华东师范大学软件学院课程作业

课程名称：软件质量分析	年级：2023 级本科	姓名：张梓卫
作业主题：总结软件可靠性定义	学号：10235101526	作业日期：2024/10/10
指导老师：陈仪香	组号：	

目录

一 系统级软件 FMEA 的定义

1 定义

系统级软件 FMEA（Failure Modes and Effects Analysis）通过识别软件失效模式 *，分析造成的后果，研究分析各种失效模式产生的原因，寻找消除和减少其有害后果的方法，以尽早发现潜在的问题，并采取相应的措施，从而提高软件的可靠性和安全性。用于预防软件系统中的潜在问题，确保系统的可靠性和安全性。常用于嵌入式领域。

FMEA 的目标是识别可能的故障源头，评估这些故障的影响，并在故障实际发生前采取预防或减轻措施。FMEA 通常应用于软件开发的早期阶段，以便在设计阶段就能发现潜在的问题。

* **软件失效模式：**软件失效（软件故障引起的，泛指程序运行中丧失全部或部分功能）发生的不同方式。分析它是软件 FMEA 的基础。

二 FMEA 的步骤

1 系统定义以确定分析级别和分析对象

功能：确定分析重点，说明系统主要和次要功能，

优势：根据软件系统的功能、结构特征等确定系统的分析级别，在高层较易全面分析，在低层的分析更深入。

注意：若无法全面分析，可在分析前确定分析重点，识别对系统功能和安全性影响较大的危险事件作为 FMEA 分析的重点。

2 确定分析分析对象的失效模式

功能：针对每个分析对象，参考失效模式。

举例：响应超时、输出错误。

3 分析失效模式的可能原因

功能：尽可能全面分析失效原因，为制定改进措施提供依据。

一般失效原因种，可以分为多种具体失效原因。

归类如下：

- 逻辑遗漏或执行错误
  - 遗忘细节或步骤
  - 逻辑重复
  - 忽略极限条件
  - 不必要的函数
  - 需求的错误表达
  - 未进行条件测试
  - 检查错误变量
  - 循环错误
- 算法的编码错误
  - 等式不完整或不正确
  - 丢失运算结果
  - 操作数错误
  - 括号使用错误
  - 精度损失
  - 含入和舍去错误
  - 混合类型
  - 标记习惯不正确
- 软硬件接口失效
  - 中断句柄错误
  - I/O 时序错误
  - 时序错误导致数据丢失
  - 子函数调用不当
  - 子函数调用位置错误
  - 调用不存在的子函数
  - 子函数不一致
- 数据错误或丢失
  - 传感器数据错误或丢失
  - 操作数据错误或丢失
  - 嵌入到表中的数据错误或丢失
  - 外部数据错误或丢失
  - 输出数据错误或丢失
  - 输入数据错误或丢失
- 数据操作错误
  - 数据初始化错误
  - 数据存取错误
  - 数据打包解包错误
  - 标志或索引设置不当
  - 变量参考错误数据
  - 数据越界
  - 变量缩放比率或单位不正确
  - 变量维度不正确
  - 变量类型错误
  - 变量下标错误
  - 数据范围不对

4 分析失效影响及其严重性

功能：分析每个失效模式对局部、高一层，直至整个系统的影响，以及失效影响的严重性，以识别软件失效所造成后果的严重程度，以便按照优先级为不同的情况制定改进措施。

5 制定改进措施

功能：根据上述分析得到的失效原因和影响的严重性，确定需要采取的改进措施。

改进方式：

5 .1 软件层面

修改软件需求、设计或编码中的缺陷，增加软件的防护措施。

5 .2 硬件层面

增加硬件防护措施。

5 .3 FMEA 表格

进行软件 FMEA 时，应填写 FMEA 表。该表应能完整地体现分析的目的和取得成果。

编号	单元	功能	失效模式	可能的失效原因	失效影响			严酷度	改进措施
					局部影响	高一层次影响	最终影响		
1.1	输出	输出数据提交用户显示	数值高于正常范围	逻辑问题 计算问题 数据操作问题	N/A	无	任务降级	III (中等)	.....
1.2			数值低于正常范围	逻辑问题 计算问题 数据操作问题	N/A	无	任务降级	III （中等）	.....
1.3			输出数据没有显示	逻辑问题 计算问题 数据操作问题	N/A	无	任务中止	II或I 致命或灾难	.....
1.4			.....	.....	.....	.....	.....	.....	.....
1.X									

