第二章
软件可靠性分析

陈仪香

华东师范大学软件学院可信智能团队TrIG

2024年9月18日

TrIG 华东师范大学软件学院可信智能团队
Trustworthy Intelligence Group

1.1 软件

- 软件=程序+文档

著名计算机科学家徐家福

1.2 软件质量

质： 性质，一种事务区别于其他事务的根本属性

质量：产品、工作等的优劣程度。

陆书平，万森，张秋霞编，现代汉语字典，商务印书馆，2014.7

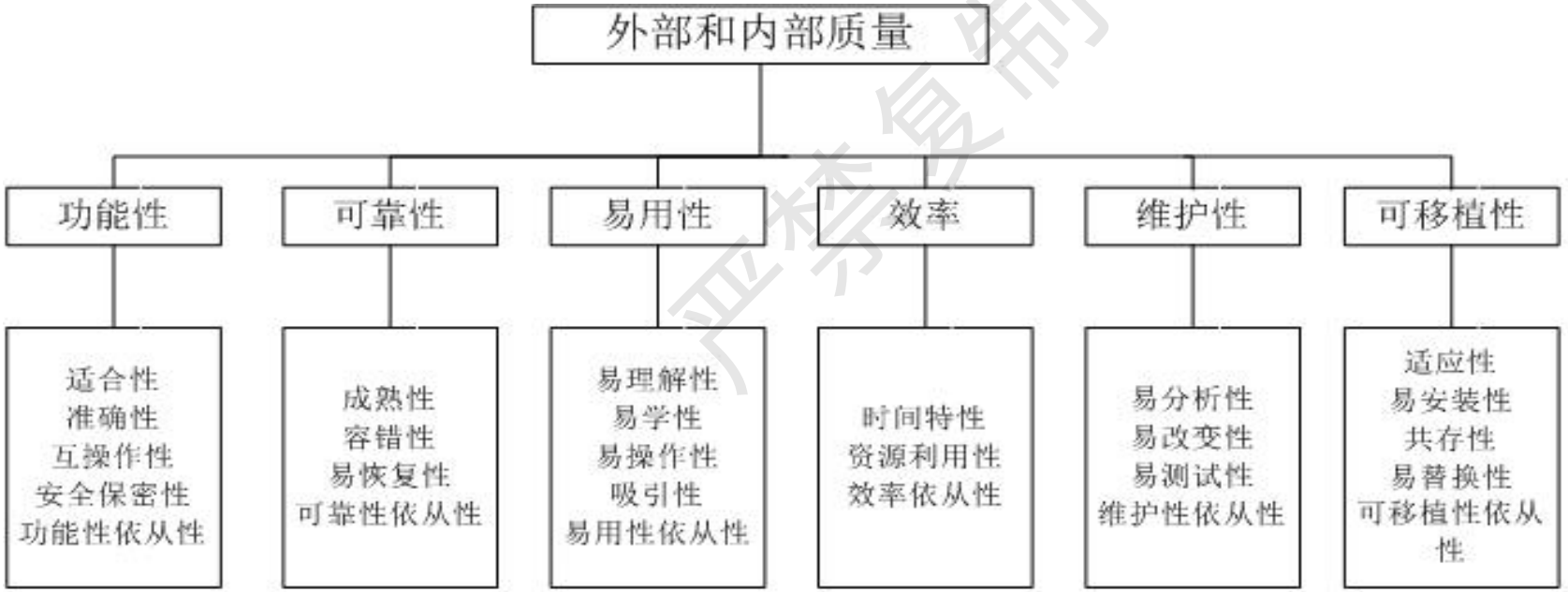
软件质量： 软件的优劣程度

软件=程序+文档（徐家福）

程序=算法+(程序设计语言)+数据结构（Wirth）

1.3 软件质量模型ISO 9126

1991年，ISO/IEC推出ISO/IEC 9126：1991《信息技术 软件产品评价 质量特性及其使用指南》，在此标准中，定义了六种质量特性，并且描述了软件产品评估过程的模型；



ISO 9126: Analysis of Quality Models and Measures，
Chapter · September 2010
DOI: 10.1002/9780470606834.ch10

1.3 软件质量模型

Gilles 1992

Six sessions were carried out with a range of companies from different market sectors: a management consultancy, a software house, a retail organization, a bank, a utility company and a manufacturing company。

The exercise defines relationships as:
(1) If characteristic A is enhanced, then Characteristic B is likely to be degraded, (-) or
(2) If characteristic A is enhanced, then Characteristic B is unlikely to be affected, (0) or
(3) If characteristic A is enhanced, then Characteristic B is likely to be enhanced (+).

学习什么内容？

课程内容安排为7章，与配套的教材相关，计划13周（上课26课时），17周考试。
（第2周中秋放假，第4周国庆节放假，第7周校运动会放假

学习什么内容

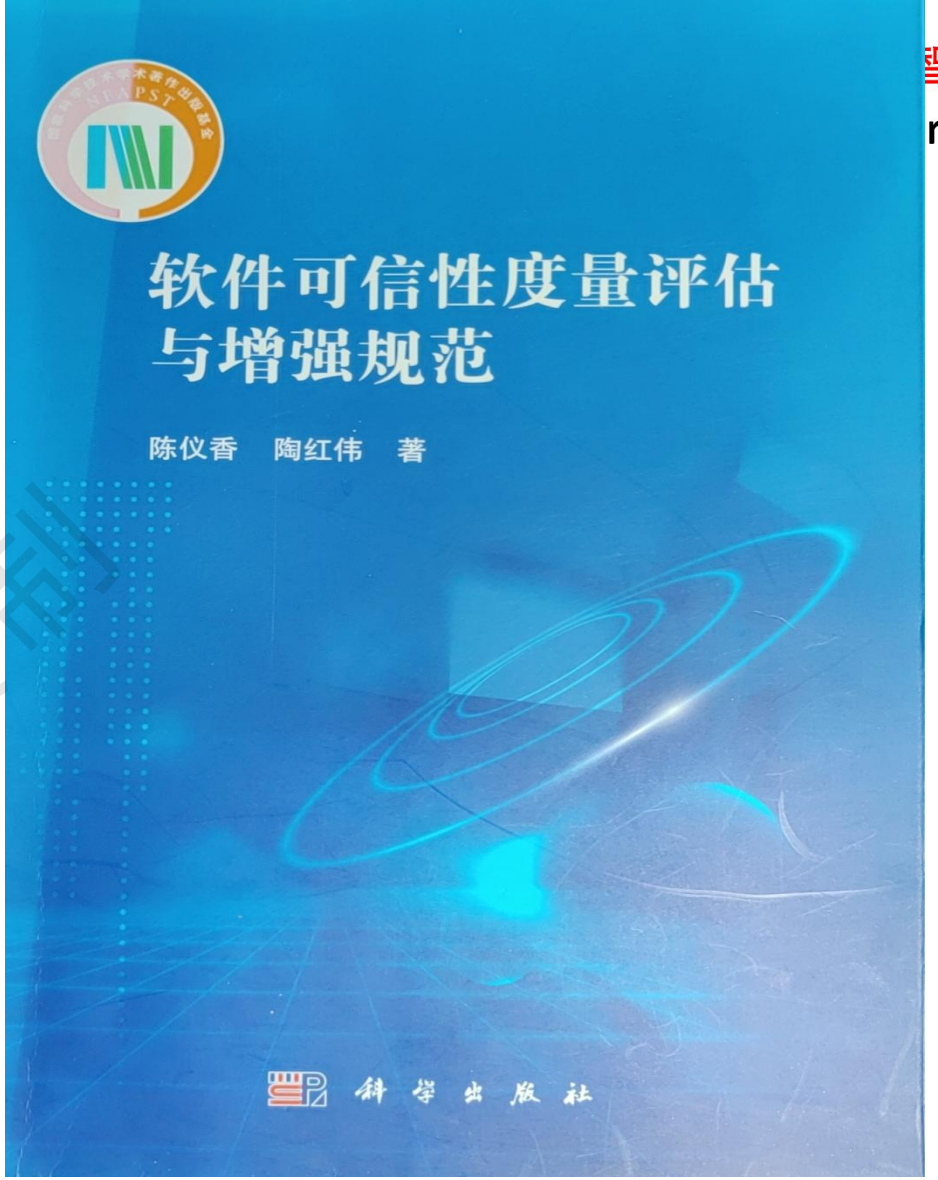
第一章 软件质量模型（支撑课程目标1、2）
本章重点解释软件质量的基本概念和模型，以及软件质量发展历史和标准。
讲授软件质量模型的概念和简史。

