

第11章

软件老化可信度量模型

陈仪香

华东师范大学软件学院可信智能团队TrIG

2024年12月18日

第11章 软件老化可信度量模型

11.1 软件老化

11.2 老化类别

11.3 度量元模型

11.4 可信计算框架



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

软件老化

软件老化是一种在经历长时间运行后的系统中极有可能出现的现象，主要表现为软件系统在长时间运行后，受内存泄漏、碎片问题、数值累计错误等原因影响，会产生软件状态异常或性能下降的情况，情况严重时甚至会出现软件失效或系统宕机[1]。

该现象是在许多软件系统中根据经验观察得出的，一般特征是随着系统或进程的运行时间增加，其故障率也会增加[2]，而引起软件老化的缺陷统称为 Aging-Related Bug (**ARB**)。

[1] 孟令泽. 基于人工蜂群算法与 BP 神经网络模型的云软件老化预测机制研究[D]. 内蒙古大学, 2020.

[2] Michael Grottke, Rivalino Matias, and Kishor S Trivedi. The fundamentals of software aging. In 2008 IEEE International conference on software reliability engineering workshops (ISSRE Wksp), pages 1–6. Ieee, 2008.

软件老化

- 失信证据是指可以反映软件运行结果偏离预期或性能下降的相关指标。
- 由于与运行时间相关的 bug 在所有 bug 中占比非常小，所以需要从大量错误报告中提取失信证据。
- 为了使获取的失信证据涉及面更广、反映问题更全面，将《致命 bug》中所述历史上知名的软件事故。

TURING

이리온

[韩] 金钟河 著 叶蕾蕾 译

致命 Bug

软件缺陷的灾难与启示



中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

编号	缺陷名称	缺陷描述	国别 年份
1	爱国者导弹	0.0000000095的误差夺走28条生命：系统舍去二进制的25位以下的数字，只保存24位，所以每运行1个clock，就会产生（按照十进制的话约为0.0000000095）的误差。	美国 1991
2	火星探测器	火星气候轨道探测器MCO在着陆过程中，MCO向地面站传回发动机推力信息时，使用国际标准单位“牛顿”，而地面站使用的程序是SM_FORCE，使用的却是美国通用的力的单位--磅力，1磅力约等于4.45牛顿。	美国 1999
3	阿丽亚娜5	火箭发射37秒后，发生爆炸。测定当前火箭高度和速度组件SRI,阿5使用阿4的SRI。规定输入SRI的数值是16位元整数。但当时输入SRI的数值是64位元浮点数，因而SRI是无法使用16位元整数表示的。用SRI的16位元整数形式对该数值进行转换过程造成了溢出错误。无法发出正确的火箭高度和速度。	欧洲 1996

- Grottke 等人 研究使用的四个带有公共漏洞库和活跃漏洞库的开源软件系统作为研究对象，在前人研究的基础上进一步探究 bug 与软件老化之间的联系，提取失信证据。这四个拥有众多用户的软件系统涵盖了不同类型的软件，可以从中获取大量错误报告来提取失信证据：

1. Linux 内核，用于从嵌入式系统到超级计算机等多个领域，是一个功能丰富的操作系统； <https://bugzilla.kernel.org> (Linux 2.6),

2. MySQL，占有重要的市场份额，是最常用的数据库管理系统之一；
<http://bugs.mysql.com> (MySQL 5.1)

3. Apache HTTPD 服务器，为用户提供动态 Web 服务，是 Apache 产品线下著名的一款 web 服务器； <https://issues.apache.org/bugzilla> (Apache HTTPD 2),

4. Apache AXIS 框架，许多公司采用其 Web 服务来运行 Web 应用程序，是一个开源、基于 XML 的 Web 服务架构。

<https://issues.apache.org/jira/secure/IssueNavigator.jspax> (Axis 1)

- Grottke 等人 研究使用的四个带有公共漏洞库和活跃漏洞库的开源软件系统作为研究对象，在前人研究的基础上进一步探究 bug 与软件老化之间的联系，提取失信证据。这四个拥有众多用户的软件系统涵盖了不同类型的软件，可以从中获取大量错误报告来提取失信证据：

1. Linux 内核，用于从嵌入式系统到超级计算机等多个领域，是一个功能丰富的操作系统；（Linux 2.6）
2. MySQL，占有重要的市场份额，是最常用的数据库管理系统之一；（MySQL 5.1）
3. Apache HTTPD 服务器，为用户提供动态 Web 服务，是 Apache 产品线下著名的一款 web 服务器；（Apache HTTPD 2）
4. Apache AXIS 框架，许多公司采用其 Web 服务来运行 Web 应用程序，是一个开源、基于 XML 的 Web 服务架构。（Apache Axis 1）

●例 3.1 根据 Linux 缺陷库中 Bug3171 的记载，在 2004 年 Linux 的日志文件系统 ext3 运行的一段时间后自己的磁盘空间所剩无几，这是一个与运行时间相关的 bug。详细研究错误报告发现，内核本应在 ext3 删除日志后回收相应的磁盘空间，但是由于源码中存在单行嵌套错误，文件句柄一直没有关闭导致回收失败，进而造成这次磁盘空间泄露，得到表X.1。

出处	描述	失信原因	失信证据
LinuxBug3171	Linux 的日志文件系统 ext3 运行的一段时间后发生磁盘泄漏	磁盘空间泄漏导致磁盘空间占用、虚拟内存不足、系统效率降低	软件占用预期之外的磁盘空间占总空间的比
			对系统性能产生影响所需的时间

- 例 3.2 根据《致命 bug》中温哥华证券交易所事件的记载，在 1982 年初温哥华 证券交易所推出了初始设定为 1000 点的股票指数。该所每天要对股票进行 2800 次计算和更新，由于程序编写失误导致计算时会直接舍去小数点后三位而非四 舍五入，即使一次计算只产生 0.001 的误差，积少成多后导致股票指数每天都会下滑一到两个百分点，一年后股票指数几乎缩水一半。造成这次重大软件事故的 正是舍入误差的累积，由此得到表X.2。

出处	描述	失信原因	失信证据
温哥华证券交易所事件	温哥华交易所推出的股票指数一年后缩水近一半	微小的运算操作误差经过较长时间的累积导致结果偏离预期	误差对结果产生了不可忽视的影响，所需的时间

- 从 4 大缺陷库中统计的 bug 数总计 963 个，经过筛选，共得 55 个与软件老化有关的错误报告，结合《致命 bug》中涉及的 18 个著名的软件事故，其中有 4 个与软件老化有关，以此作为设计度量元的依据。

第11章 软件老化可信度量模型

11.1 软件老化

11.2 老化类别

11.3 度量元模型

11.4 可信计算框架



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

老化类别

基于软件老化的失信证据度量元除了表现形式外，也可以从导致软件老化的原因角度进行研究。参考 Cotroneo 等人 对软件老化相关 bug 的分类方 法，将度量元按失信的成因进行分类，分为 5 类，如表X.9，

软件老化原因分类	描述
MEM（内存管理）	度量元导致与内存管理有关的错误积累（例内存泄漏，未刷新缓冲区）
STO（存储空间）	度量元导致影响存储空间的错误累积（例如，错误占用磁盘空间）
LOG（逻辑资源）	度量元导致“其他逻辑资源”泄露，即依赖于系统的数据结构（例如，使用未释放的套接字或索引节点）
NUM（误差累积）	度量元导致数字误差累积（例如，舍入误差，整数溢出）
ARU(未知原因 ARB’s Subtype:Unknown	度量元仅知道发生错误的该率会随着时间而增加（例如，因为它仅在多次调用某个功能之后才导致失败），或是没有足够的关于确切失败机制的信息，无法找到明确的原因