三 选做题作业

构建 Jelinski-Moranda 模型

收集并处理数据

首先，处理给定的失效时间数据：

```python
# 给定的失效时间数据（单位时间）
failure_times = [15, 13, 9, 10, 21, 23, 18, 22, 17, 16]

# 计算累计失效时间和总测试时间
cumulative_times = []
total_time = 0
for t in failure_times:
    total_time += t
    cumulative_times.append(total_time)

S = total_time          # 总的测试时间
n = len(failure_times)  # 已发生的失效次数
```

估计参数 N（初始故障总数）

Jelinski-Moranda 模型的关键是估计软件中的初始故障总数 N。需要解以下方程：

$$\sum_{k=1}^{n} \frac{1}{N-(k-1)} = \frac{n}{S}$$

由于无法直接求解，我们使用数值方法（如二分法）来估计 N：

```python
def equation(N, n, S):
    left_sum = sum([1 / (N - (k - 1)) for k in range(1, n + 1)])
    right_sum = n / S
    return left_sum - right_sum

from scipy.optimize import root_scalar

def func(N):
    return equation(N, n, S)

# 设置求解区间，N > n
sol = root_scalar(func, bracket=[n + 1, n + 1000], method='bisect')
N_estimated = sol.root
print(f"估计的 N 值为: {N_estimated}")
```

估计参数 ϕ

计算参数 ϕ：

$$\phi = \frac{n}{\sum_{k=1}^{n} (N-k+1) \cdot T_k}$$

其中，T_k 为第 k 次失效的间隔时间。

```python
# 计算分母
denominator = sum([(N_estimated - k + 1) * failure_times[k - 1] for k in range(1, n + 1)])

# 计算
phi = n / denominator
print(f"估计的    值为: {phi}")
```

计算第 11 次失效前的失效率 λ

$$\lambda_{n+1} = \phi \cdot (N-n)$$

```python
lambda_11 = phi * (N_estimated - n)
print(f"第 11 次失效前的失效率    为: {lambda_11}")
```

计算平均失效前时间（MTTF）

$$\text{MTTF} = \frac{1}{\lambda_{n+1}}$$

```python
MTTF = 1 / lambda_11
print(f"平均失效前时间（MTTF）为: {MTTF}")
```

计算在 MTTF 时的软件可靠度 $R(t)$

$$R(t) = e^{-\lambda_{n+1} \cdot t}$$

令 $t = \text{MTTF}$，则：

```
import math

t = MTTF
R_t = math.exp(-lambda_11 * t)
print(f"在时间 t = MTTF 时的软件可靠度 R(t) 为: {R_t}")
```

计算不可靠度 $U(t)$

$$U(t) = 1 - R(t)$$

```
U_t = 1 - R_t
print(f"在时间 t = MTTF 时的软件不可靠度 U(t) 为: {U_t}")
```

计算失效密度 $f(t)$

$$f(t) = \lambda_{n+1} \cdot e^{-\lambda_{n+1} \cdot t}$$

```
f_t = lambda_11 * math.exp(-lambda_11 * t)
print(f"在时间 t = MTTF 时的失效密度 f(t) 为: {f_t}")
```

完整代码

```
failure_times = [15, 13, 9, 10, 21, 23, 18, 22, 17, 16]
cumulative_times = []
total_time = 0
for t in failure_times:
    total_time += t
    cumulative_times.append(total_time)

S = total_time
n = len(failure_times)

def equation(N, n, S):
    left_sum = sum([1 / (N - (k - 1)) for k in range(1, n + 1)])
    right_sum = n / S
    return left_sum - right_sum

from scipy.optimize import root_scalar

def func(N):
    return equation(N, n, S)

sol = root_scalar(func, bracket=[n + 1, n + 1000], method='bisect')
N_estimated = sol.root
print(f"估计的 N 值为: {N_estimated}")

denominator = sum([(N_estimated - k + 1) * failure_times[k - 1] for k in range(1, n + 1)])
phi = n / denominator
print(f"估计的    值为: {phi}")

```

```python
lambda_11 = phi * (N_estimated - n)
print(f"第 11 次失效前的失效率    为: {lambda_11}")

MTTF = 1 / lambda_11
print(f"平均失效前时间（MTTF）为: {MTTF}")

import math
t = MTTF
R_t = math.exp(-lambda_11 * t)
print(f"在时间 t = MTTF 时的软件可靠度 R(t) 为: {R_t}")

U_t = 1 - R_t
print(f"在时间 t = MTTF 时的软件不可靠度 U(t) 为: {U_t}")

f_t = lambda_11 * math.exp(-lambda_11 * t)
print(f"在时间 t = MTTF 时的失效密度 f(t) 为: {f_t}")
```