第二章软件可靠性模型（支撑课程目标1、2）
本章讲授软件可靠性概念和度量模型、以及软件可靠性分析。

第三章 软件可信性模型（支撑课程目标1、2）
本章讲授软件可信性的概念和发展过程，软件可信属性模型、软件属性分层模型和软件质量可信度量元模型。

第四章 基于属性的软件可信度量模型（支撑课程目标1、2）
本章讲授基于属性的软件可信度量性质和度量模型、基于属性分解的软件可信度量性质和度量模型、以及分级模型。

推荐教材



2.《软件可靠性工程》，主编：陆民 燕，国防工业出版社，2011.4年

1.《 软件可信性度量评估与增强规范 》（第一版），陈仪香，陶红伟著，科学出版社，2019年。

第二章 软件可靠性分析

2.1 软件可靠性

2.2 软件可靠性分析

2.3 软件可靠性评估

本章参考书

2. 《软件可靠性工程》，主编：陆民 燕，国防工业出版社，2011.4年

第二章 软件可靠性分析

2.1 软件可靠性

2.2 软件可靠性分析

2.3 软件可靠性评估

2.1 软件可靠性

●按照GB/T11457-95-软件工程术语，软件可靠性的定义为：

（1） 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。系统输入将确定是否运到已存在的缺陷（如果有缺陷存在的活）。

（2）在规定的时间周期内所述条件下程序执行所要求的功能的能力。

其中(1)是一个定量的定义，用此定义称之为可靠度更为确切，

而（2）则是一个定性的定义。

2.1 软件可靠性

规定的条件是指：(1) 软件运行的软、硬核环境：软件环境包括运行的操作系统、应用软件、编译系统、数据库系统等；硬件环境包括计算机的CPU、Cache、Memory、I/O等。

(2) 软件操作剖面：不严格地说，是指软件运行的输入空间及其概率分布。

软件的输入空间是指软件所有可能得输入值构成的空间。按照欧空局标准的定义，软件的操作剖面是指”对系统使用条件的定义。即系统的输入值用其按时间的分布或按他们在可能输入范围内得出现概率得分布来定义“。

2.1 软件可靠性

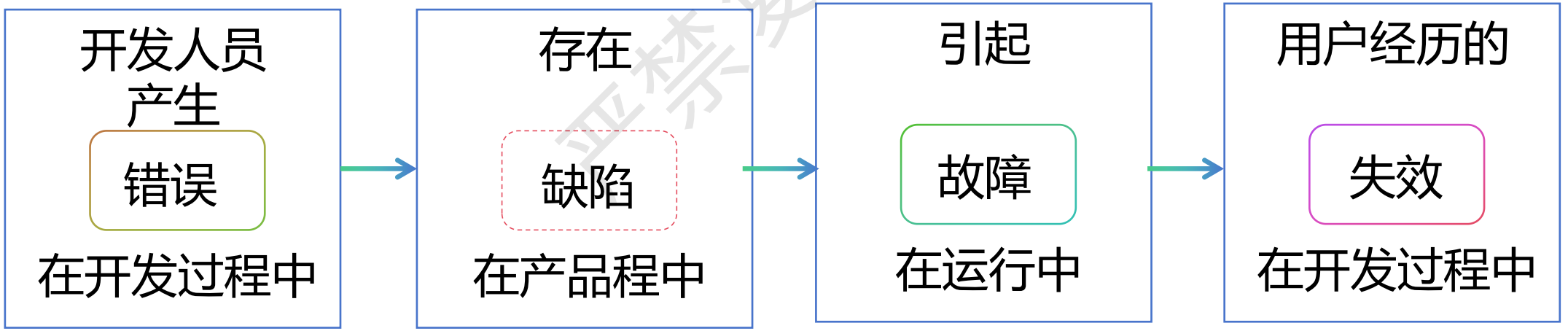
规定的时间：一般可分为执行时间、日历时间和时钟时间。执行时间是指执行一个程序所用的实际时间或中央处理器时间，或者程序处于执行过程中的一段时间；日历时间指的是编年时间，包括计算机可能未运行的时间。 时钟时间是指程序执行开始到程序执行完毕所经过的钟表时间，该时间包括了其他程序运行的时间。

大多数的软件可靠性模型是针对执行时间建立的，因为真正激励软件发生失效的是CPU时间。

规定的功能：软件的可靠性还与规定的功能密切相关。所谓规定的功能是指软件应完成的工作。事先必须明确规定的功能，只有这样，才能对软件是否发生失效有明确的判断。

2.1 软件可靠性

软件失效机理：当软件发生失效是， 我们说软件不可靠，发生的失效数越多，或者发生失效的时间间隔越短，我们认为软件越不可靠。那么软件为什么会失效呢？ 软件失效机理见下图：

