Reliability

- Reliability of a software system is the probability that the software system will function without failure for a “specified time” in a specified environment. [Musa,et al.], given that the system was functioning properly at the beginning of the time period.
- Reliability can also be measured/expressed as Failure Intensity and/or Failure Density.
  - **Failure Intensity** is the rate failures are happening and is typically the inverse of MTBF (Mean Time Between Failures)) e.g. 2 failures per 1000 transactions.
  - **Failure Density** is the failures per KLOC or per Function Point of developed code e.g. 2 failure per KLOC etc.
- Reliability Testing - Although faults could be removed using static code analysis, unit testing, white box testing, black box testing etc.; testing for reliability, in particular, involves creating operational profile of the software system.
- Growth models and fault injection are some of the techniques used to predict reliability.
- Thread safety, stateful data transfer and data validation are among common Reliability Assurance techniques.

可靠性

- 软件系统的可靠性，是指该软件系统在特定环境下、于“规定时间”内无故障运行的概率。[Musa 等人], 指出，该定义的前提是系统在该时间段起始时刻处于正常运行状态。
- 可靠性亦可通过失效强度和/或失效密度
  - **失效强度**指单位时间内发生的失效次数，通常为平均无故障时间（MTBF，Mean Time Between Failures）的倒数，例如每 1000 笔事务发生 2 次失效。
  - **失效密度**指每千行代码（KLOC）或每个功能点（Function Point）所对应的失效次数，例如每千行代码发生 2 次失效等。
- 可靠性测试——尽管可通过静态代码分析、单元测试、白盒测试、黑盒测试等方法消除缺陷，但专门针对可靠性的测试则需构建软件系统的运行剖面。
- 增长模型与故障注入是用于预测可靠性的若干技术。
- 线程安全、有状态数据传输及数据验证等属于常见的可靠性保障技术。

Reliability Testing

- An **operational profile** is a complete set of operations with their probabilities of occurrence during the operational use of the software. Operational profiles underline how users will use the software system.
- Reliability testing thus involves repeatedly testing the **most probable use cases and inputs** to expose issues such as buffer overflow, memory leaks etc, and then measuring Failure Intensity (i.e. MTBF) etc., during this testing.
- Load testing is conducted to expose faults that wouldn't appear if the system was lightly loaded e.g. issues with concurrency and big data.

可靠性测试

- **运行剖面**是指软件在实际运行过程中所执行的全部操作及其各自出现概率的完整集合。运行剖面阐明了用户将如何使用该软件系统。
- 因此，可靠性测试需反复针对**最可能的使用场景和输入**开展测试，以暴露缓冲区溢出、内存泄漏等问题，并在此过程中测量故障强度（即平均无故障时间 MTBF）等指标。
- 负载测试旨在揭示系统在轻载情况下不会显现的缺陷，例如并发处理问题和大数据相关问题。

Operational Profile – A PhotoGallery app

User role	**Probability**
Casual user	0.80
Travel Blogger	0.20

Modes	**Probability**
Not Connected	0.10
Connected to WiFi	0.30
Connected to 3G/4G	0.60

Tasks	**Probability**
Specifying the folder path in settings page	.01
Specifying the URL of the blog site	.03
Take a photo	.20
View photos in the photo gallery	.30
Upload photo to a remote site	.10
View enlarged picture	.30
Delete a picture in the gallery	.06

运行概况——一款照片画廊应用

用户角色	**概率**
普通用户	0.80
旅行博主	0.20

使用模式	**概率**
未连接	0.10
已连接到 Wi-Fi	0.30
已连接到 3G/4G	0.60

任务	**概率**
在设置页面中指定文件夹路径	.01
指定博客网站的 URL	.03
拍摄照片	.20
在照片图库中查看照片	.30
将照片上传至远程网站	.10
查看放大后的图片	.30
删除图库中的一张图片	.06

Reliability Models

- Reliability depends on how the software is used and therefore defining a model of usage when characterizing reliability is required.
- Reliability typically improves over time as certain bugs are fixed. Such models are referred to as reliability growth or trend models.
- Since failures may happen at random time, probabilistic models of failure could be established and used to predict reliability.
- The objective behind these models is to help determine how many bugs still stay in the software and how long will it take to identify and remove these bugs.