老化类别

ID	A-TYPE	REASON
linux3171	STO	单行 {} 嵌套错误导致大量磁盘空间泄漏
linux5137	MEM	驱动程序错误，巨大的 fifo 溢出
linux7959	ARU	Tg3 驱动程序存在挂起/恢复问题
mysql34335	NUM	主键自动增长时 bigint 溢出崩溃
mysql34335	NUM	主键自动增长时 bigint 溢出崩溃
mysql40386	LOG	截断后不刷新查询缓存，如果表不是空的，record 的值永远不能是 0

老化类别

ID	A-TYPE	REASON
mysql46656	MEM	删除表时会发生内存泄漏，关机时没有释放内存
mysql49535	MEM	可用内存检查使崩溃恢复速度降低数十倍， 在读取崩溃恢复日志的过程中使用超过 90% 的 CPU
mysql56340	LOG	非索引更新后，innodb 过于频繁地更新索引统计信息
apache13511	STO	如果日志文件达到 2GB，Apache 会失灵。错误日志 达到 2G 后就无法写入了
apache29962	MEM	字节范围过滤器将缓冲整个响应在了内存中
AXIS-1248	ARU	SOAPHeaderElement.detachNode（）中的空指针异常。 第二次调用 detachNode（）后，将引发空指针异常。

第11章 软件老化可信度量模型

11.1 软件老化

11.2 老化类别

11.3 度量元模型

11.4 可信计算框架



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

度量元模型

失信证据是指可以反映软件运行结果偏离预期或性能下降的相关指标。

失信证据对应的风险值越大， 风险等级也就越高， 可能导致软件进入预期之外状态的种类也就越多， 表X.4给出风险等级以及可能导致的最坏风险 状态的映射关系。

风险等级	风险值	风险状态
V	10	运行时长会显著影响软件运行结果
IV	9	运行时长有大概率影响软件运行结果或对性能产生影响
III	7	运行时长有概率影响软件运行结果或对性能产生影响
II	4	运行时长对运行结果没有影响， 但是有概率对软件性能产生影响
I	1	运行时长对软件没有影响

度量元模型

鉴于风险等级越高，可能导致的异常状态数就越多，平均到单个风险状态对 风险值的影响应逐渐降低，因此将风险值按近似黄金分割的比例逐层递减直至 失信证据对软件运行没有影响，将此时的风险值设为 1。
还是以 LinuxBug3171 为例，如表X.5。

出处	度量元名称	失信原因	失信证据	风险等级对应指标		风险值
Linux3171	磁盘空间泄露	磁盘空间泄漏导致磁盘空间占用、 虚拟内存不足、 系统效率降低	软件占用预期之外的磁盘空间占总空间的比	87%-100%	V 级	10
				59%-87%	IV 级	9
				40%-59%	III 级	7
				20%-40%	II 级	4
				0-20%	I 级	1
		对系统性能产生影响平均所需的时间（h）		0-10	V 级	10
				10-30	IV 级	9
				30-60	III 级	7
				60-110	II 级	4
				>110	I 级	1

度量元模型

当 Linux 的日志文件系统运行 48 小时后，软件就会额外占用系统 80% 的磁 盘空间，导致系统运行失常，这里两个失信证据分别为 80% 和 48 小时，对应了 IV 级和 III 级的风险等级，两个失信证据对应的风险值就为 9 和 7。

出处	度量元名称	失信原因	失信证据	风险等级对应指标		风险值
Linux3 171	磁盘空间泄露	磁盘空间 泄漏导致 磁盘空间 占用、 虚 拟内存不 足、 系统 效率降低	软件占用预 期之外的磁盘 空间占总空间 的比	87%-100%	V 级	10
				59%-87%	IV 级	9
				40%-59%	III 级	7
				20%-40%	II 级	4
				0-20%	I 级	1
		对系统性能产 生影响平均所 需的时间（h）		0-10	V 级	10
				10-30	IV 级	9
				30-60	III 级	7
				60-110	II 级	4
				>110	I 级	1

基于软件老化的失信证据度量元是指可能导致软件老化的可被度量的 直接或间接软件项目元素，通常在软件开发和运维服务过程中获取，用六元组表示：

Metric element based on evidence=<Name,M-Type,Reason,A-Type,Evidence,R-Value>

其中，

- ◆ Name 表示度量元的名称；
- ◆ M-Type 表示度量元的类型；
- ◆ Reason 表示度量元 失信的原因；
- ◆ A-Type 表示软件老化原因的类型；
- ◆ Evidence 表示失信证据；
- ◆ R-Value 表示失信证据的风险值。

软件老化的表现形式多种多样，有些表现为运行中由于涉及某些操作导致系统崩溃，有些则是在运行一段时间后突然宕机，还有的可能是随着运行时间的增加系统性能不断下降等等。

为了覆盖所有可能的情况，设计了三种类型的度量元，分别为：

触发型度量元：指只有在特定情况下才会触发问题的软件项目元素。

增长型度量元：指随运行时间增长，原本影响较小却可不断累积的软件项目元素。

突变型度量元：是指某些指标随时间增长至临界点时会发生突变的软件项目元素。

触发型度量元：指只有在特定情况下才会触发问题的软件项目元素。

例如 “LinuxBug11935 权限管理错误”，软件运行中遇到需要对文件进行读写的情况，而该文件状态为只读，导致软件无法正常运行。具体如表X.6。

表 X.6 触发型度量元分类示例

度量元名称	度量元类型	失信原因	失信证据
权限管理错误	触发型	系统无法访问自己本该被授权的资源或访问自己未被授权的资源	操作相关文件状态不满足规定要求的比例

增长型度量元：指随运行时间增长，原本影响较小却可不断累积的软件项目元素。例如“MySQLBug46656 删除表未释放内存”，在使用MySQL的 delete 操作来删除数据文件后，占用的空间没有被完全释放，经过长时间的运行被占用的空间就会积累到不可忽视的地步。

经过研究发现，其实 delete 操作只是指挥系统只是删除了数据文件对应的标识位，而具体文件依旧存在于数据库中，导致空间仍然被占用，只有新写入数据时才会进行覆盖，且可能留下数据碎片，无法完全释放空间，具体如表X.7。

度量元名称	度量元类型	失信原因	失信证据
删除表未释放内存	增长型	delete 操作只是指挥系统只是删除了数据文件对应的标识位，而具体文件依旧存在于数据库中，导致空间仍然被占用，只有新写入数据时才会进行覆盖，且可能留下数据碎片，无法完全释放空间。	实际使用内存与程序固定占用内存之比

突变型度量元：是指某些指标随时间增长至临界点时会发生突变的软件项目 元素。

例如“MySQLBug34335 数值溢出”，数据库中自动增长的编号在运行一段时间后大于数据类型上限，软件失效，具体如表X.8。

度量元名称	度量元类型	失信原因	失信证据
数值溢出	突变型	当自动增长的数据大于当前数据类型上限导致数据失效	达到临界值平均所需的时间

通过对提取的失信证据按两种分类方法同时进行研究，去除重复、冗余的 部分，设计出最终的度量元模型，与定义的六元组对应，部分度量元展示如下 表X.11，完整的度量元表格。

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标（V-I 递减）	风险值 <i>r(t)</i>	出处
1	浮点数寄存器误差	增长型	24 位的固定小数点寄存器会舍去 25位以下的数字，导致难以避免的误差	ARU	48 小时后误差对预期结果的影响	结果完全不可信	10	爱国者导弹事件
						影响较大	9	
						有影响结果不可靠	7	
						影响结果小，可接受	4	
						无影响	1	