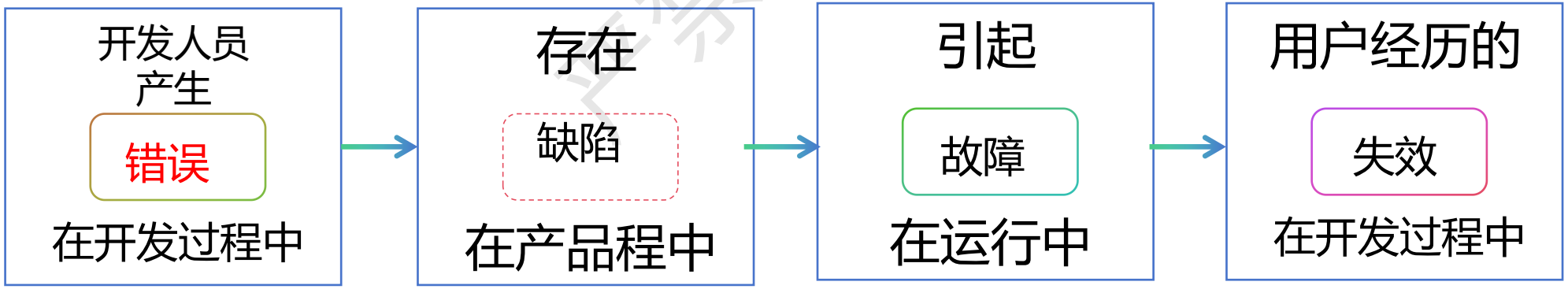


$$y = \frac{1}{x}, \quad z = \sqrt{x + y}$$

2.1 软件可靠性

错误(error)： 指开发人员在开发过程中出现的失误、疏忽和错误。例如，遗漏或误解软件需求规格说明中的用户需求，不正确的翻译或遗漏设计规格说明的需求。

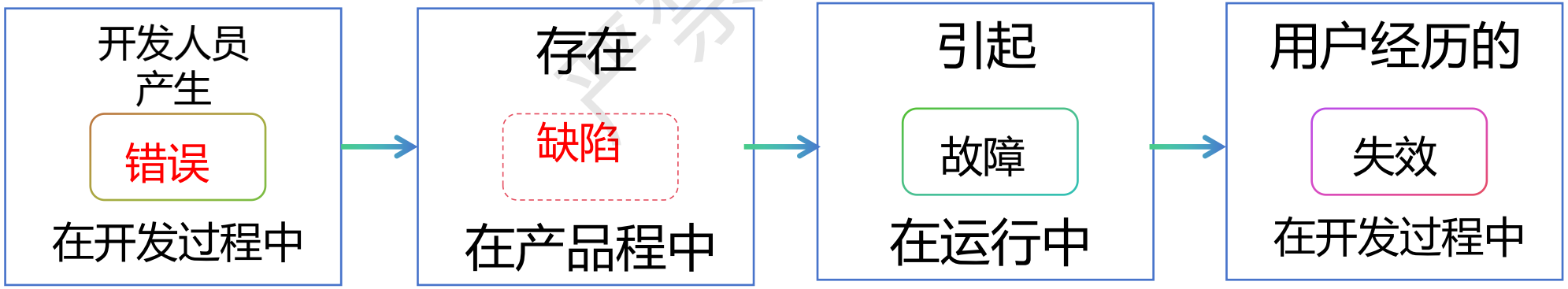
如爱国者导弹的舍二进制的25位以下的数字、火星气候轨道探测器MCO的单位不一致、阿丽亚娜5火箭高度和速度组件SRI的数值位数表示。



软件失效机理图

2.1 软件可靠性

缺陷(defect)： 指代码中能引起一个或多个失效的错误的编码（步骤、过程、数据定义等），软件缺陷是程序故有的，只要不修改程序去除已有的缺陷，缺陷就会永远留在程序中。另外，软件文档中不正确的描述也称为缺陷，如不正确的功能需求、遗漏的性能需求等。

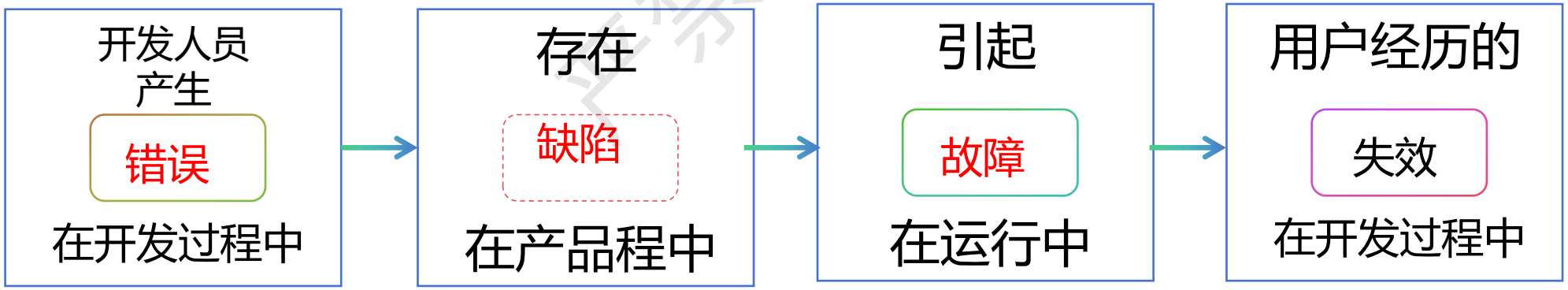


软件失效机理图

2.1 软件可靠性

故障(fault)： 指软件在运行过程中出现的一种不希望或不可接受的内部状态，通常是由于软件缺陷在运行时引起并产生的错误状态。

如不正确的数值，数据在传输过程中产生的偏差。

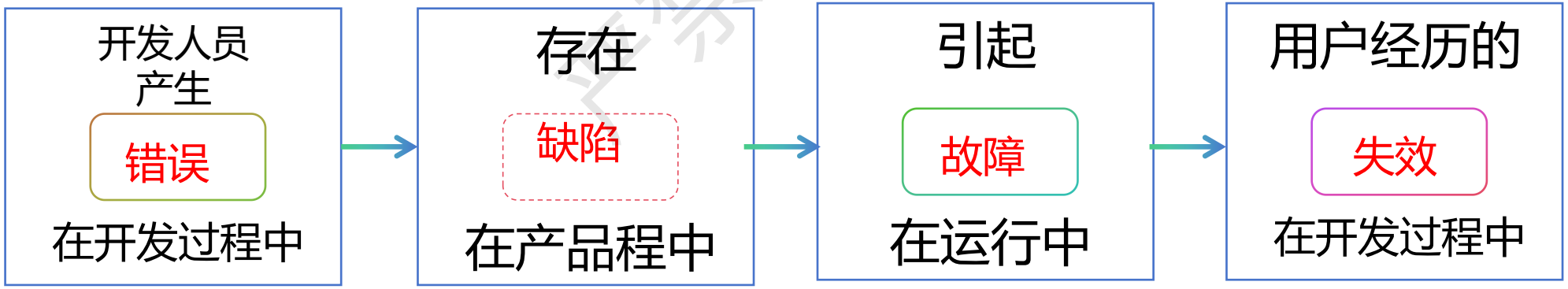


软件失效机理图

2.1 软件可靠性

失效(failure)： 指程序的运行偏离了需求，是动态运行的结果。软件执行遇到软件中的缺陷时可能会导致软件的失效。

如死机、错误的输出结果、没有在规定的时间内响应都是失效。爱国者导弹的0.000000095的误差夺走28条生命、阿丽亚娜5火箭高度和速度组件SRI的数值错误表示导致火箭发射失败。



