第四讲 软件可靠性

李宁

2020.4

对可靠性的需求

- 用软件系统规模越做越大越复杂，其可靠性越来越难保证。应用本身对系统运行的可靠性要求越来越高：
  - 在一些关键的应用领域，如航空、航天等，其可靠性要求尤为重要，在银行等服务性行业，其软件系统的可靠性也直接关系到自身的声誉和生存发展竞争能力。
- 特别是软件可靠性比硬件可靠性更难保证，会严重影响整个系统的可靠性。

主要内容

1. 软件可靠性的定义
2. 可靠性模型及其评价标准
3. 软件可靠性测试和评估
4. 软件可靠性研究的主要问题

1. 软件可靠性的定义

- 1983年美国IEEE计算机学会对“软件可靠性”一词正式作出了如下的定义：
  - 在规定的条件下，在规定的时间内，软件不引起系统失效的概率，该概率是系统输入和系统使用的函数，也是软件中存在的缺陷的函数。
  - 在规定的时间周期内，在所述条件下程序执行所要求的功能的能力。

- 目标：规定的任务和功能(不引起失效)
- 系统输入和系统使用(规定的条件)
- 时间周期(规定的时间)

1. 软件可靠性的定义

➢ 软件使用剖面

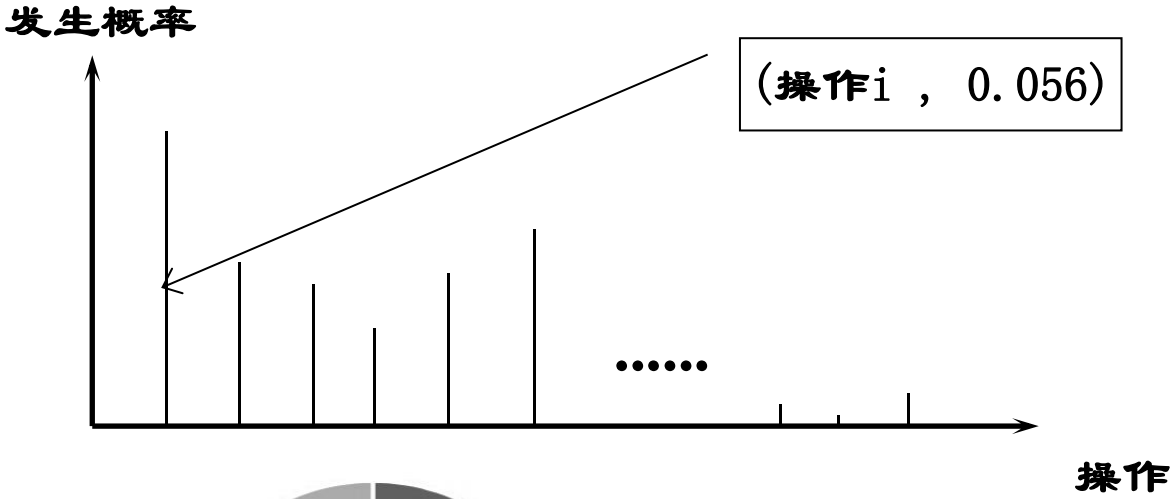
- ❖ 使用剖面是对软件使用的刻画，包括使用者和使用方式。
- ❖ 使用者包括人、硬件、外部软件等。
- ❖ 使用方式包括使用的操作、频度等。
- ❖ 操作环境可以用操作剖面或操作模型来定义。

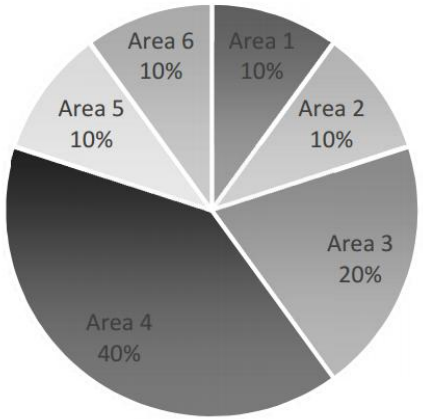
操作剖面的主要形态:

1. 用彼此互斥的操作集合的出现概率描述操作剖面
2. 使用马尔科夫链模型指定软件系统状态以及状态之间的转移作为操作剖面。

1. 软件可靠性的定义

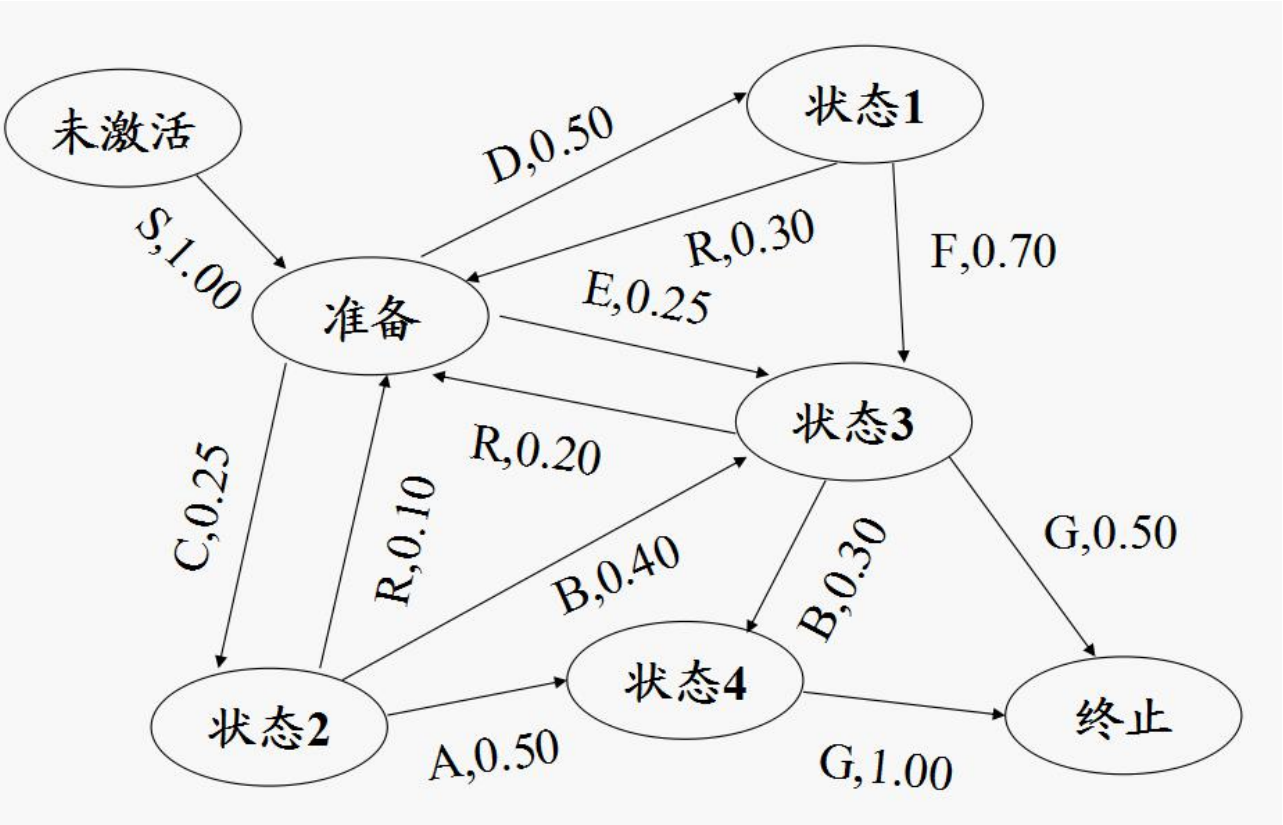
- 彼此互斥的操作集合的出现概率





1. 软件可靠性的定义

- **软件使用剖面** (usage profile)
  - 马尔科夫链模型的操作剖面模型:

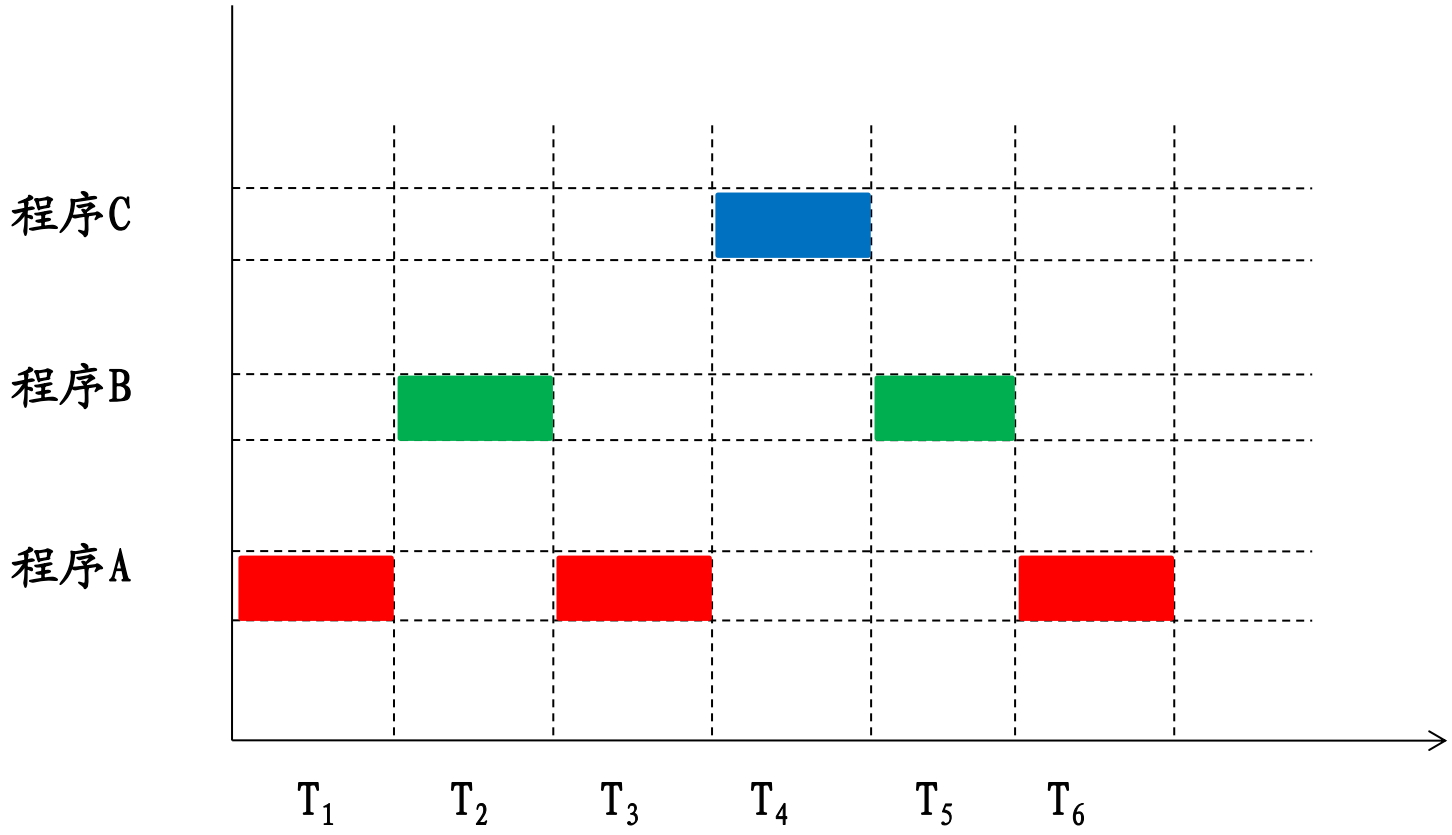


1. 软件可靠性的定义

- 规定的时间 – 时间表述
  - 失效的时间点
  - 失效间的时间间隔
  - 到达给定时间经历的失效数
  - 在给定时间间隔内发生的失效数

- 规定的时间 – 时间选取
  - 执行时间
  - 日历时间
  - 时钟时间

1. 软件可靠性的定义



程序A(执行时间)=$T_1+T_3+T_6$

程序A(时钟时间)=$T_1+T_2+T_3+T_4+T_5+T_6$

1. 软件可靠性工程

- 定义
  - 为获得软件可靠性而进行的一系列开发、维护软件产品的活动
- 基础
  - 软件工程化开发
- 特点
  - 软件过程和产品的定量控制和管理

1. 软件可靠性工程

- **软件不可靠的原因：**
  1. 不完善的需求定义
  2. 客户与开发人员缺乏交流
  3. 偏离软件需求
  4. 逻辑设计错误
  5. 编码错误
  6. 编码与文档不一致
  7. 缺少测试过程
  8. 接口定义错误
  9. 不受控的变更

1. 软件可靠性工程

- 目标
  - 分析、管理并提高软件产品的可靠性
  - 在时间、成本和质量之间获得平衡
  - 确定软件质量达到发行要求的时间，将发行风险降至最低
  - 避免过度测试对市场开拓的影响

主要内容

1. 软件可靠性的定义
2. 可靠性模型及其评价标准
3. 软件可靠性测试和评估
4. 软件可靠性研究的主要问题

2. 软件可靠性模型

□ 软件可靠性模型

- 为了对软件可靠性评估，除了进行软件测试之外，我们还需要借助软件可靠性模型的帮助。
- 软件可靠性模型（Software Reliability Model）是指为预计或估算软件的可靠性所建立的可靠性数学模型。
- 建立可靠性模型可以将复杂系统的可靠性逐级分解为简单系统的可靠性，以便于定量预计、分配、估算和评价复杂系统的可靠性。

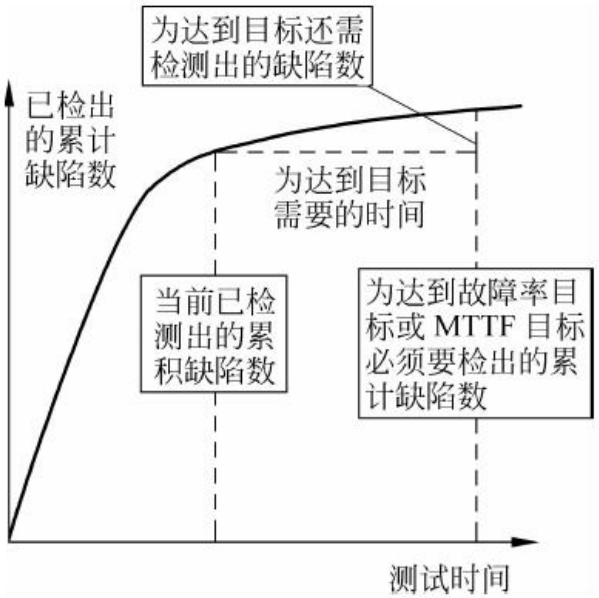
2. 软件可靠性模型

基本方法

- 用过去的失效数据建立可靠性模型，然后用所建立起来的模型估计现在和预测将来的软件可靠性。
- 该方法使用的先验条件是给定过去某个时期内的软件失效次数或软件的失效时间间隔。

分类：

- Input Domain
- Time Domain



2. 软件可靠性模型

- IDRM: Input domain reliability model.
  - Nelson Model:
    - Running for a sample of n inputs.
    - Randomly selected from set E:

$$E = \{E_i : i = 1, 2, \ldots, N\}$$

    - Sampling probability vector:

$$\{P_i : i = 1, 2, \ldots, N\}$$

    - $\{P_i\}$: Operational profile.
    - Number of failures: f.
    - Estimated reliability:

$$R = 1 - r = 1 - \frac{f}{n} = \frac{n - f}{n}$$

    - Failure rate: r.

2. 软件可靠性模型

- IDRM: Input domain reliability model.

  - Nelson model for web applications
    - daily error rates: Table 22.2 (p.377)

Daily Error Rate	min	max	mean	std dev	rse
errors /hits	0.0287	0.0466	0.0379	0.00480	0.126
errors /day	501	1582	1101	312	0.283

2. 软件可靠性模型

Time Domain Measures and Models

- Mean time to failure (MTTF)
- Mean time to repair (MTTR)
- Mean time between failures (MTBF)

  MTBF = MTTF + MTTR

- Reliability R = MTTF/(1+MTTF)
- Availability A = MTBF/(1+MTBF)
- Maintainability M = 1/(1+MTTR)

2. 软件可靠性模型

Time Domain Measures and Models

- **Example: failure time (day)**
  - **SF1: 180, 675, 315, 212, 278, 503, 431**
  - **SF2: 477, 1048, 685, 396**
  - **SF3: 894, 1422**

  - **$MTTF_{SF1}$ = 2594/7 = 370.57 R=370.57/(1+370.57)=0.997**
  - **$MTTF_{SF2}$ = 2606/4 = 651.5 R=651.5 /(1+651.5) =0.998**
  - **$MTTF_{SF3}$ = 2316/2 = 1158 R=1158/(1+1158)=0.999**