度量元设计

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标 (V-I 递减)	风险值 $r(t)$	出处
2	内存需求过大	触发型	程序所需内存大于系统可用内存可能导致预期功能无法实现或系统效率降低	MEM	程序运行需要占用的内存空间占总内存的比	88%-100%	10	Linux2425, Linux7536
						59%-88%	9	
						40%-59%	7	
						20%-40%	4	
						0-20%	1	
3	索引节点爆满	突变型	索引节点用尽时系统无法再创建新目录或文件	STO	程序所需索引节点占系统拥有索引节点的比例	88%-100%	10	Linux3431
						59%-88%	9	
						40%-59%	7	
						20%-40%	4	
						0-20%	1	
					达到临界值平均所需的时间(h)	0-10	10	
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标（V-I 递减）	风险值 $r(t)$	出处
4	权限管理错误	触发型	系统无法访问自己本该被授权的资源或访问自己未被授权的资源	MEM	操作相关文件状态不满足规定的要求的比例	87%-100%	10	Linux11935
						59%-87%	9	
						20%-59%	7	
						0-20%	4	
						0	1	
5	缓冲区溢出	突变型	缓冲区本身的容量有限，在写入数据时可能一次超出了上限，造成溢出甚至覆盖其他数据	MEM	达到临界值平均所需的时间（h）	0-10	10	Linux5137, Linux5711, AXIS1270
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标（V-I 递减）	风险值 $r(t)$	出处
6	数值溢出	突变型	当自动增长的数据大于当前数据类型上限导致数据失效	NUM	达到临界值平均所需的时间（h）	0-10	10	MySQL 34335
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
7	死锁	触发型	多个进程因竞争共享资源而处于永远等待的状态	LOG	触发死锁的概率	87%-100%	10	MySQL 52814, AXIS1369
						59%-87%	9	
						20%-59%	7	
						0-20%	4	
						0	1	

一个度量元可以由多个失信证据支撑，对应多个风险值，该度量元最终风险值取其中的最大值，因为只要有一个失信证据风险值偏大就说明该度量元导致软件进入预期之外运行状态数较多，发生软件老化的可能性就大。

表格中指标的 设计源于超过两百位编程人员的问卷调查，在对问卷结果进行整理汇总的基础上结合实际情况，最终确定为上述数值，具体的问卷情况可查阅附录。

文中最终 设计了共计 30 个基于软件老化的失信证据度量元，后续可扩充取材范围进行添加，本章仅提供设计思路以供参考。

度量元设计(MEM15)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值 $r(t)$	出处
M1	内存需求过大	触发型	程序所需内存大于系统可用内存可能导致预期功能无法实现或系统效率降低	MEM	程序运行需要占用的内存空间占总内存的比	88%–100%	10	Linux2425, Linux7536
						59%–88%	9	
						40%–59%	7	
						20%–40%	4	
						0–20%	1	
M2	资源占用	触发型	高优先级或其他资源长时间占用资源导致资源无法释放	MEM	长时间占用的内存大小占总内存的比	88%–100%	10	Linux5144, Linux9468, Linux13293, MySQL49535, MySQL48993, Apache12320, Apache27751
						59%–88%	9	
						40%–59%	7	
						20%–40%	4	
						0–20%	1	
					对系统性能产生影响平均所需的时间	00–10	10	
						10–30	9	
						30–60	7	
						60–110	4	
						>110	1	
M3	权限管理错误	触发型	系统无法访问自己本该被授权的资源或访问自己未被授权的资源	MEM	操作相关文件状态不满足规定要求的比例	87%–100%	10	Linux11935
						59%–87%	9	
						20%–59%	7	
						0–20%	4	
						0	1	
M4	程序异常繁殖	增长型	程序自我复制超出预期，占用资源过多	MEM	规定时间内程序额外占用内存大小	88%–100%	10	Linux34622
						58%–88%	9	
						40%–58%	7	
						19%–40%	4	
						0–19%	1	
					对系统性能产生影响平均所需的时间	00–10	10	
						10–30	9	
						30–60	7	
						60–110	4	
						>110	1	

度量元设计(MEM15)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值r(t)	出处
M5	空指针相关问题	触发型	当一个对象不存在时又调用其方法会产生异常， 当访问或修改一个对象不存在的字段时会产生异常	MEM	指针调用不当数目	>11	10	Linux1209, Linux6043, AXIS1248
						6-11	9	
						3-6	7	
						1-3	4	
						0	1	
M6	内存越界	触发型	当内存输入超出了预分配的空间大小，就会覆盖该空间之后的一段存储区域，导致系统异常	MEM	出现大于预分配空间 大小的输入的概率	30%-100%	10	Linux6114, MySQL38191, “火星全球勘测者”号事件
						15%-30%	9	
						5%-15%	7	
						0-5%	4	
						0	1	

M7	缓冲区溢出	突变型	向缓冲区填充数据超其本身的容量，而导致数据溢出到被分配空间之外的内存空间，溢出的数据覆盖其他内存数据	MEM	达到临界值平均 所需的时间(h)	00-10	10	Linux5137, Linux5711, AXIS1270
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	

度量元设计(MEM15)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标 (V-I递减)	风险值 $r(t)$	出处
M8	删除表未释放内存	增长型	delete 操作只是指挥系统只是删除了数据文件对应的标识位，而具体文件依旧存在于数据库中，导致空间仍然被占用，只有新写入数据时才会进行覆盖，且可能留下数据碎片，无法完全释放空间。	MEM	实际使用内存与程序 固定占用内存之比	88%-100%	10	MySQL46656
						58%-88%	9	
						40%-58%	7	
						19%-40%	4	
						0-19%	1	
					对系统性能产生影响 平均所需的时间	0-10	10	
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
M9	引用未清空	增长型	使用完的引用由于引用未及时清空导致无法自动回收内存	MEM	引用未清理占比	55%-100%	10	AXIS1202
						30%-55%	9	
						15%-30%	7	
						5%-15%	4	
						0-5%	1	
M10	循环过度	突变型	循环过度导致运行时间过程或者内存被全部占用	MEM	循环复杂度	>100	10	Linux5284， 2003 年美国东北部 大停电事件
						50-100	9	
						30-50	7	
						10-30	4	
						1-10	1	

度量元设计(MEM15)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标 (V-I递减)	风险值 $r(t)$	出处
M11	异常终止	增长型	异常终止后部分操作未执行，导致程序程序崩溃	MEM	未释放内存大小	>200m 150m-200m 100m-150m 50m-100m 0-50m	10 9 7 4 1	MySQL45989 , MySQL33247
					对系统运行产生影响 平均所需的时间	0-10 10-30 30-60 60-110 >110	10 9 7 4 1	
M12	回滚机制问题	增长型	回滚机制设计不完善，导致资源浪费	MEM	未释放内存大小	>200m 150m-200m 100m-150m 50m-100m 0-50m	10 9 7 4 1	MySQL32709
					对系统运行产生影响 平均所需的时间	0-10 10-30 30-60 60-110 >110	10 9 7 4 1	