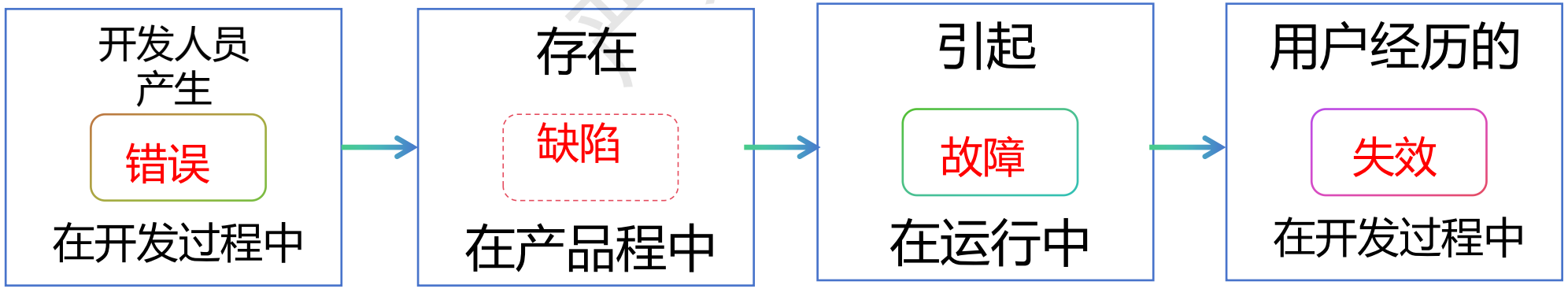
软件失效机理图

2.1 软件可靠性

软件失效的随机性：从软件可靠性的定义可以看出，软件可靠性是使用概率度量的，因此软件失效的发生是随机的。

但是经验又告诉我们，在使用一个程序是，在其他条件（如初始值、配置）不变的前提下，相同的输入数据会得到相同的输出结果，无论结果是否正确都是确定的，不会存在随机性问题。

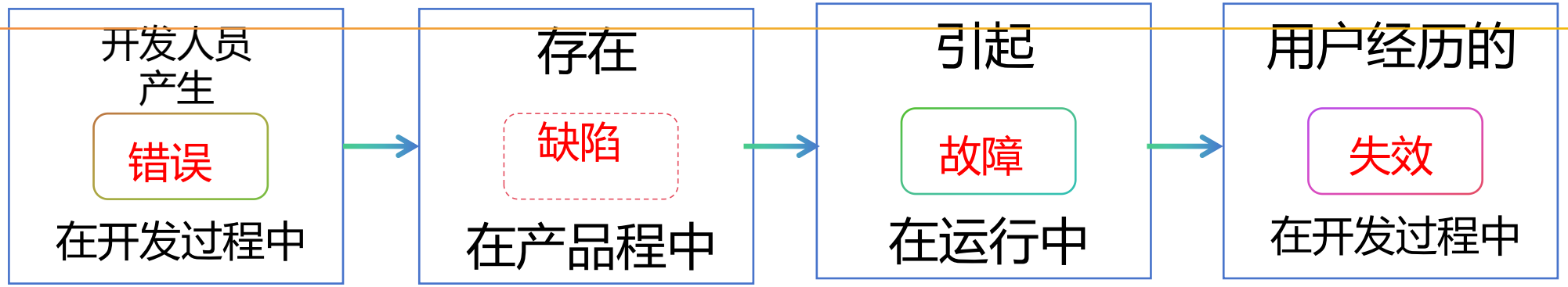
那么如何理解软件失效的随机性问题呢？



2.1 软件可靠性

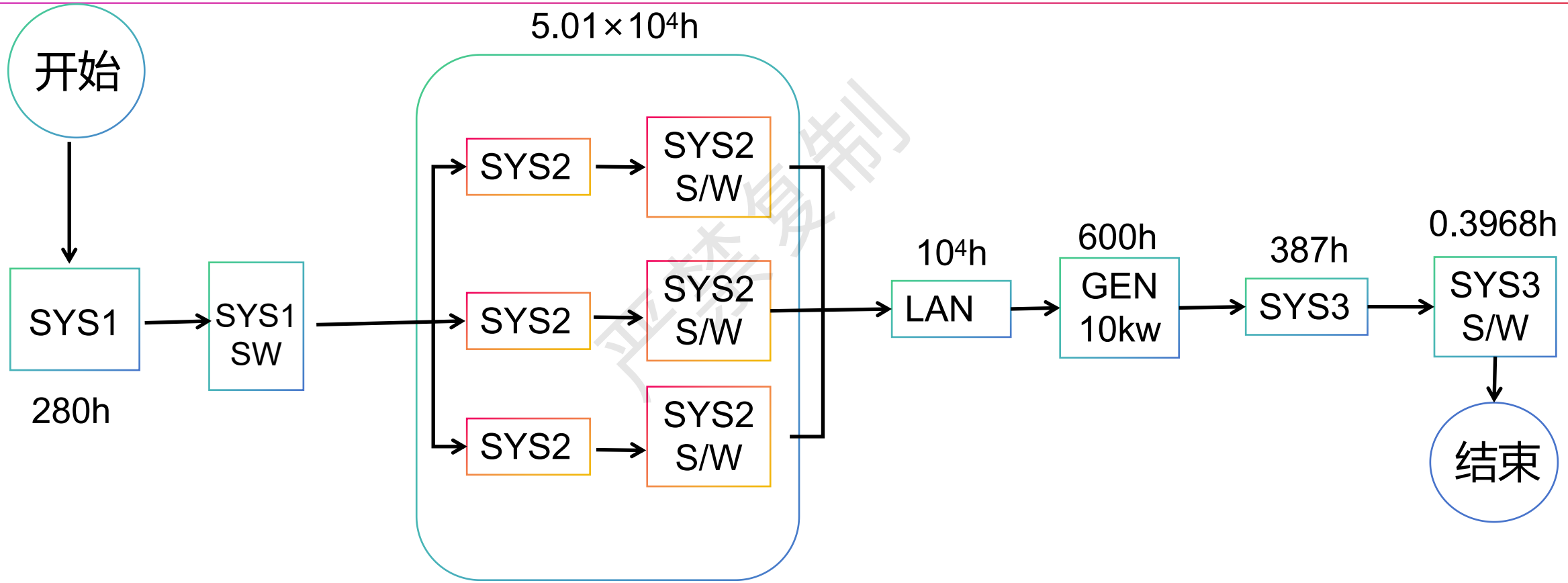
软件失效的随机性：从软件可靠性的定义可以看出，软件可靠性是使用概率度量的， 因此软件失效的发生是随机的。但是经验有告诉我们，在使用一个程序是，在其他条件（如初始值、配置）不变的前提下，相同的输入数据会得到相同的输出结果，无论结果是否正确都是确定的，不会存在随机性问题。 那么如何理解软件失效的随机性问题呢?