2. 软件可靠性模型

Time Domain Models

SRGM: Software reliability growth model

Label	Reference	Model
JM	[28]	Jelinski-Moranda de-eutrophication model
Sho	[51]	Shooman model
Geo	[36]	Moranda's geometric de-eutrophication model
GO	[25]	Goel-Okumoto NHPP model
S	[74]	Yamada-Ohba-Osaki S-shaped NHPP model
GP	[46]	Schafer *et al.* generalized Poisson model
SW	[47]	Schick-Wolverton model
BMB	[13]	Brooks-Motley model, binomial variation
BMP	[13]	Brooks-Motley model, Poisson variation
Musa	[37]	Musa's basic execution time model
MO	[41]	Musa-Okumoto logarithmic Poisson execution time model
Sch	[48]	Schneidewind model
LV	[32]	Littlewood-Varrel Bayesian model

Table 2: Labels and references for some commonly used SRGMs

2. 软件可靠性模型

- SRGM: Software reliability growth model

NHPP（non-homogeneous Poisson process，非齐次泊松过程）

Basic assumptions of NHPP models

1. A Software system is subject to failure during execution caused by faults remaining in the system.
2. The number of faults detected at any time is proportional to the remaining number of faults in the software.
3. Failure rate of the software is equally affected by faults remaining in the software.
4. On a failure, repair efforts starts and fault causing failure is removed with certainty.
5. All faults are mutually independent from a failure detection point of view.
6. The proportionality of failure occurrence/fault isolation/fault removal is constant.
7. Corresponding to the fault detection/removal phenomenon at the manufacturer/user end, there exists an equivalent fault detection/fault removal at the user/manufacturer end.
8. The fault detection/removal phenomenon is modeled by NHPP.

2. 软件可靠性模型

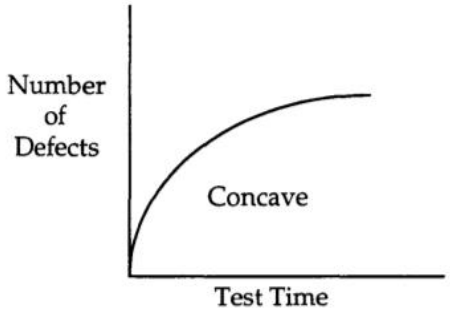
- SRGM: Software reliability growth model

NHPP（non-homogeneous Poisson process，非齐次泊松过程)）

- Goel-Okumoto model

$$m(t) = N(1 - e^{-bt})$$

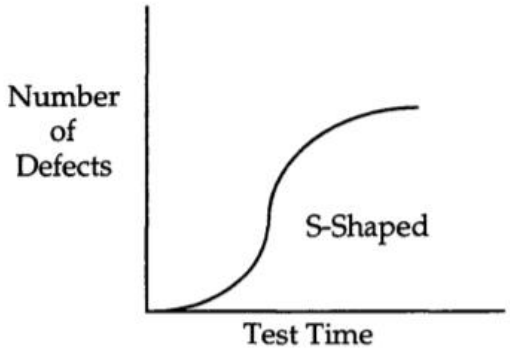
  - N: estimated # of defects
  - b: model curvature



- S-shaped model:

$$m(t) = N(1 - (1 + bt)e^{-bt})$$

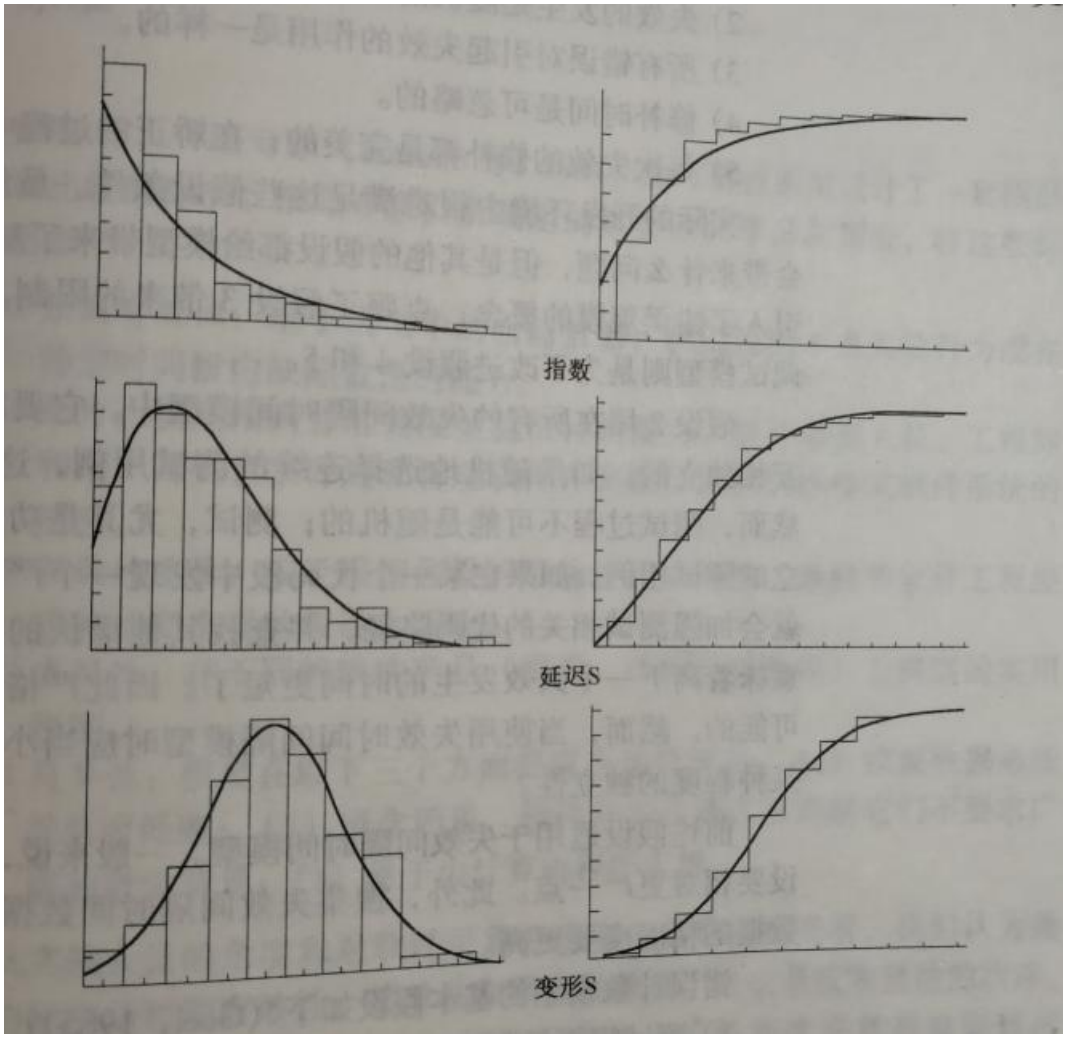
  - allow for slow start
  - may be more descriptive