度量元设计(MEM15)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值r(t)	出处
M13	僵尸进程过多	增长型	大量僵尸进程占用系统资源，影响性能	MEM	占用的内存大小	>200m	10	Apache9168
						150m-200m	9	
						100m-150m	7	
						50m-100m	4	
						0-50m	1	
					对系统运行产生影响 平均所需的时间	0-10	10	
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
M14	高并发缓存问题	触发型	缓存一致性、缓存并发、缓存穿透、缓存的雪崩现象	MEM	一次性操作数据的规模 占可用资源的比	87%-100%	10	Linux11100,
						59%-87%	9	Linux11377,
						40%-59%	7	Apache9708,
						20%-40%	4	Apache29962
						0-20%	1	
M15	内存泄漏	增长型	用动态存储分配函数动态开辟的空间，在使用完毕后未释放，结果导致一直占据该内存单元，直到程序结束	MEM	平均每次使用软件增加的额外内存占总内存的百分比	55%-100%	10	MySQL56709 Apache14098 , Apache24991 , Apache26562 , Apache27106 , AXIS1423 , AXIS2278
						30%-55%	9	
						15%-30%	7	
						5%-15%	4	
						0-5%	1	

度量元设计(STO4)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值 $r(t)$	出处
S1	磁盘空间泄露	增长型	磁盘空间泄漏导致 磁盘空间占用、 虚拟内存不足、 系统效率降低	STO	软件占用预期之外的磁 盘空间占总空间的比	87%-100%	10	Linux3171
						59%-87%	9	
						40%-59%	7	
						20%-40%	4	
						0-20%	1	
					对系统性能产生影 响平均所需的时间	00-10	10	
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
S2	索引节点爆满	突变型	索引节点用尽时 系统无法再创建 新目录或文件	STO	程序所需索引节点数 占系统拥有索引节点 数的比例	88%-100%	10	Linux3431
						59%-88%	9	
						40%-59%	7	
						20%-40%	4	
						0-20%	1	
					达到临界值平均 所需的时间(h)	00-10	10	
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
S3	数据插入问题	触发型	数据插入方式错误 或插入 位置错误, 导致缓冲区或 者磁盘空间被填满	STO	触发插入问题的概率	88%-100%	10	MySQL52964
						58%-88%	9	
						19%-58%	7	
						0-19%	4	
						0	1	

度量元设计(STO4)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标 (V-I递减)	风险值 $r(t)$	出处
S4	写入失败	突变型	日志文件过多未及时清理，导致空间被占满，新日志无法写入	STO	达到临界值平均所需的时间（h）	0-10	10	Apache13511
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	

度量元设计(LOG7)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值 $r(t)$	出处
L1	嵌套字功能异常	触发型	发生套接字超时或	LOG	嵌套字功能异常的概率	86%-100%	10	Linux32832, AXIS769
			其他嵌套字功能			59%-86%	9	
			异常时线程阻塞			20%-59%	7	
						0-20%	4	
						0	1	
L2	未刷新缓存	触发型	缓存未及时刷新导致 结果偏离预期	LOG	截断后缓存未刷新	1	10	MySQL40386
						0	1	
L3	死锁	触发型	多个进程因竞争共享资源而处于永远等待的状态	LOG	触发死锁的概率	87%-100%	10	MySQL52814, AXIS1369
						59%-87%	9	
						20%-59%	7	
						0-20%	4	
						0	1	
L4	无效更新	触发型	重复自动更新变化 内容, 或更新索引时未发生变化 导致资源浪费	LOG	无效更新的占比	87%-100%	10	MySQL56340
						59%-87%	9	
						40%-59%	7	
						20%-40%	4	
						0-20%	1	
L5	临时文件未及时清理	增长型	临时文件未及时清理, 导致占用过多内存	LOG	占用的内存大小	>200m	10	MySQL40013
						150m-200m	9	
						100m-150m	7	
						50m-100m	4	
						0m-50m	1	

度量元设计(LOG7)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标 (V-I递减)	风险值 $r(t)$	出处
L6	适配器选择错误	触发型	适配器模式中接口 选择不当	LOG	适配器选择不当	1	10	Linux7718
						0	1	
L7	嵌套字用尽	突变型	嵌套字资源用尽导致无嵌套字可用	LOG	达到临界值平均所需的时间 (h)	0-10	10	AXIS238
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	

度量元设计(NUM2)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值 $r(t)$	出处
N1	数值溢出	突变型	当自动增长的数据大于当前数据类型上限导致数据失效	NUM	达到临界值平均所需的时间(h)	0-10	10	MySQL34335
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
N2	舍入误差累计	增长型	运算中舍入操作经过较长时间的累积导致误差巨大	NUM	误差对结果产生不可忽视的影响所需的时间	一个月以内	10	Linux20882, 温哥华证券交易所事件
						一个月至一年	9	
						一至十年	7	
						十年以内	4	
						一直无影响	1	

度量元设计(ARU2)

ID	度量元名称	度量元类型	失信原因	软件老化原因分类	失信证据	风险等级对应指标(V-I递减)	风险值r(t)	出处
A1	浮点数寄存器误差	增长型	24 位的固定小数点寄存器会舍去 25 位以下的数字，导致难以避免的误差	ARU	48 小时后误差对 预期结果的影响	结果完全不可信	10	爱国者导弹事件
						影响较大	9	
						有影响结果不可靠	7	
						影响小结果可接受	4	
						无影响	1	
A2	系统功能异常	触发型	仅指系统连续运行一段时间后，系统性能下降或功能出现异常，但具体原因无从考证的情况	ARU	出现异常平均 所需的时间	0-10	10	Linux3134, Linux5870, Linux7959,
						10-30	9	
						30-60	7	
						60-110	4	
						>110	1	
					发生功能异常 的概率	30%-100%	10	
						15%-30%	9	
						5%-15%	7	
						0-5%	4	
						0	1	