首先，程序执行的条件一般是不可预测的、是变化的，因而程序的使用千差万别。此次，缺陷或错误在程序中的位置是未知的。因为由于开发人员造成差错而引起的错误是复杂而难以预计的，即你无法预测程序中哪里存在缺陷。如果你事先知道哪里存在缺陷，你就可以事先把它除掉了。因此，在实际使用软件中，何时遇到程序中的缺陷从而导致软件失效就呈现出随机性。



2.1 软件可靠性

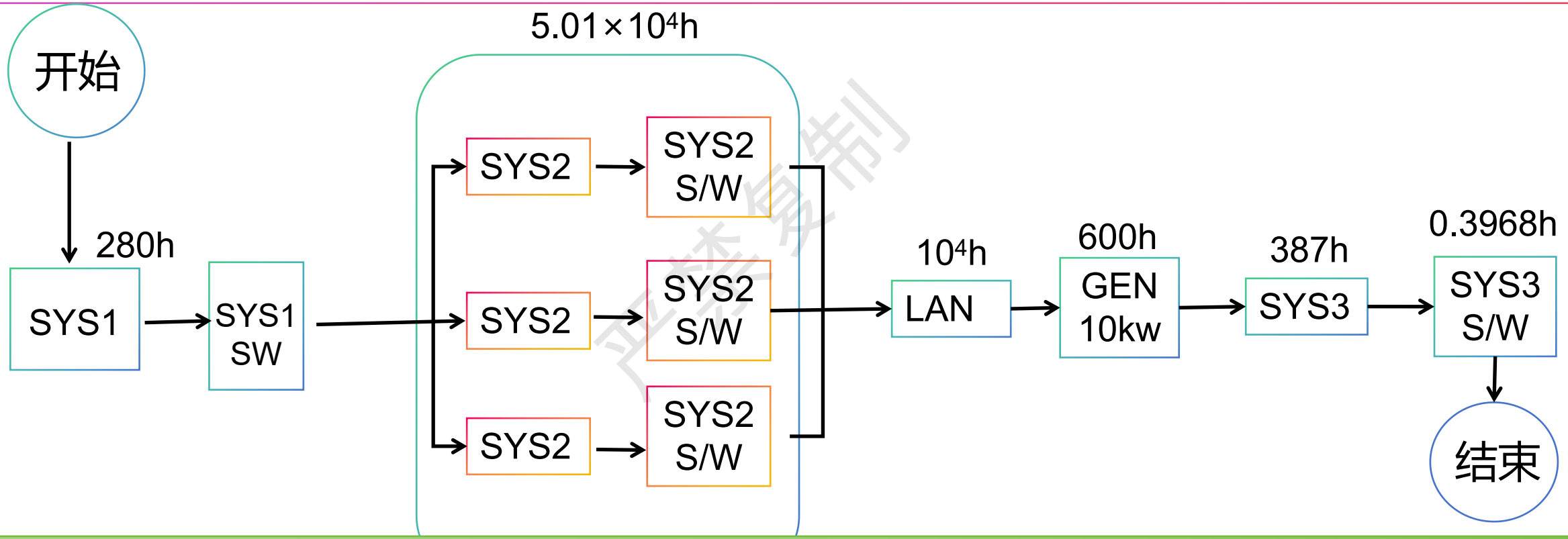
例子：某军用分布式处理系统结构示意图。SYS1、SYS2、SYS3三个分系统以及一个局域网LAN和10kW发电机组成， 而三个SYS2则构成一个冗余系统（至少两个工作即可）



其中SYS1、SYS2、SYS3均包含对应的硬件系统（图中用SYS1、SYS2、SYS3表示）和软件系统（图中使用SYS1 S/W、SYS2 S/W、SYS3 S/W表示）

2.1 软件可靠性

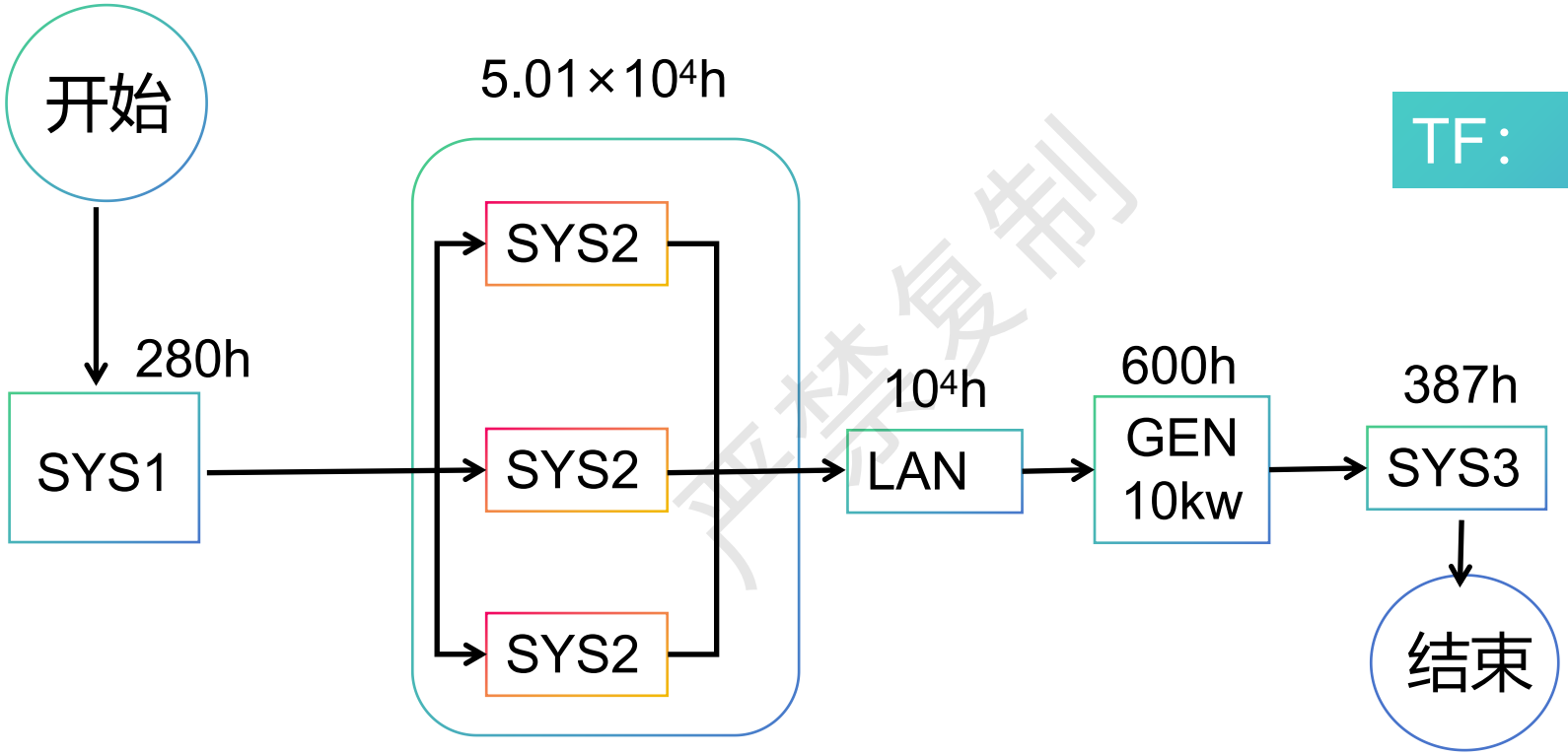
例子：某分布式处理系统结构示意图。SYS1、SYS2、SYS3三个分系统以及一个局域网LAN和10kW发电机组成， 而三个SYS2则构成一个冗余系统（至少两个工作即可）



依据系统可靠性要求，对硬件模块的平均失效时间MTTF(Mean Time To Failure)已经分别预计出并已标记在对应模块的上方，问此设计能否满足MTTF为100h的系统可靠性要求?