NHPP模型假设:
1. 失效事件随机发生，且相互独立。 2. 有限个失效。
3. 每次只修正一个错误，当软件故障出现时，引发故障的错误被立即排除，并不会引入新的错误。

2. 软件可靠性模型

- SRGM: Software reliability growth model



峰值：测试开始阶段
（考虑缺陷测试过程）

峰值：延迟的峰值
（考虑失效观测和报告之间的缺陷隔离过程）

峰值：更迟、更尖的峰值
（被检测的缺陷之间具有相互依赖性）

2. 软件可靠性模型

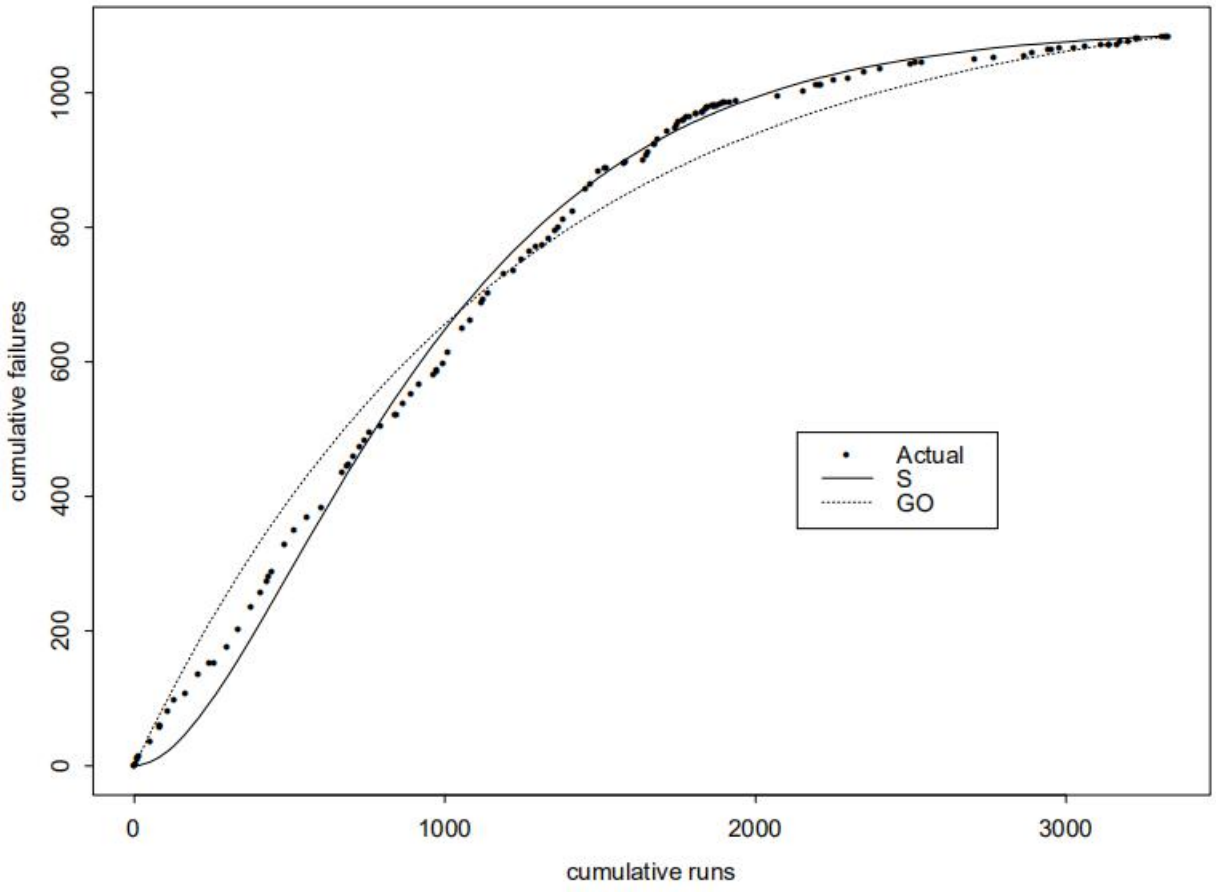
- SRGM: Software reliability growth model

❖ 模型假设：假设越合理，模型效果就越好。

例：J-M模型假设：

1. 在测试开始时，有N个未知的软件错误。
2. 失效的发生是随机的（失效间隔时间互相对立）。
3. 所有错误对引起失效的作用相同。
4. 修补时间可以忽略。(G-O模型改进)
5. 每次修补都是完美，不会引入新的错误。(G-O模型改进)

2. 软件可靠性模型


cumulative failures
0
200
400
600
800
1000
0
1000
2000
3000
cumulative runs
Actual
S
GO

2. 软件可靠性模型

可靠性建模的过程

1. 考察数据：failure + time
2. 根据对测试过程、数据和模型假设的理解，选择一个或者多个模型进行数据拟合。
3. 估计模型的参数（可以使用最大似然法、最小二乘法等）。
4. 将参数估计值代入被模型中，得到拟合好的模型。
5. 进行拟合程度检验，并估计模型的合理性。
6. 基于拟合的模型做出可靠性预测。