第11章 软件老化可信度量

11.1 软件老化

11.2 老化类别

11.3 度量元模型

11.4 可信计算框架



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

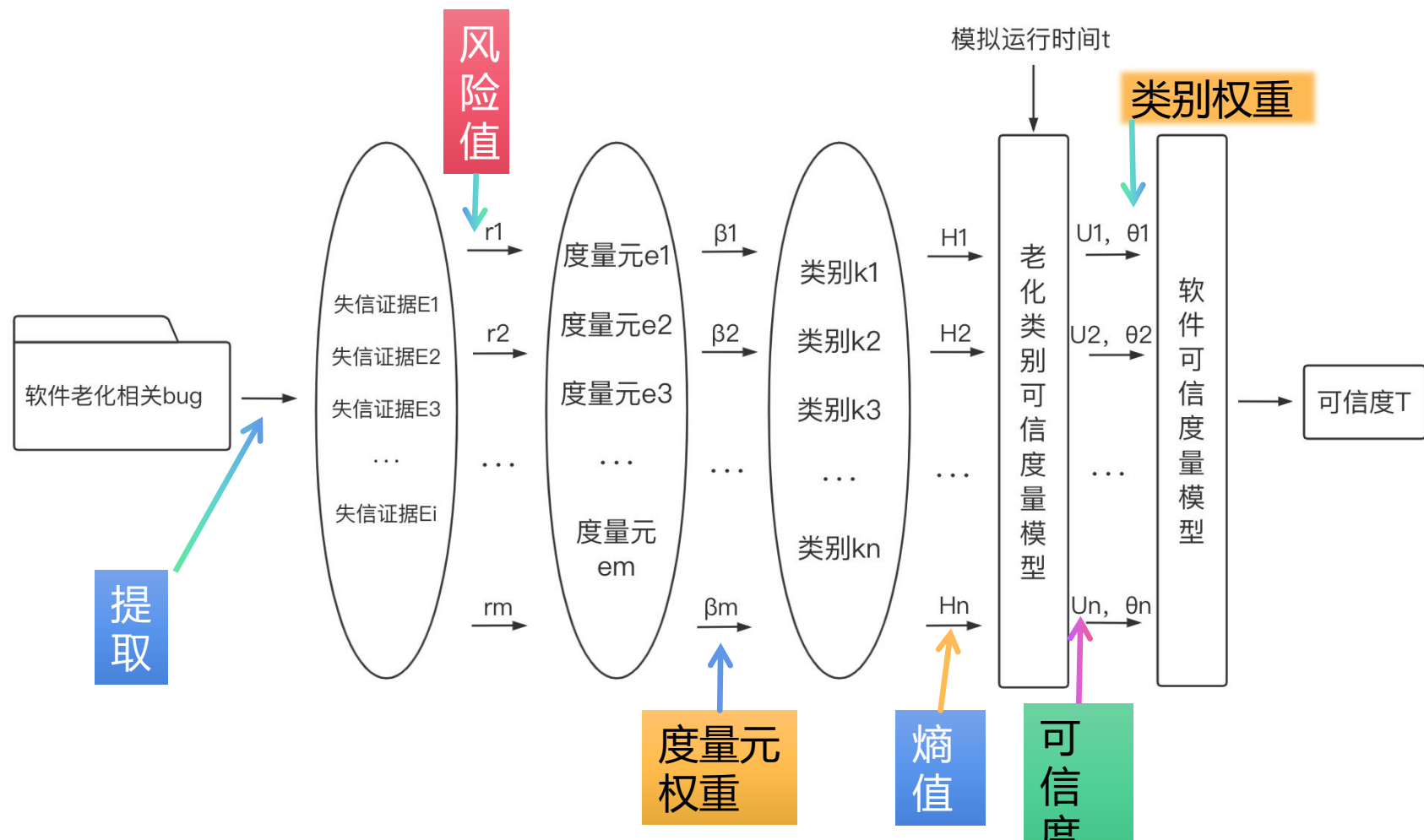
可信计算框架

1. 从研究软件老化相关 bug 出发，首先从 bug 中提取失信证据、设计度量元，得到了基于软件老化的失信证据度量元。
2. 将度量元按导致软件老化的原因分组，分别计算各类别的可信度。
3. 将分类可信度进行汇总，得到综合的软件可信度量模型。
4. 以各类别包含的度量元个数以及度量元对应的实例数为数据，采用 Brassard 优先次序全面分析标准法计算权重。
5. 输入模拟运行时间，得到预期时间点的软件可信度。
6. 从软件老化角度提出了从数据采集到可信性度量模型的完整框架，如图X.1所示

可信计算框架

从软件老化角度提出了从数据采集到可信性度量模型的完整框架，如图X.1所示

其中， $E1 \dots Ei$ 表示从搜集到的软件老化相关 bug 中提取出的 i 个失信证据， m 表示基于软件老化的失信证据度量元个数，用 $e1 \dots em$ 表示， $r1 \dots rm$ 表示具体度量元对应的风险值， $\beta1 \dots \beta m$ 表示具体度量元对应的权重， n 表示度量元涉及 n 种导致软件老化的原因， $k1 \dots kn$ 表示度量元按软件老化的原因分成了 n 个类别， $H1 \dots Hn$ 表示具体软件老化原因分类的熵， $U1 \dots Un$ 表示具体类别对应的可信度， $\theta1 \dots \theta n$ 表示具体类别对应的权重， t 表示软件的运行时长， T 表示预测软件在 t 时刻的可信度。



度量元权重计算

β_i 为第 i 个度量元的权重，权重采用统一的计算方式。权重 β_i 代表了各度量元与同类型度量元相比的相对重要程度，数值越大说明该度量元对分类可信度的影响也就越大。这里用一个类别 ARU 为例子，每个类别都有其各自对应的一组 β_i 。

类别 ARU 共有 2 个度量元：A1 浮点数寄存器误差(1)，A2 系统功能异常(3)。名称后面的括号内是该度量元包含的实例数目，例如“浮点数寄存器误差(1)”代表了这个度量元对应了一份问题报告：爱国者导弹事件，而系统功能异常(3)代表 3 个问题报告，分别是 Linux3134，Linux5870，Linux7959。

度量元权重计算

类别ARU共有 2个度量元：A1浮点数寄存器误差(1), A2系统功能异常(3).名称后面的括号内是该度量元包含的实例数目，例如“浮点数寄存器误差(1)”代表了这个度量元对应了一份问题报告：爱国者导弹事件，而系统功能异常(3)代表3个问题报告，分别是 Linux3134, Linux5870, Linux7959。

这里的数据代表了该度量元与其他度量元重要程度的比较结果，通过每个度量元在统计数据中出现次数之比反应各度量元重要性的不同，该度量元对应的错误报告越多，说明该度量元对应的软件老化问题出现的越频繁，那这个度量元就越重要，对分类可信度的影响也就越大。

构造一个矩阵：

类别ARU度量元权重

	A1 (1) 浮点数寄存器误差	A2 (3) 系统功能异常	行总计	(权重 β_i)
A1 (1)		1/3	1/3	0.10
A2 (3)	3/1		3	0.90
列总计	3	1/3	10/3	

$$\beta_1=(1/3)/(10/3)=0.10, \quad \beta_2=(3)/(10/3)=0.90.$$

由于度量对象可能是一个复杂的软件，涉及众多可能的软件老化原因和状态参数，因此采用在处理高度复杂问题时效果出众的优先顺序矩阵来解决。

Brassard 确定权重 β_i 的计算方法

度量元权重计算

类别LOG共有 7个度量元：嵌套字功能异常（2）、适配器选择错误（1）、未刷新缓存（1）、死锁（2）、无效更新（1）、临时文件（1）、嵌套字用尽（1）。名称后面的括号内是该度量元包含的实例数目，例如“死锁（2）”代表了死锁这个度量元对应了两份问题报告，分别是 MySQL- Bug52814 和 AXISBug1369。

度量元权重计算



0.002分配给哪两个度量元？，取四位小数

类别LOG度量元权重

	嵌套字功能异常 (2)	适配器选择错误 (1)	未刷新缓存 (1)	死锁 (2)	无效更新 (1)	临时文件 (1)	嵌套字用尽 (1)	行总计	权重；总计所占百分比 β_i
嵌套字功能异常 (2)		2(=2/1)	2	1	2	2	2	11	0.234=11/47 0.23404255319
适配器选择错误 (1)	1/2(=1/2)		1	1/2	1	1	1	5	0.106=5/47 (0.1063829) (0.1064)
未刷新缓存 (1)	1/2	1		1/2	1	1	1	5	0.106=5/47 (0.1064)
死锁 (2)	1	2	2		2	2	2	11	0.2340=11/47
无效更新 (1)	1/2	1	1	1/2		1	1	5	0.106=5/47 (0.1064)
临时文件 (1)	1/2	1	1	1/2	1		1	5	0.106=5/47 (0.1064)
嵌套字用尽 (1)	1/2	1	1	1/2	1	1		5	0.106=5/47 (0.1064)
列总计	3.5	8	8	3.5	8	8	8	47	0.998(1)

类别LOG度量元权重

类别LOG有 7个度量元的权重为：

1. 嵌套字功能异常 (β_1) = 0.2340、
2. 适配器选择错误 (β_2) = 0.1064、
3. 未刷新缓存 (β_3) = 0.1064、
4. 死锁 (β_4) = 0.2340、
5. 无效更新 (β_5) = 0.1064、
6. 临时文件 (β_6) = 0.1064、
7. 嵌套字用尽 (β_7) = 0.1064。

度量元权重计算

类别MEM度量元权重

[illegible]