2.1 软件可靠性

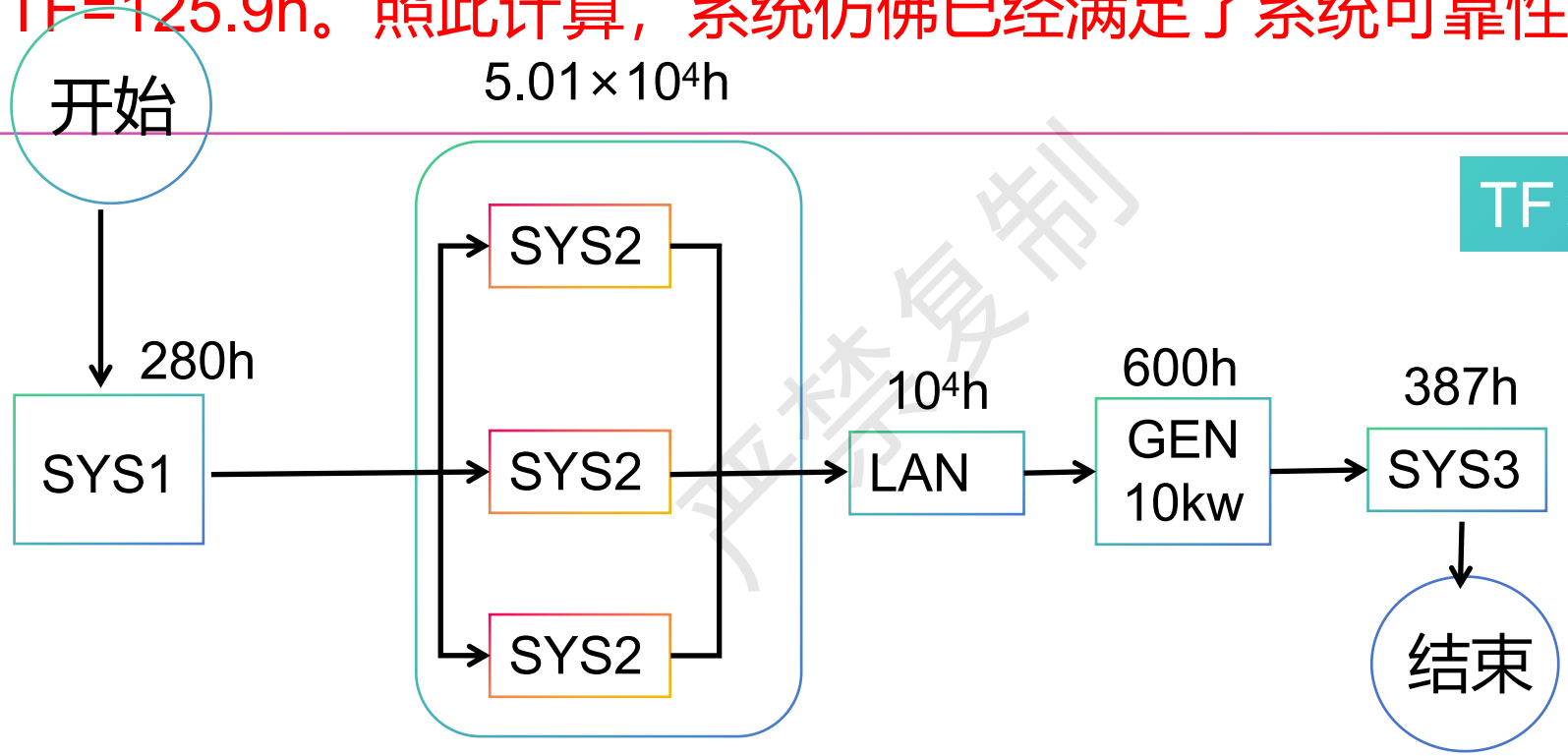
我们分两种情况进行分析：情况1：不考虑软件失效情况。此时整个系统是由SYS1、3个SYS2构成的冗余系统、LAN、GEN以及SYS3构成的一个串联系统其MTTF计算公式:



$$\frac{1}{\mathrm{TFs}}=\sum_{i=1}^{n}\frac{1}{\mathrm{T}_{\mathrm{TF}i}}=\frac{1}{280}+\frac{1}{5.01\times10^{4}}+\frac{1}{10^{4}}+\frac{1}{600}+\frac{1}{387}=0.007942(1/\mathrm{h})$$

2.1 软件可靠性

我们分两种情况进行分析：情况1：不考虑软件失效情况。此时整个系统是由SYS1、3个SYS2构成的冗余系统、LAN、GEN以及SYS3构成的一个串联系统其MTTF计算公式:
系统的MTTF=125.9h。照此计算，系统仿佛已经满足了系统可靠性的初步要求

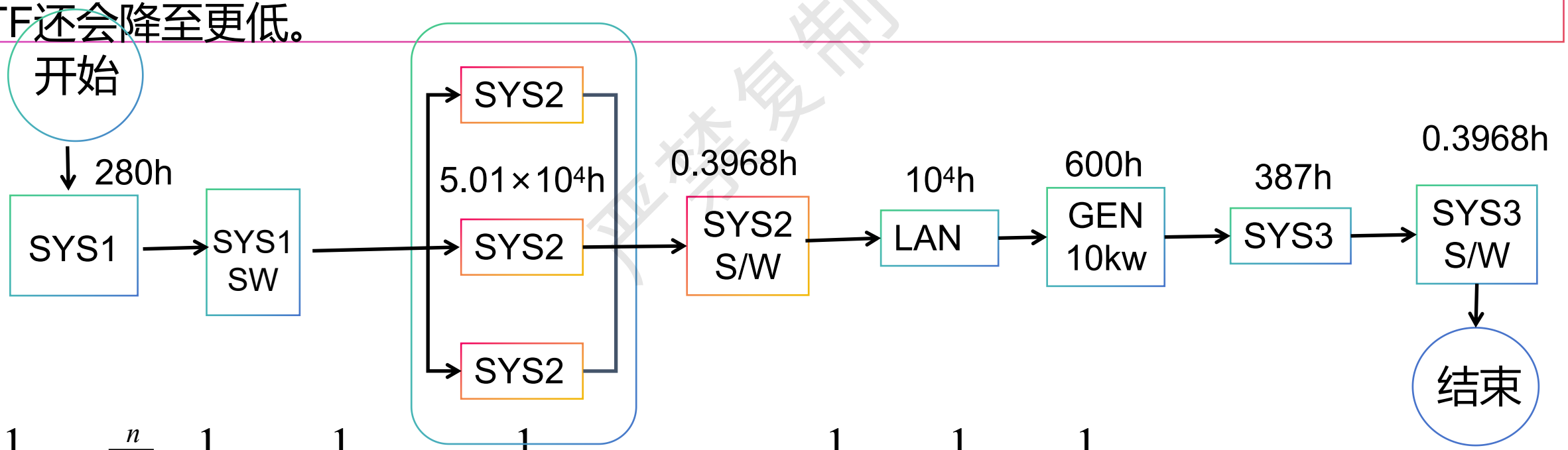


TF： To Failure

$$\frac{1}{\mathrm{TFs}}=\sum_{i=1}^{n}\frac{1}{\mathrm{T}_{\mathrm{TF}i}}=\frac{1}{280}+\frac{1}{5.01\times10^{4}}+\frac{1}{10^{4}}+\frac{1}{600}+\frac{1}{387}=0.007942(1/\mathrm{h})$$