运行结果

```
估计的 N 值为: 169.00000000000006
估计的    值为: 0.0003717857830703583
第 11 次失效前的失效率    为: 0.05918594751117227
平均失效前时间（MTTF）为: 16.889800000000006
在时间 t = MTTF 时的软件可靠度 R(t) 为: 0.3678794411714422
在时间 t = MTTF 时的软件不可靠度 U(t) 为: 0.6321205588285578
在时间 t = MTTF 时的失效密度 f(t) 为: 0.02186010370865581
```

结论

- **软件可靠度** $R(t)$：在第 11 次失效发生的平均时间（MTTF）时，软件可靠度约为 **0.3679**。
- **不可靠度** $U(t)$：在同一时间点，软件不可靠度约为 **0.6321**。
- **失效密度** $f(t)$：在 MTTF 时的失效密度约为 **0.02186**。
- **平均失效前时间（MTTF）**：约为 **16.89** 时间单位。

四 其余笔记

1 系统级软件 FMEA

在软件开发阶段的早期，用于发现软件需求或软件体系结构等存在的缺陷。主要分析对象：**软件需求分析阶段的软件功能或设计阶段的软件部件**

失效模式（9 种）：

- （1）操作系统挂起
- （2）程序挂起
- （3）程序失败：程序不能启动；程序运行不能终止；程序不能退出
- （4）输入问题：错误输入被接受；正确输入被拒绝；描述不正确或遗漏；参数不正确或遗漏
- （5）输出问题：错误的格式；不正确的结果或数据；不完全或遗漏；拼写问题、语法问题
- （6）未达到要求的性能：错误的格式；不正确的结果或数据；不完全或遗漏；拼写问题、语法问题
- （7）发现整个产品失败

- （8）系统错误信
- （9）其他：程序运行改变了系统配置参数；程序运行改变了其他程序的运行结果数据

软件失效模式分为两类：通用失效模式和详细失效模式。

详细失效模式是对通用失效模式的细化，又分为五子类：输入失效、输出失效、程序失效、未满足功能及性能要求失效、其他类型。

软件失效模式示例

- **软件通用失效模式**
  - 运行时不符合要求
  - 输入不符合要求
  - 输出不符合要求
- **软件详细失效模式**
  - 输入失效
    - 未收到输入
    - 收到错误输入
    - 收到数据轻微超差
    - 收到数据中度超差
    - 收到数据严重超差
    - 收到参数不完全或遗漏
    - 其他
  - 输出失效
    - 输出结果错误（如输出项缺损或多余等）
    - 输出数据精度轻微超差
    - 输出数据精度中度超差
    - 输出数据精度严重超差
    - 输出参数不完全或遗漏
    - 输出格式错误
    - 输出打印字符不符合要求
    - 输出拼写/语法错误
    - 其他
  - 程序失效
    - 程序无法启动
    - 程序运行中断
    - 程序无法终止
    - 程序无法退出
    - 程序陷入死循环
    - 程序运行影响其他单元或环境
    - 程序轻微超时
    - 程序明显超时
    - 程序严重超时
    - 其他
  - 其他失效
    - 程序改变配置无法启动
    - 程序改变其他程序数据
    - 操作系统错误
    - 硬件错误
    - 整个系统错误
    - 人为操作错误
    - 接口失效
    - I/O 定时不准致数据丢失
    - 维护不合理
    - 其他

五　实施注意事项

1　基础内容

1. 软件 FMEA 的应用重点在软件开发过程的早期，找出可能存在的与功能和性能相关的缺陷，以尽早完善需求分析与概要设计。2. 根据分析阶段和级别不同分为系统级和详细级。

2　主要内容

1. 软件失效模式是软件 FMEA 的基础。失效模式的分析是否全面合理决定了软件 FMEA 的分析效果，是整个分析过程中最为关键的一步。

2. 软件失效原因是由于软件缺陷在运行时被触发而产生的。对软件 FMEA 失效原因分析就是要找出潜在的软件缺陷。

3. 失效影响既可对每一个软件失效模式造成的“局部、高一层次和最终”三级影响进行分析，又可对“具备、高一层次”的影响进行分析，或直接对“局部、最终”影响进行分析。

4. 分析对象既可以是软件功能模块、软件部件，又可以是底层的软件单元

六 参考文献

- https://arxiv.org/pdf/1108.5185
- J-M 模型的计算机实现：[https://www.wdfxw.net/doc20562618.htm]