It is important to note that Failures are recorded at the system boundary. They are basically Errors that have propagated to the system boundary and have become observable. Faults, Errors and Failures operate according to a mechanism. This mechanism is sometimes known as a Fault-Error-Failure chain. As a general rule a fault, when activated, can lead to an error (which is an invalid state) and the invalid state generated by an error may lead to another error or a failure (which is an observable deviation from the specified behaviour at the system boundary).

Once a fault is activated an error is created. An error may act in the same way as a fault in that it can create further error conditions, therefore an error may propagate multiple times within a system boundary without causing an observable failure. If an error propagates outside the system boundary a failure is said to occur. A failure is basically the point at which it can be said that a service is failing to meet its specification. Since the output data from one service may be fed into another, a failure in one service may propagate into another service as a fault so a chain can be formed of the form: Fault leading to Error leading to Failure leading to Error, etc.

Means

Since the mechanism of a Fault-Error-Chain is understood it is possible to construct means to break these chains and thereby increase the dependability of a system. Four means have been identified so far:

1. Prevention
2. Removal
3. Forecasting
4. Tolerance

Fault Prevention deals with preventing faults being incorporated into a system. This can be accomplished by use of development methodologies and good implementation techniques.

Fault Removal can be sub-divided into two sub-categories: Removal During Development and Removal During Use.
Removal during development requires verification so that faults can be detected and removed before a system is put into production. Once systems have been put into production a system is needed to record failures and remove them via a maintenance cycle.

Fault Forecasting predicts likely faults so that they can be removed or their effects can be circumvented.

Fault Tolerance deals with putting mechanisms in place that will allow a system to still deliver the required service in the presence of faults, although that service may be at a degraded level.

Dependability means are intended to reduce the number of failures presented to the user of a system. Failures are traditionally recorded over time and it is useful to understand how their frequency is measured so that the effectiveness of means can be assessed.

Dependability of information systems and survivability

Recent works, such upon dependability take benefit of structured **information systems**, e.g. with SOA, to introduce a more efficient ability, the **survivability**, thus taking into account the degraded services that an Information System sustains or resumes after a non-maskable failure.

The flexibility of current frameworks encourage system architects to enable reconfiguration mechanisms that refocus the available, safe resources to support the most critical services rather than over-provisioning to build failure-proof system.

With the generalisation of networked information systems, **accessibility** was introduced to give greater importance to users' experience.

To take into account the level of performance, the measurement of **performability** is defined as "quantifying how well the object system performs in the presence of faults over a specified period of time".