可靠性模型

- 可靠性取决于软件的使用方式，因此在表征可靠性时，需定义相应的使用模型。
- 随着某些缺陷被修复，可靠性通常会随时间推移而提升。此类模型被称为可靠性增长模型或趋势模型。
- 由于故障可能在任意时刻随机发生，因此可建立并运用基于概率的故障模型来预测可靠性。
- 这些模型的目标在于帮助确定软件中尚存多少缺陷，以及识别并消除这些缺陷所需的时间。

Reliability Models – Growth Model

Error #	Time Since Last Failure (hours)	Failure Intensity
1	4	.25
2	6	.166
3	8	.125
4	5	.20
5	6	.166
6	10	.1
7	12	.083
8	16	.0625
9	18	.055
10	25	.04

In the above example

MTBF = (4+6+8+5+6+10+12+16+18+25)/10 = 11 hour

Given that the system has started correctly, the probability that the system is operational at time t is given as:

$$R(t) = e^{-t / MTBF}$$

Thus given that MTBF is 11hrs,

$$R(1) = e^{-1/11}.$$

The probability that the system is not operational 1 hour after the start is then

$$1 - R(1).$$

The constant 'e' is the base of natural log and has a value of 2.718.

可靠性模型——增长模型

错误编号 #	距上次故障时间（小时）	故障强度
1	4	.25
2	6	.166
3	8	.125
4	5	.20
5	6	.166
6	10	.1
7	12	.083
8	16	.0625
9	18	.055
10	25	.04

在上述示例中

MTBF = (4+6+8+5+6+10+12+16+18+25)/10 = 11 hour

假设系统已正常启动，则系统在时刻 t 处于运行状态的概率为：

$$R(t) = e^{-t / MTBF}$$

因此，若平均无故障工作时间（MTBF）为 11 小时，

$$R(1) = e^{-1/11}.$$

则系统在启动后 1 小时处于非运行状态的概率为：

$$1 - R(1).$$

常数‘e’是自然对数的底数，其值为2.718。

Reliability Models: Fault Forecasting

- Fault forecasting [Mills 1972]: is achieved by injecting (seed) some faults in the program and thereafter estimating the remaining bugs based on how many seeded faults are detected.
- Assuming that the probability of detecting the seeded and non-seeded faults are the same:

$$N_d = N_s * (n_d/n_s)$$

where N_d, N_s, n_d, n_s are the total actual faults, total seeded faults, detected actual faults and detected seeded faults.

If $N_s = 20$; $n_s = 10$; and $n_d = 50$

Then $N_d = 100$.

可靠性模型：故障预测

- 故障预测 [米尔斯法 1972]：通过在程序中注入（植入）若干已知故障，再根据已检测出的植入故障数量，估算程序中尚存的未发现缺陷数量。
- 假设检测到植入故障与非植入故障的概率相同：

$$N_d = N_s * (n_d/n_s)$$

其中，N_d、N_s、n_d、n_s 分别表示实际总故障数、植入总故障数、已检测到的实际故障数和已检测到的植入故障数。

If $N_s = 20$; $n_s = 10$; and $n_d = 50$

则 $N_d = 100$。

Safety

- Safety is absence of catastrophic consequences on the users and the environment.
- Safety is an extension of reliability in the sense that it is reliability with respect to catastrophic failures.
- Analyzing safety requirements of a software system is a risk-driven process aimed at understanding the risks faced by the system as a consequence of errors and specifying requirements that reduce or alleviate these risks.
- This process typically results in a set of “excluding” requirements that define states and conditions that must not arise.
- FMEA (Failure Modes and Effects Analysis) and FTA (Fault Tree Analysis) are inductive and deductive techniques, respectively, commonly employed for analyzing safety critical systems.

安全性

- 安全性是指用户及环境不会遭受灾难性后果。
- 安全性是可靠性的延伸，即针对灾难性故障的可靠性。
- 分析软件系统的安全性需求是一种以风险为导向的过程，旨在理解系统因错误而面临的风险，并制定可降低或缓解这些风险的需求。
- 该过程通常会生成一组“禁止性”需求，用以定义不得出现的状态和条件。
- FMEA（失效模式与影响分析）和FTA（故障树分析）分别是常用于分析安全关键系统的归纳法与演绎法技术。

FMEA (Insulin Pump)