实际拟合例子：R程序

2. 软件可靠性模型

- 软件可靠性模型的评估依据？

- 模型拟合性
- 模型预计有效性
- 模型误差
- 模型噪声
- ……

标准重要性：　预测有效性 > 简单性 > 假设的质量

主要内容

1. 软件可靠性的定义
2. 可靠性模型及其评价标准
3. 软件可靠性测试和评估
4. 软件可靠性研究的主要问题

3. 软件可靠性测试与评估

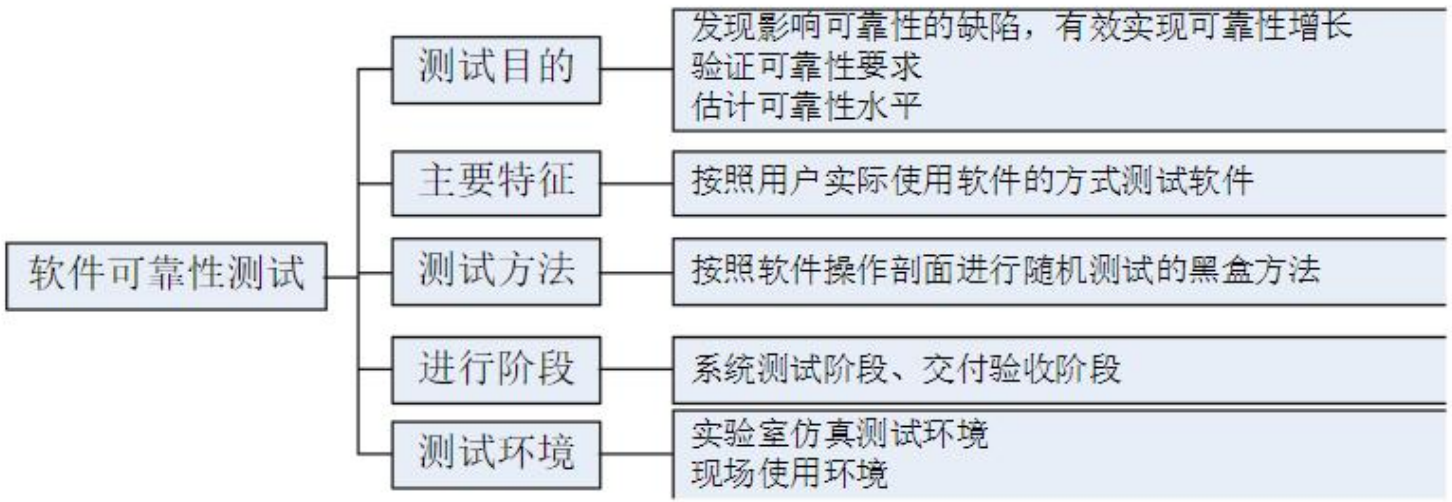
- **软件可靠性评测是主要的软件可靠性评价技术。**
  - 测试
  - 评价

  既适用于软件开发过程，也可针对最终产品。

- **用途**
  - 更快速地找出对可靠性影响最大的错误
  - 结合软件可靠性增长模型估计软件当前的可靠性，以确认是否可以终止测试和发布软件。
  - 预计软件要达到相应的可靠性水平所需要的时间和测试量，论证在给定日期提交软件可能给可靠性带来的影响。

3. 软件可靠性测试与评估

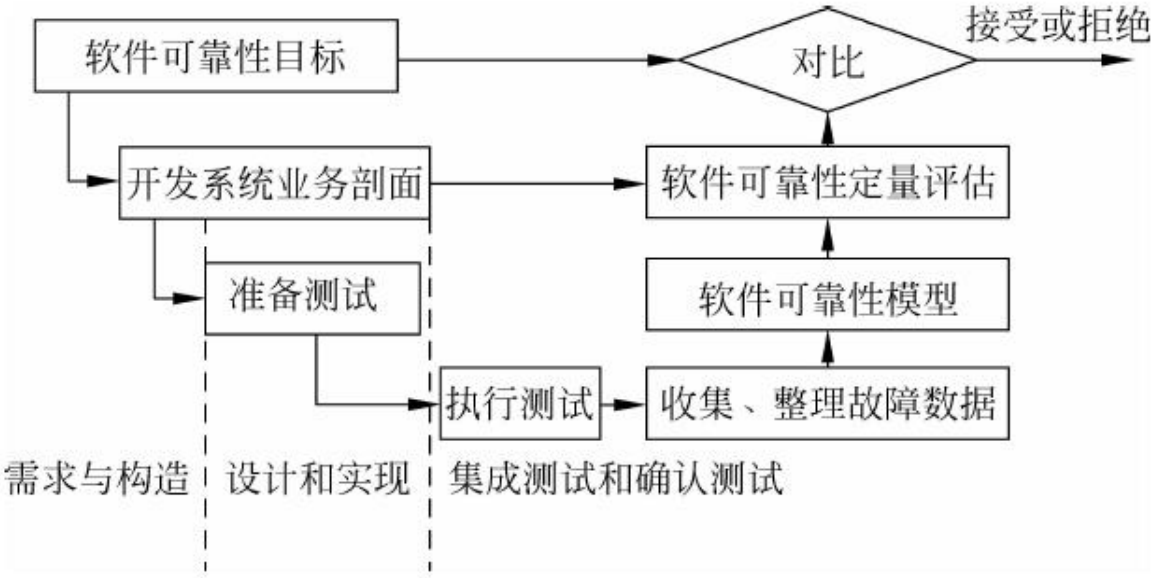
软件可靠性测试特点



3. 软件可靠性测试与评估

□ 软件可靠性测试过程实施

- 测试目的
- 测试准备和执行：运行剖面（统计知识）



3. 软件可靠性测试

Usage-Based Statistical Testing(UBST)