类别STO度量元权重

	S1 (1) 内磁盘空间泄露	S2 (1) 索引节点爆满	S3 (1) 数据插入问题	S4 (1) 写入失败	行总计	(权重 β_i)
S1 (1)		1/1	1/1	1/1	3	0.25
S2 (1)	1/1		1/1	1/1	3	0.25
S3 (1)	1/1	1/1		1/1	3	0.25
S4 (1)	1/1	1/1	1/1		3	0.25
列总计	3	3	3		12	

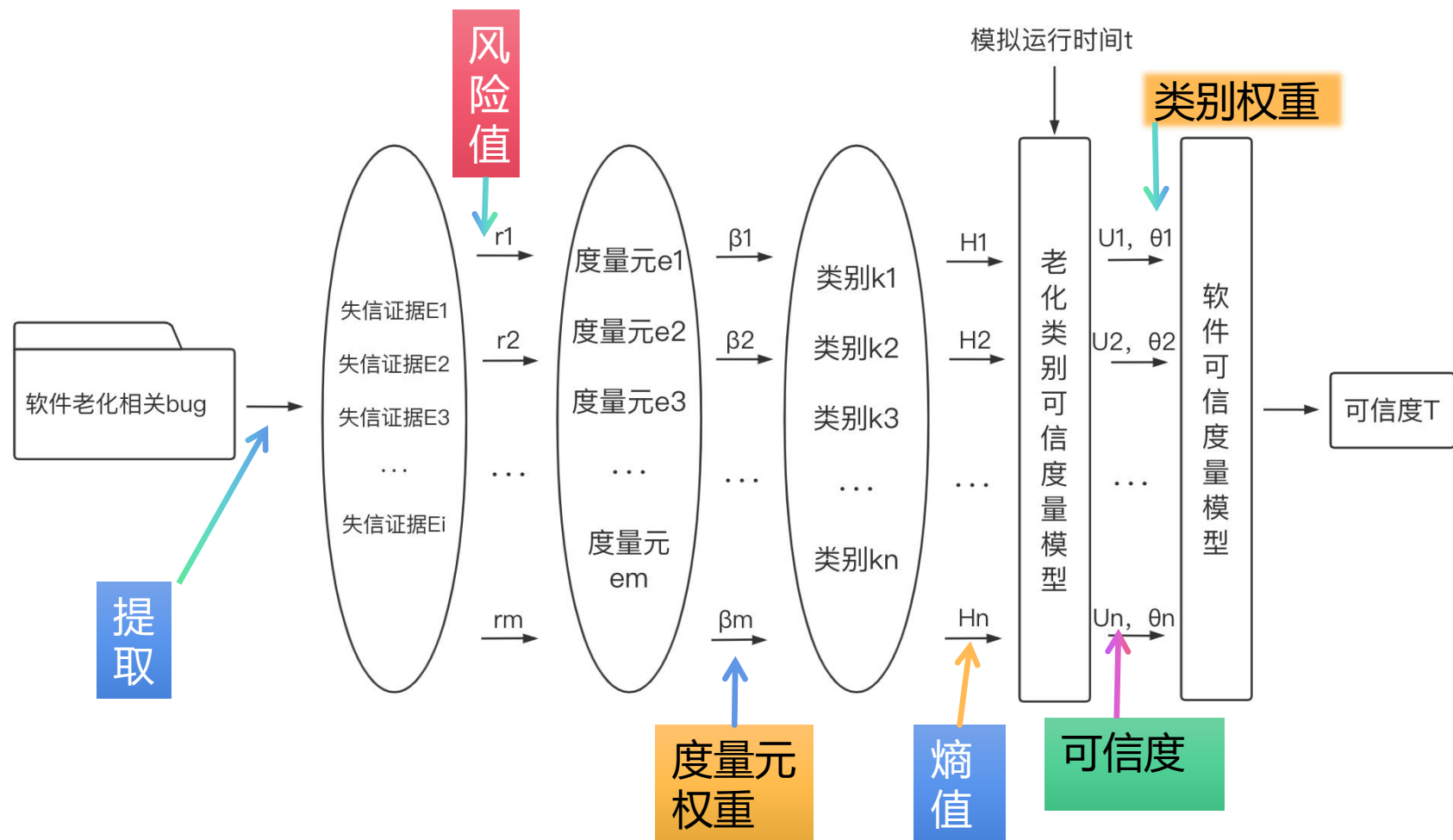
类别NUM度量元权重

	N1 (1) 数值溢出	N2 (2) 舍入误差累计	行总计	(权重 β_i)
N1 (1)		1/2	1/2	0.20
N2 (2)	2/1		2	0.80
列总计	2	1/2	5/2	

可信计算框架

从软件老化角度提出了从数据采集到可信性度量模型的完整框架，如图X.1所示

其中， $E1 \dots Ei$ 表示从搜集到的软件老化相关 bug 中提取出的 i 个失信证据， m 表示基于软件老化的失信证据度量元个数，用 $e1 \dots em$ 表示， $r1 \dots rm$ 表示具体度量元对应的风险值， $\beta1 \dots \beta m$ 表示具体度量元对应的权重， n 表示度量元涉及 n 种导致软件老化的原因， $k1 \dots kn$ 表示度量元按软件老化的原因分成了 n 个类别， $H1 \dots Hn$ 表示具体软件老化原因分类的熵， $U1 \dots Un$ 表示具体类别对应的可信度， $\theta1 \dots \theta n$ 表示具体类别对应的权重， t 表示软件的运行时长， T 表示预测软件在 t 时刻的可信度。