Identified hazard	Hazard probability	Hazard severity	Estimated risk	Acceptability
1. Insulin overdose	Medium	High	High	Intolerable
2. Insulin underdose	Medium	Low	Low	Acceptable
3. Power failure	High	Low	Low	Acceptable
4. Machine incorrectly fitted	High	High	High	Intolerable
5. Machine breaks in patient	Low	High	Medium	ALARP
6. Machine causes infection	Medium	Medium	Medium	ALARP
7. Electrical interference	Low	High	Medium	ALARP
8. Allergic reaction	Low	Low	Low	Acceptable

FMEA（胰岛素泵）

已识别的危害	危害 发生概率	危害 严重程度	估算值 风险	可接受性
1. 胰岛素过量	中等	High	High	不可耐受
2. 胰岛素剂量不足	中等	Low	Low	可接受
3. 断电	High	Low	Low	可接受
4. 设备安装不正确	High	High	High	无法容忍
5. 机器导致患者故障	Low	High	中等	ALARP
6. 机器引发感染	中等	中等	中等	ALARP
7. 电气干扰	Low	High	中等	ALARP
8. 所有过敏反应	Low	Low	Low	可接受

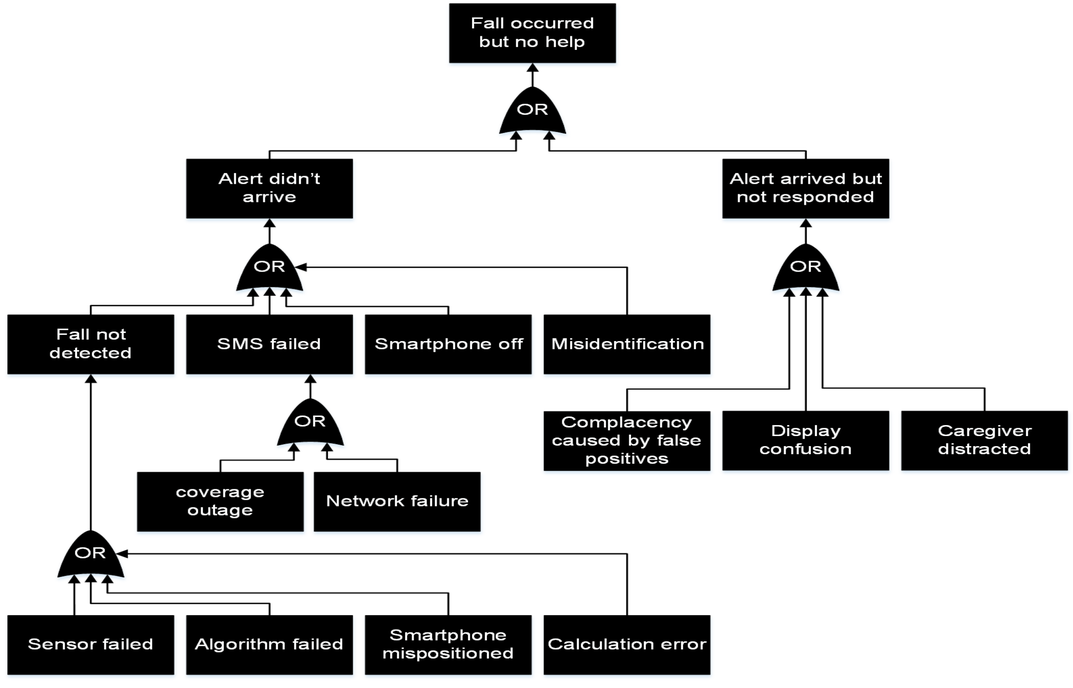
FMEA (Fall Detection & Emergency Alert)

Process Step	Failure Modes	Causes	Effects	O	S	
User fell but no help arrived	Misidentification	Emergency contact not correctly entered	Long term harm to the user	Low	High	H
		Wrong emergency contact entered		Low	High	H
		User ID not entered		Low	High	H
		Wrong User ID entered		Low	High	H
		Duplicate User IDs		Low	High	H
	Hardware, Software or Network Breakdown	Sensor data processing error		Medium	High	M
		Algorithmic error		Medium	High	M
		Wireless coverage unavailable		Medium	High	L
		SMS software, protocol or network infrastructure failed		Low	High	L
		Network interface card failed		Low	High	L
		Sensor malfunction		Low	High	L
		Battery outage		High	High	H
		Smartphone crashed		Medium	High	L
	Usability	Smartphone wrongly placed or oriented	Long term harm to the user	High	High	M
		User forgot to carry smartphone		High	High	L
		Onboard SMS app confused the care giver		Medium	Medium	M
		Unclear User Identity		Medium	Medium	M
	Alert Ignored	Complacency resulting from oversensitivity causing false positives across the system or few select smartphones	Likely harm to the user	Medium	Medium	M