2.1 软件可靠性

情况2：考虑软件失效情况。这个系统分别具有30000行源代码的SYS2和SYS3的初始失效率为2.52/h，即SYS2 S/W和SYS3 S/W的MTTF均为0.396825h。而在SYS2中，因为软件版本是相同的，冗余系统对软件失效无法起到冗余作用，故而考虑软件失效后只是相当于在原系统上同时串联上了SYS2 S/W和SYS3 S/W，可靠性框图如下：（不考虑SYS1 S/W失效情况下），系统MTTF为11.9min，若考虑SYS1软件失效性，则系统MTTF还会降至更低。



$$\frac{1}{\mathrm{TFs}}=\sum_{i=1}^{n}\frac{1}{\mathrm{T}_{\mathrm{TF}i}}=\frac{1}{280}+\frac{1}{5.01\times10^{4}}+2.52+\frac{1}{10^{4}}+\frac{1}{600}+\frac{1}{387}+2.52=5.046(1/\mathrm{h})$$

2.1 软件可靠性

这个例子告诉我们，软件可靠性对系统可靠性具有重要的影响，要保证系统的可靠性，不仅要对硬件可靠性提出定量要求，同时也必须对软件的可靠性提出定量的要求。

此外，确定软件可靠性要求可以使用户和开发人员找出影响系统整体可靠性的关键部分，对可能不可靠的部分进行重新设计，提高可靠性。

我们将面向软件用户的可靠性度量参数称为软件可靠性参数。

失效及失效等级

2.1 软件可靠性--失效

- 定义失效、失效严重等级

失效定义是判断软件失效的依据， 应在软件需求说明中给出明确的项目特定的软件失效定义， 这是确定软件可靠性要求、特别是有效进行可靠性度量的前提。 这是因为：

（1） 软件可靠性是对失效行为特性的刻画，不同的定义或理解会导致不同的度量结果。

（2）对于不同严重程度的失效用户常常有不同的要求，因此还需要对失效的严重等级进行划分。

- 软件失效等级的划分可从对人员生命、成本、任务完成等方面考虑。
- 分为1级到5级不等。 1级为最严重级。

2.1 软件可靠性--失效

● 定义失效、失效严重等级

软件失效等级的划分可从对人员生命、任务完成、成本等方面考虑。

按照对人员生命影响划分的软件失效严重等级

失效严重等级	定义
1	丧失生命或系统
2	对完成的任务有影响
3	可采用绕过的措施，因而对过程只是极小的影响（达到了任务目标）
4	对需求或标准有轻微的违反，用户在运行使用中看不见
5	表面问题，对于将来的行动应予以注意或追踪，但不一定是现在需解决的问题

2.1 软件可靠性--失效

- 定义失效、失效严重等级

软件失效等级的划分可从对人员生命、任务完成、成本等方面考虑。

按照成本划分的软件失效严重等级

失效严重等级	定义/美元
1	>100,000
2	10,000~100,000
3	1,000~10,000
4	<1000

2.1 软件可靠性--失效

● 定义失效、失效严重等级

软件失效等级的划分可从对人员生命、任务完成、成本等方面考虑。

按照对系统能力影响划分的软件失效严重等级

失效严重等级	定义
1	用户不能进行一项或多项关键操作
2	用户不能进行一项或多项重要操作
3	用户不能进行一项或多项操作，但是有补救方法
4	一项或多项操作中的小缺陷

2.1 软件可靠性--失效

● 对于每个失效严重等级可以选择不同的软件可靠性参数，取决于用户关系的角度。下表给出了不同失效严重等级的软件可靠性要求示例。

软件可靠性定量要求示例--跨网络交易

失效严重等级	等级描述	失效示例	可靠性要求
1	暂时的，有毁损	一次跨网络的交易模式导致了数据库损毁	无法定量给出，在系统的生存周期中应该永远不发生
2	暂时的，无毁损	无法读取一张非损坏卡上的磁条数据	成功率1次/1000次交易
3	永久的，无毁损	系统对任何输入的卡都没反应，软件必须重启来更正失效	失效率1次/1000天

2.1 软件可靠性--度量参数

这个例子告诉我们，软件可靠性对系统可靠性具有重要的影响，要保证系统的可靠性，不仅要对硬件可靠性提出定量要求，同时也必须对软件的可靠性提出定量的要求。

此外，确定软件可靠性要求可以使用户和开发人员找出影响系统整体可靠性的关键部分，对可能不可靠的部分进行重新设计，提高可靠性。

我们将面向软件用户的可靠性度量参数称为软件可靠性参数。

2.1 软件可靠性--度量参数

- 可靠度：

软件可靠度R是指软件在规定的条件下， 规定的时间段内完成预定的功能的概率。或者说是软件在规定时间内无失效发生的概率。

设规定的时间段t_0，软件发生失效的时间为,则

$$R(t_0)=P(\xi>t_0)$$

说明： 该参数是关于软件失效行为的概率描述，是软件可靠性的基本定义，它和硬件可靠性的定义相同。可利用一般的概率规律将它和系统其他部分，如硬件部分， 组合在一起。

该参数适用于对规定时间内无失效工作要求高的系统， 如航空电子系统。

2.1 软件可靠性--度量参数

●失效率和失效强度：

这两个术语在有关软件可靠性的书籍、文献中经常看到。但它们并不相同，不过有着密切关系。

1）失效率定义

失效率是指在t时刻尚未发生失效的条件下，在t时刻后单位时间内发生失效的概率。即：设ξ为发生失效的时间， Z为失效率，则有

$$Z(t)=\lim_{\Delta t\to 0}\frac{P(t<\xi<t+\Delta t\mid \xi>t)}{\Delta t}=\lim_{\Delta t\to 0}\frac{P(t<\xi<t+\Delta t)}{P(\xi>t)\cdot\Delta t}=\lim_{\Delta t\to 0}\frac{R(t)-R(t+\Delta t)}{R(t)\cdot\Delta t}$$

$$=-\frac{1}{R(t)}\cdot\frac{\mathrm{d}R}{\mathrm{d}t}=\frac{1}{R(t)}\cdot f(t)$$