度量元熵值计算H

基于软件老化的失信证据**度量元的熵**是指仅在单一基于软件老化的失信证据度量元影响下，表示软件运行状态混乱程度的量。

这里运行状态的混乱程度考虑预期内运行状态和风险状态，因为预期内运行状态是固定的，风险状态越多软件运行中偏离预期的可能就越大，系统就越混乱，熵就会增大。

熵值的计算公式如下：

$$S = \log_{10} r(t)$$

其中 $r(t)$ 为度量元对应的最大风险值，**度量元可能有两个风险值。**

类别熵值计算H

软件老化原因分类的熵是指在由同一种软件老化原因导致的基于软件老化的失信证据度量元影响下，表示软件运行状态混乱程度的量，是软件运行时间的函数。

软件老化原因类别熵值(在时刻 t)的计算公式表示为：

$$H(t) = \sum_{i=1}^n \beta_i \log_{10} r_i(t)$$

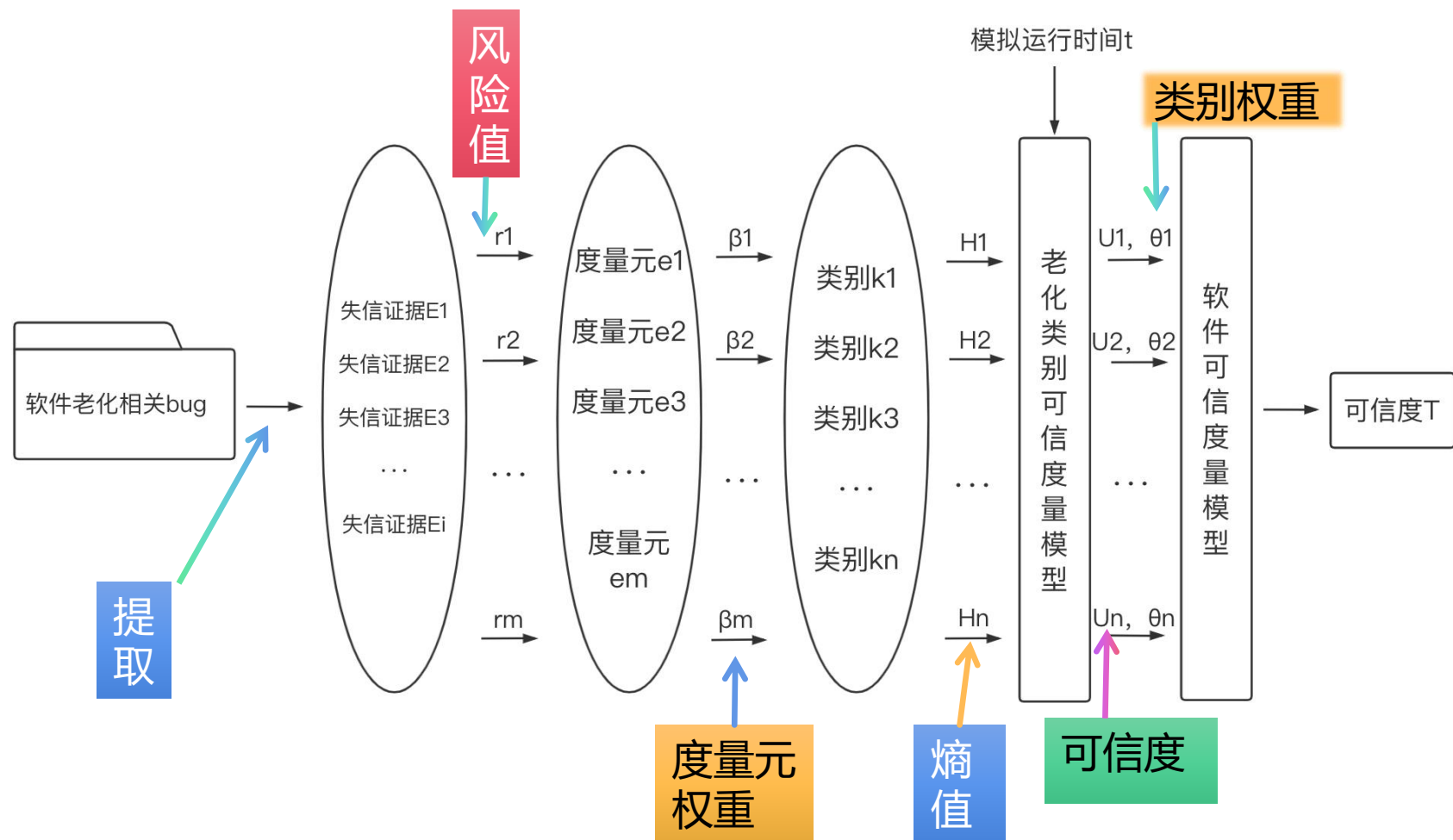
其中 β_i 为第 i 个度量元的权重，权重采用Brassard的计算方式， $r_i(t)$ 为第 i 个度量元的最大风险值。

单一类型的熵值越高，代表了该类型度量元可能引发软件老化的风险越大，软件可信度随时间降低的速度也就越快。

可信计算框架

从软件老化角度提出了从数据采集到可信性度量模型的完整框架，如图X.1所示

其中， $E1 \dots Ei$ 表示从搜集到的软件老化相关 bug 中提取出的 i 个失信证据， m 表示基于软件老化的失信证据度量元个数，用 $e1 \dots em$ 表示， $r1 \dots rm$ 表示具体度量元对应的风险值， $\beta1 \dots \beta m$ 表示具体度量元对应的权重， n 表示度量元涉及 n 种导致软件老化的原因， $k1 \dots kn$ 表示度量元按软件老化的原因分成了 n 个类别， $H1 \dots Hn$ 表示具体软件老化原因分类的熵， $U1 \dots Un$ 表示具体类别对应的可信度， $\theta1 \dots \theta n$ 表示具体类别对应的权重， t 表示软件的运行时长， T 表示预测软件在 t 时刻的可信度。



软件老化原因类别权重

- 所有涉及的软件老化原因类型，分别为 MEM (15) , STO (4) , LOG (7) , NUM (2) , ARU (2) 表示五种不同的分类。
- 每个分类后括号内的数字代表了该分类中包含的度量元数目，MEM (15) 意味着总计有 15 个与内存管理有关的度量元。
- 这里表中的数据代表了该类型与其他类型重要程度的比较结果，通过各分类包含的度量元数目之比反应各分类重要性的不同，包含的度量元数目越多，该类型的软件老化原因相比其他类型越重要，对可信度的影响也就越大。
- 而每一行的行总计即为单个比较得分的总和，括号内是这些总和相对所有分类的比例权重，也就是要求的 θ_i 。
- 权重 θ_i 代表了各软件老化原因的相对重要程度，数值越大说明该类型的软件老化原因对整体软件可信度的影响也就越大。

软件老化原因类别权重

	MEM (15) 内存管理	STO (4) 存储空间	LOG (7) 逻辑资源	NUM (2) 误差累积	ARU (2) 未知原因	行总计	(权重; 总计所占百分比 θ_i)
MEM (15)		15/4	15/7	15/2	15/2	585/28= 20.893	0.539
STO (4)	4/15		4/7	2	2	508/105= 4.838	0.125
LOG (7)	7/15	7/4		7/2	7/2	553/60= 9.217	0.238
NUM (2)	2/15	1/2	2/7		1	403/210= 1.919	0.049
ARU (2)	2/15	1/2	2/7	1		403/210= 1.919	0.049
列总计	1	13/2	23/7	14	14	543/14 38.786	1.000