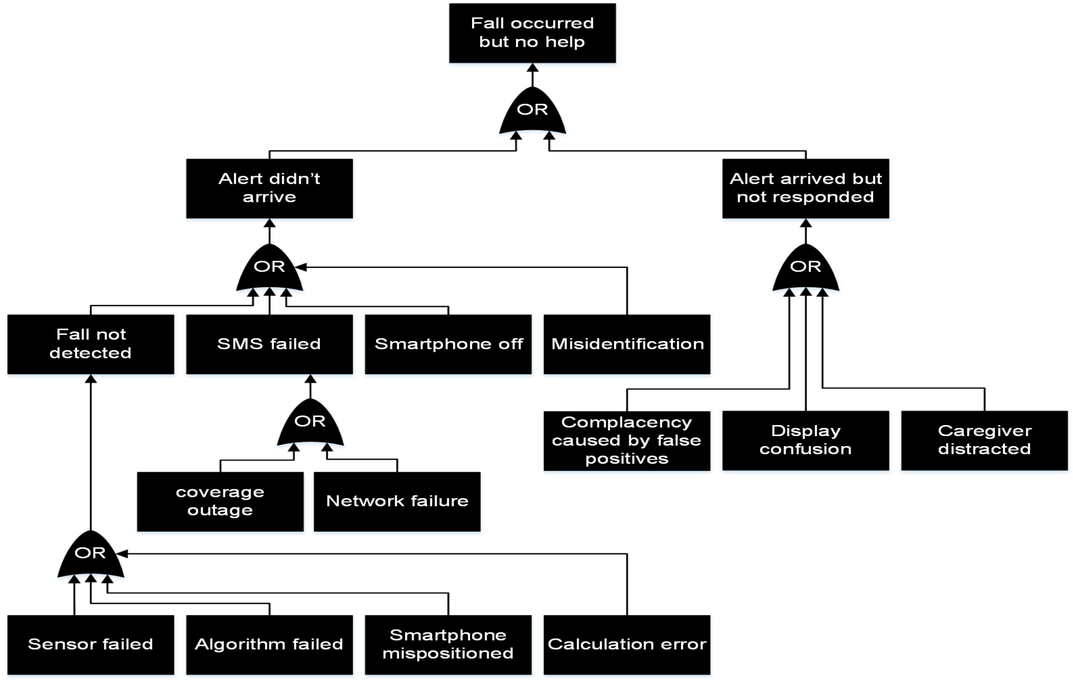
FMEA（秋季防护与应急警报rt)

流程步骤	失效Modes	原因	影响	O	S	
User fell but no help 已到达	误识别	Em 紧急联系人信息未正确录入 录入了错误的紧急联系人；用户ID未录入；录入了错误的用户ID；用户ID重复	长期harm to该用户	低 低 低 低 低	High High High High High	H H H H H
	硬件，软件or网络故障分析	传感器数据处理错误 算法 IC 错误 无线覆盖不可用（SM） 软件 、协议或网络 基础设施故障 网络接口卡故障 传感器 功能异常 电池断电 Sm 智能手机崩溃	长期harm to用户	中中中低 低低高中	高高 高高 高高 高高	M M L L L L H L
	可用性	Sm 艺术手机放置位置错误或方向错误 用户忘记携带智能手机 艺术手机 内置安全管理模块 S应用使照护者感到困惑 用户身份不明确	长期harm to该用户	高高中中	高高中中	M L M M
	警告已忽略	Com 自满 导致 from 过度敏感 导致 假 阳性结果 全系统 the 系统 or few 选择 sm 艺术手机	很可能harm to用户	中等	中等	M

FTA


Fall occurred but no help
OR
Alert didn't arrive
Alert arrived but not responded
OR
OR
Fall not detected
SMS failed
Smartphone off
Misidentification
OR
Complacency caused by false positives
Display confusion
Caregiver distracted
coverage outage
Network failure
OR
Sensor failed
Algorithm failed
Smartphone mispositioned
Calculation error