2.1 软件可靠性--度量参数

- 失效强度
- 假设软件在是t时刻发生的失效数是N(t)， 显然N(t)是一个随机数，且随时间t的变化而不同， 即{N(t), t>0}是一个随机过程。设u(t)为随机变量N(t)的均值， 即有 u(t)=E(N(t)), 则t时刻的失效强度λ(t)定义为：

$$\lambda(t)=\frac{\mathrm{d}u(t)}{\mathrm{d}t}$$

2.1 软件可靠性--度量参数

- 失效率和失效强度
- 说明：从定义中可以看出， 失效率和失效强度是两个不同的概念， 失效率的定义和硬件可靠性中瞬时失效率的定义完全一致的， 是基于寿命的观点给出的。 它是一个条件概率密度。而失效强度则是基于随机过程定义的，是失效数均值的变化率。

- 失效率/失效强度适用于要求失效发生频率比较低的系统， 如操作系统。

2.1 软件可靠性--度量参数

- 平均失效前时间MTTF和平均失效间隔时间MTBF

1)平均失效前时间MTTF(Mean Time To Failure)

- MTTF是指当前时间到下一次失效时间的均值。

假设当前时间到下一次失效的时间为ξ，ξ具有累计概率密度函数 $F(t)=P(\xi \le t)$，即可靠度函数 $R(t)=1-F(t)=P(\xi > t)$

则基于MTTF的软件可靠度T_{TF}为：

$$T_{TF}=\int_0^{\infty} R(t)dt$$

2.1 软件可靠性--度量参数

●平均失效前时间MTTF和平均失效间隔时间MTBF

2)平均失效间隔时间MTBF(Mean Time Between Failure)定义

■ MTBF是指两次相邻失效时间间隔的均值。

■假设两次相邻失效时间间隔为ξ，具有累计概率密度函数 $F(t)=P(\xi \le t)$

即可靠度函数 $R(t)=1-F(t)=P(\xi > t)$

则基于MTBF的软件可靠度T_{BF}为

$$T_{BF}=\int_0^{\infty} R(t)dt$$

2.1 软件可靠性--度量参数

- 平均失效前时间MTTF和平均失效间隔时间MTBF

3)说明：在硬件可靠性中， MTTF用于不可修复产品， MTBF用于可修复产品；对于软件则不能简单地用同样的概念进行区分。

软件不存在不可修复的失效，亦即，软件失效是可以修复的。但是，修复活动对失效特性的影响和硬件存在着很大的不同。

2.1 软件可靠性--度量参数

- 平均失效前时间MTTF和平均失效间隔时间MTBF

硬件	软件
存在失效不可修复的产品，如轮胎、电刷	不存在不可修复的失效，即软件失效都是可修复的。
失效后若进行完全修复，则失效特性不变， 因而MTBF不变	失效后完全修复，失效特性发生变化，因而MTBF变化
如果失效后不修而继续使用，将导致系统功能降级（与失效前之比），系统的失效特性发生变化	失效后可以不修而继续使用，但与未失效之前比，并不存在着功能降级。如果对软件的操作剖面也保持不变的话，软件失效特性也保持不变。这是因为只要不修改，导致失效的程序中的缺陷无论失效与否都存在，只要满足一定的条件，缺陷即可导致软件失效。

2.1 软件可靠性--度量参数

- 一般的软件可靠性参数选取原则

（1）对失效发生频率要求较低的系统，可靠性参数可选失效率或失效强度，如操作系统、电话交换系统软件等。

（2）对在规定时间内能无失效工作要求比较高的系统，可选可靠度作为软件可靠性参数，如火力控制系统软件等。

（3）对使用比较稳定的软件，可选平均失效前时间/平均失效间隔时间作为软件可靠性参数，如通用软件包等。

2.1 软件可靠性--度量参数

缺陷密度（Defect Density,DD）： 是软件缺陷的基本度量，可用于设定产品质量目标，支持软件可靠性模型（如Rayleigh模型）预测潜藏的软件缺陷，因而对软件质量进行分析、跟踪和管理。支持基于软件缺陷计数的软件可靠性增长模型(如Musa-Okumoto模型），对软件质量目标进行跟踪并判定能否结束软件测试。

用途：这个度量分析每个版本或模块中每千行代码中的缺陷数量。这样可以在一系列版本或模块中追踪软件质量。

适用阶段：适用于需求、设计、编码、测试、使用及维护阶段。

2.1 软件可靠性--度量参数

缺陷密度DD：

数据元素：D是每个版本或每个模块规定严重性等级下的缺陷（或偏差报告）数。

KSLOC是在每个版本或每个模块中可执行代码和非可执行数据声明的千行源代码数(=代码行数/1000)。(Line of Codes, LOC)

缺陷密度DD:

$$DD = \frac{D}{KSLOC}$$

2.1 软件可靠性--度量参数

故障密度FD（Fault Density）：

$$FD = \frac{F}{KSLOC}$$

用途：使用这个度量，可以通过按严重性分类将计算的故障密度与目标值比较来确定是否已经完成足够的测试。

使用阶段：测试、使用和维护

数据元素：F是指每个版本或模块中发现的导致具有规定的严重性等级的失效的唯一（注：唯一是指相同的故障计为一次）故障数。

KSLOC 是在每个版本或每个模块中可执行代码和非可执行数据声明的千行源代码数。

2.1 软件可靠性--度量参数

需求依从性（Requirements Compliance）：

用途：该度量反映软件需求分析工作的度量。根据该度量可以确定在需求分析阶段，软件需求规格说明中的需求不一致的比例，需求不完整的比例，需求曲解的比例，可以确定主要的需求问题类型，从而有效改进软件需求分析的质量。

适用阶段：需求阶段

2.1 软件可靠性--度量参数

需求依从性（Requirements Compliance）：

数据元素：I_R 表示由于不一致的需求而引起的错误比例；

N_R 表示由于不完整的需求而引起的错误比例；

M_R表示由于曲解的需求而引起的错误比例。

N1表示在一个版本或者模块中不一致的需求数。

N2表示在一个版本或者模块中不完整的需求数。

N3表示在一个版本或者模块中曲解的需求数。

2.1 软件可靠性--度量参数

需求依从性（Requirements Compliance）：

数据元素：

I_R 表示由于不一致的需求而引起的错误比例；

N_R 表示由于不完整的需求而引起的错误比例；

M_R表示由于曲解的需求而引起的错误比例。

N_1表示在一个版本或者模块中不一致的需求数。

N_2表示在一个版本或者模块中不完整的需求数。

N_3表示在一个版本或者模块中曲解的需求数。

计算公式：

$$I_R = (\frac{N_1}{N_1 + N_2 + N_3}) \times 100\%$$

$$N_R = (\frac{N_2}{N_1 + N_2 + N_3}) \times 100\%$$

$$M_R = (\frac{N_3}{N_1 + N_2 + N_3}) \times 100\%$$

$N_1+N_2+N_3$是一个版本或者模块中不一致、不完整、易曲解的需求数之和。

第2.1节作业

总结软件可靠性定义以及相关度量参数，字数不少于800字

TrIG
华东师范大学软件学院可信智能团队
Trustworthy Intelligence Group