- *Reliability:* Probability of failure-free operation for a specific time period or a given set of input under a specific environment
  - ▷ Reliability: customer view of quality
  - ▷ Probability: statistical modeling
  - ▷ Time/input/environment: OP

- OP: Operational Profile
  - ▷ Quantitative characterization of the way a (software) system will be used.
  - ▷ Generate/execute test cases for UBST
  - ▷ Realistic reliability assessment
  - ▷ Development decisions/priorities

3. 软件可靠性测试

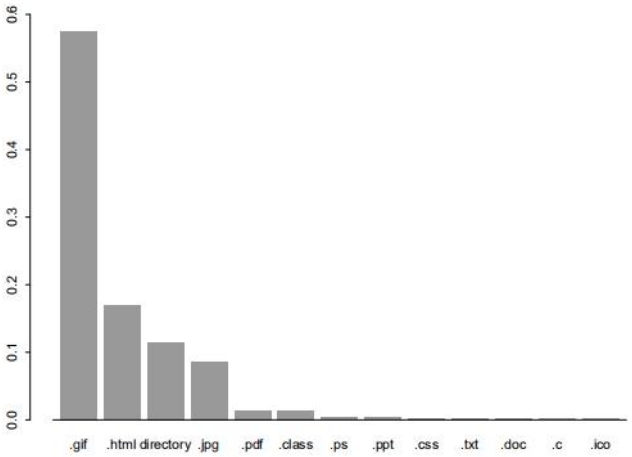
UBST General steps:

1. **Information collection.**
2. **OP construction**
3. **UBST under OP.**
4. **Analysis reliability and follow up.**

3. 软件可靠性测试

- Example: Table 8.4, p.112
  - file type usage OP for SMU/SEAS

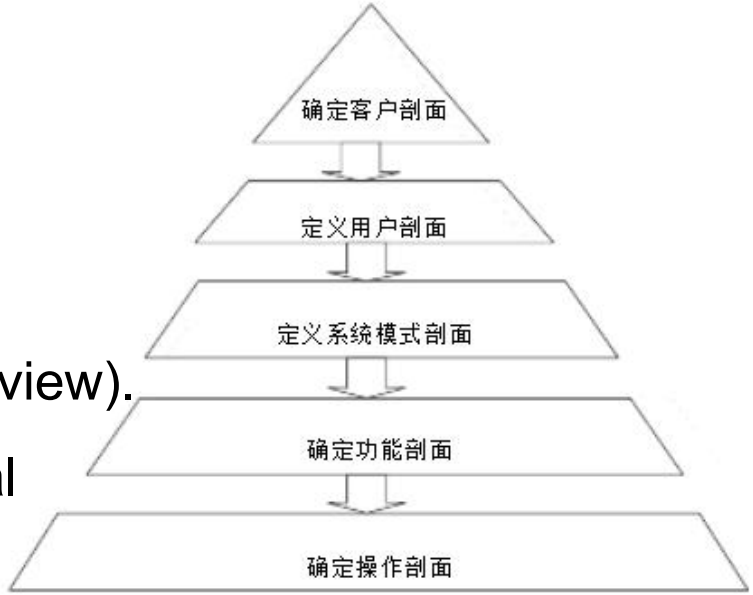
File type	Hits	% of total
.gif	438536	57.47%
.html	128869	16.89%
directory	87067	11.41%
.jpg	65876	8.63%
.pdf	10784	1.41%
.class	10055	1.32%
.ps	2737	0.36%
.ppt	2510	0.33%
.css	2008	0.26%
.txt	1597	0.21%
.doc	1567	0.21%
.c	1254	0.16%
.ico	849	0.11%
Cumulative	753709	98.78%
Total	763021	100%



Generic steps for OP Development (Musa1)

(One OP for each homogeneous group of users or operations)

1. Find the customer profile (external view).
2. Establish the user profile (external view).
3. Define the system modes (internal view).
4. Determine the functional profile (internal view).
5. Determine the operational profile (internal view).



Generic steps for OP Development (Musa1)

1. Find the customer profile (external view).

Weight assignment:

. By #customers

. By importance/marketing concerns, etc.

Customer Type	Weight
corporation	0.5
government	0.4
education	0.05
other	0.05

Generic steps for OP Development (Musa1)

2. Establish the user profile (external view).

Weighting factor assignment for user weights within customer types:

. by users (equal usage intensity)

. by usage frequency

. other factors also possible

▹ customer profile used to calculate comprehensive user profile:
0.8×0.5 (com) $+$ 0.9×0.4 (gov) $+$ 0.9×0.05 (edu) $+$ 0.7×0.05 (etc) $= 0.84$

User Type	User Profile by Customer Type					Overall User Profile
	ctype	com	gov	edu	etc	
	weight	0.5	0.4	0.05	0.05	
end user		0.8	0.9	0.9	0.7	0.84
dba		0.02	0.02	0.02	0.02	0.02
programmer		0.18	–	–	0.28	0.104
third party		–	0.08	0.08	–	0.036

Generic steps for OP Development (Musa1)

3. Define the system modes (internal view).

- System mode
  - A set of functions/operations
  - For operational behavior analysis
  - Practicality: expert for system mode
- Example modes
  - Business use mode
  - Personal use mode
  - Attendant mode
  - System administration mode
  - Maintenance mode
  - Probabilities (weighting factors)

Generic steps for OP Development (Musa1)

4. Determine the functional profile (internal view).
   - ❑ Identifying functions
   - ❑ Creating Function list
   - ❑ Determining occurrence probabilities