FTA


Fall occurred but no help
OR
Alert didn't arrive
Alert arrived but not responded
OR
OR
Fall not detected
SMS failed
Smartphone off
Misidentification
OR
Complacency caused by false positives
Display confusion
Caregiver distracted
coverage outage
Network failure
OR
Sensor failed
Algorithm failed
Smartphone mispositioned
Calculation error

Availability

- Availability is the probability that a system is operational at any time.
- If reliability improves by eliminating faults or masking them effectively then availability improves by introducing redundancy and incorporating fault tolerance.
- Availability = MTBF / (MTTR + MTBF), where MTTR is the Mean Time To Recover and MTBF is Mean Time Between Failures.
- A system with N components connected in tandem will have overall availability of Π (A_n), where A_n, given by MTTR / (MTTR + MTBF), is the availability of the individual component. On the other hand, a system with M components, all connected in parallel, will render an overall availability of [1 - Π (1 - A_n)].
- Given that a component's availability is less than or equal to 1, these statistical availability models obviate that the availability of the overall system will improve if more redundancy is introduced i.e. more components are added in parallel and will deteriorate, if on the other hand, the number of components connected in tandem increases.

可用性

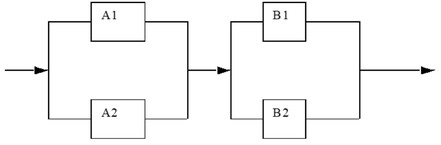
- 可用性是指系统在任意时刻处于正常运行状态的概率。
- 若通过消除故障或有效屏蔽故障来提升可靠性，则可通过引入冗余设计并融入容错机制来提高可用性。
- 可用性 = MTBF / (MTTR + MTBF)，其中 MTTR 表示平均恢复时间，MTBF 表示平均无故障工作时间。

- 由 N 个组件串联组成的系统，其整体可用性为 Π (A_n)，其中单个组件的可用性 A_n 由 MTTR / (MTTR + MTBF) 给出；而由 M 个组件并联组成的系统，其整体可用性则为 [1 - Π (1 - A_n)]。
- 由于单个组件的可用性恒小于或等于 1，因此上述统计可用性模型表明：若增加并联组件数量（即引入更多冗余），则整个系统的可用性将得以提升；反之，若串联组件数量增加，则整个系统的可用性将下降。

Consider a distributed systems architecture depicted below. Any incoming request is equally likely to traverse any of the four possible paths i.e. A1B1, A1B2, A2B1 and A2B2. The availability of the overall system in which A1 & A2 as well as B1 and B2 are pairs of replicated components introduced to induce redundancy is therefore [1 – ((1-A1)(1-A2))] x [1 – ((1-B1)(1-B2))]. Given that A1 = A2 = B1 = B2 = 0.90, the overall availability is 0.98.

[1 – ((1-0.9)x(1-0.9))] x [1 – (1-0.9)x(1-0.9)] = .99 x .99 = .98

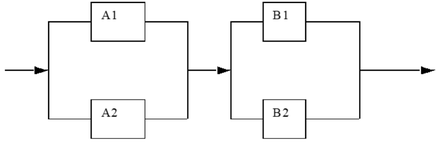
0.9 x 0.9 = .81



考虑如下所示的分布式系统架构。任何传入请求均等可能地经过以下四条路径之一，即 A1B1、A1B2、A2B1 和 A2B2。其中，A1 与 A2、B1 与 B2 分别为为实现冗余而引入的冗余组件对，因此整个系统的可用性为 [1 – ((1-A1)(1-A2))] ×[1 – ((1-B1)(1-B2))]。已知 A1 = A2 = B1 = B2 = 0.90，整个系统的可用性为 0.98。

[1 – ((1-0.9)x(1-0.9))] x [1 – (1-0.9)x(1-0.9)] = .99 x .99 = .98

0.9 x 0.9 = .81



Availability Testing

- Availability is verified and validated via stress testing that involves creating conditions e.g. resource starvation or forced shutdown of components causing the system to fail to determine how graceful, fast and correct the recovery from the failure was

可用性测试

- 可用性通过压力测试进行验证与确认，该测试通过人为制造资源耗尽或强制关闭组件等异常条件，使系统发生故障，从而评估系统从故障中恢复的优雅程度、恢复速度以及恢复结果的正确性。