5. Determine the operational profile (internal view).
   - ❑ Refining functional profile into OP
   - ❑ Defining operations
     - Partitioning input space into operations
   - ❑ Obtaining occurrence probabilities

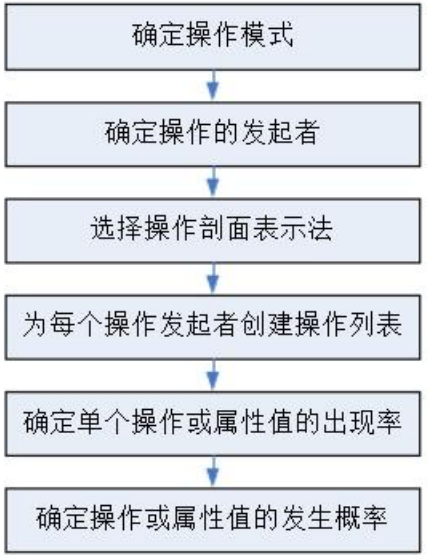
Generic steps for OP Development (Musa2)

(One OP for each operational mode)

General idea: Op. group: coarse → fine → individual

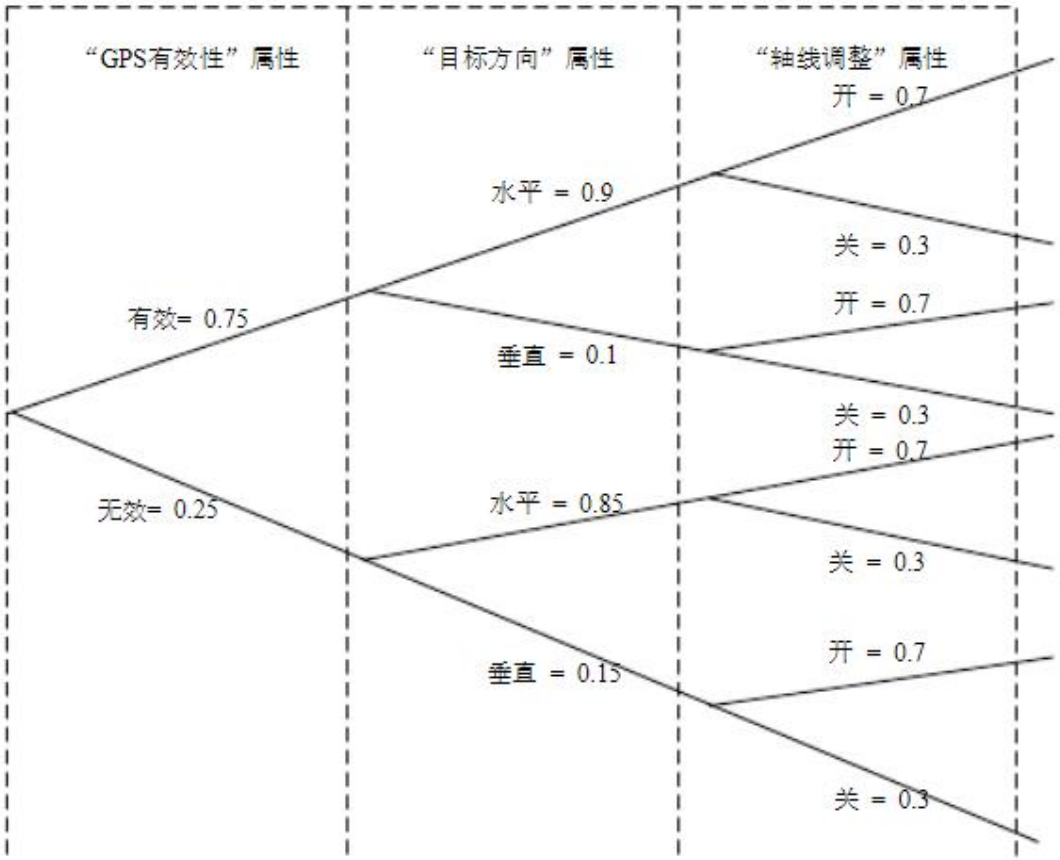
1. Identify initiators of operations.
2. Tabular or graphical representation.
3. Operations lists:

   initiators → consolidated.
4. Determine the occurrence rate.
5. Determine the occurrence probability.



3. 软件可靠性测试

Musa2 OP Development example



3. 软件可靠性测试

- Generic steps:
  - ▷ Musa-1: customer → user → sys. modes → functional → operational
  - ▷ Musa-2: initiator → representation → list → rate → probability
- Comparison
  - ▷ Size/environment/population differences.
  - ▷ One OP for each distinguished group
    - – Musa-1: user or operation group,
    - – Musa-2: operational modes.
  - ▷ Musa-1: 5 profiles, refined along.
  - ▷ Musa-2: different elements for 1 profile.

3. 软件可靠性测试与评估

- 软件可靠性评估是指运用**统计技术**对系统运行期间采集的软件**失效数据**进行处理并评估软件可靠性的过程。
  - 利用软件可靠性模型进行数据分析（参考第二部分 软件可靠性模型）

主要内容

1. 软件可靠性的定义
2. 可靠性模型及其评价标准
3. 软件可靠性测试和评估
4. 软件可靠性研究的主要问题

4. 软件可靠性研究的主要问题

- 软件可靠性模型
- 软件可靠性模型的应用问题
- 方法和工具问题
- 数据问题
- 。。。

学术研究报告：选题二

1. 针对某软件设计软件增长性测试具体实施方案。

2. 利用软件可靠性增长模型实际加以运用其分析软件可靠性，用开源数据或者模拟数据进行拟合分析。

主要内容

1. 软件可靠性的定义
2. 可靠性模型及其评价标准
3. 软件可靠性测试和评估
4. 软件可靠性研究的主要问题

http://www.cse.cuhk.edu.hk/~lyu/book/reliability/

业精于勤，荒于嬉！