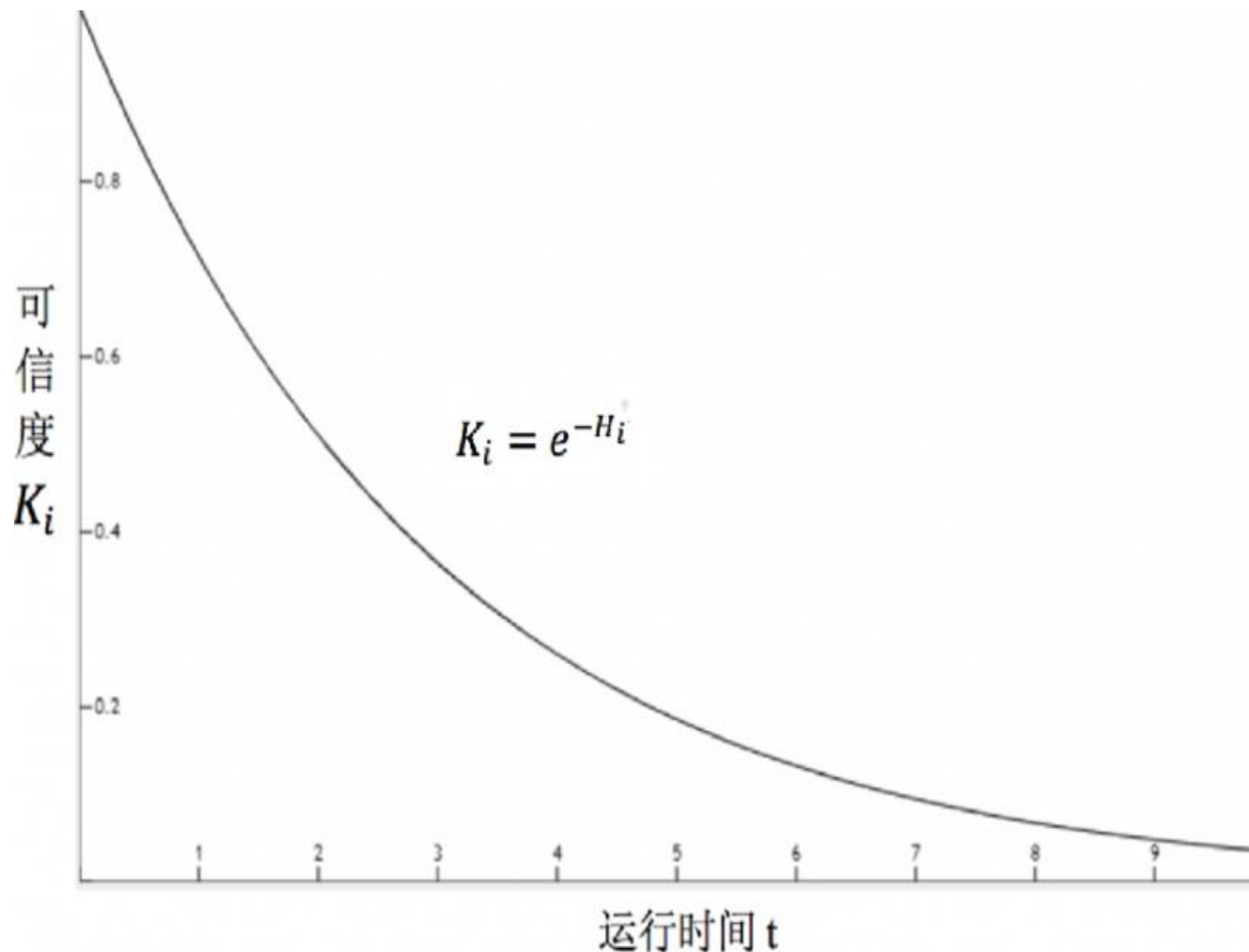
类别可信度计算

设计得到老化类别可信度量模型，各类别可信度与时间 t 的关系函数为：

$$K_i(t) = e^{-H_i(t)}$$

其中， $K_i(t)$ 为第 i 个类别在时刻 t 的可信度， $H_i(t)$ 为第 i 个软件老化类别在时刻 t 的熵， t 为软件运行时间。

可得 $K_i(t)$ 随着 t 的变化曲线大致如右图所示：



类别可信度计算

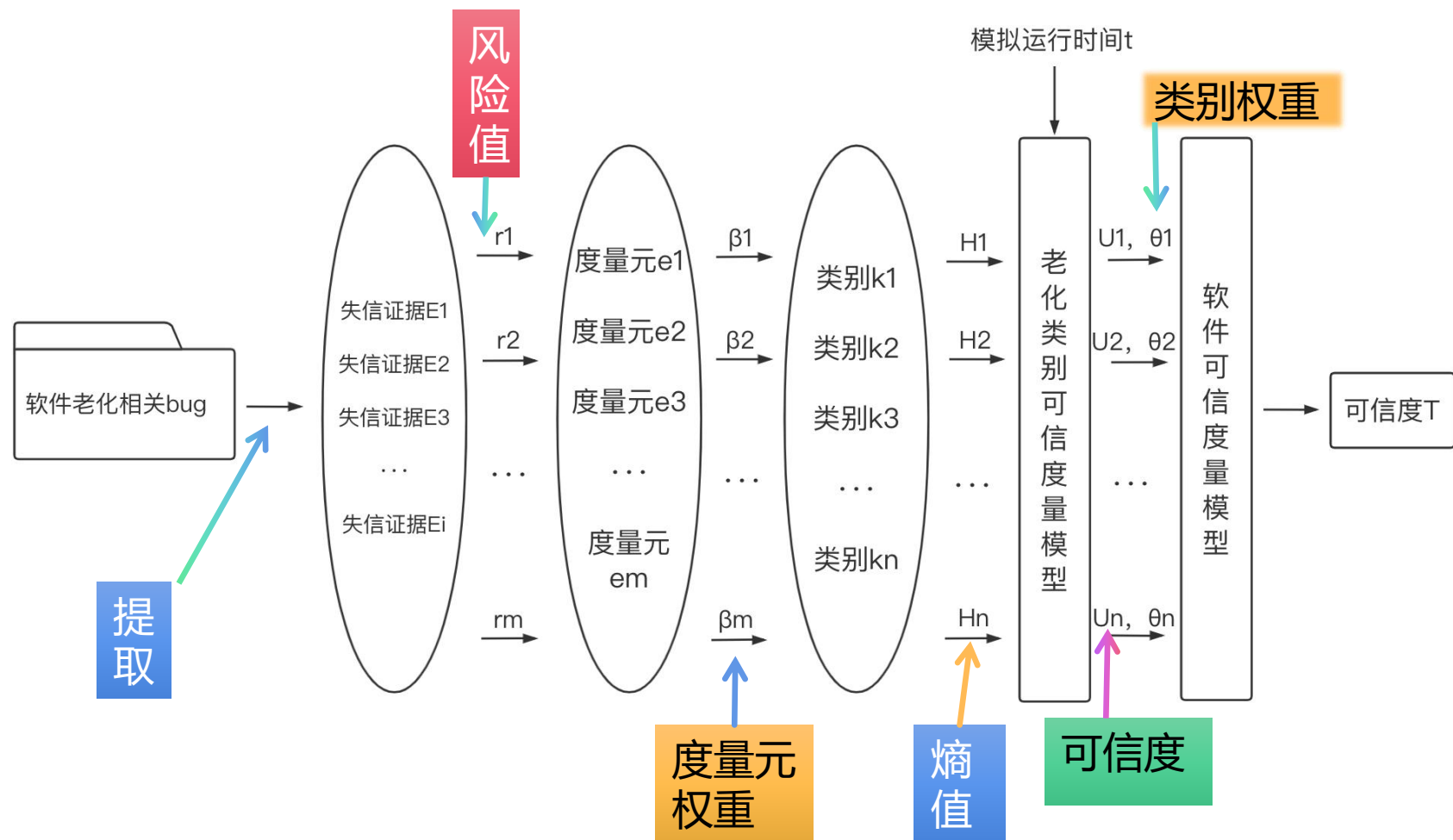
文中提升某一分类的权重，表示该类型的软件老化原因对软件可信度的影响应增大，因此需要将各**类别 (i)** 可信度对软件可信度分类可信度的值域 $[0, 1]$ 进行优化，也采用十分制，值域为 $[1, 10]$ 。因此**i类别** 可信度的十分制表示为：

$$U_i(t) = \begin{cases} 10e^{-H_i(t)}, & 0.1 \leq e^{-H_i(t)} \leq 1 \\ 1, & 0 \leq e^{-H_i(t)} < 0.1 \end{cases}$$

可信计算框架

从软件老化角度提出了从数据采集到可信性度量模型的完整框架，如图X.1所示

其中， $E1 \dots Ei$ 表示从搜集到的软件老化相关 bug 中提取出的 i 个失信证据， m 表示基于软件老化的失信证据度量元个数，用 $e1 \dots em$ 表示， $r1 \dots rm$ 表示具体度量元对应的风险值， $\beta1 \dots \beta m$ 表示具体度量元对应的权重， n 表示度量元涉及 n 种导致软件老化的原因， $k1 \dots kn$ 表示度量元按软件老化的原因分成了 n 个类别， $H1 \dots Hn$ 表示具体软件老化原因分类的熵， $U1 \dots Un$ 表示具体类别对应的可信度， $\theta1 \dots \theta n$ 表示具体类别对应的权重， t 表示软件的运行时长， T 表示预测软件在 t 时刻的可信度。



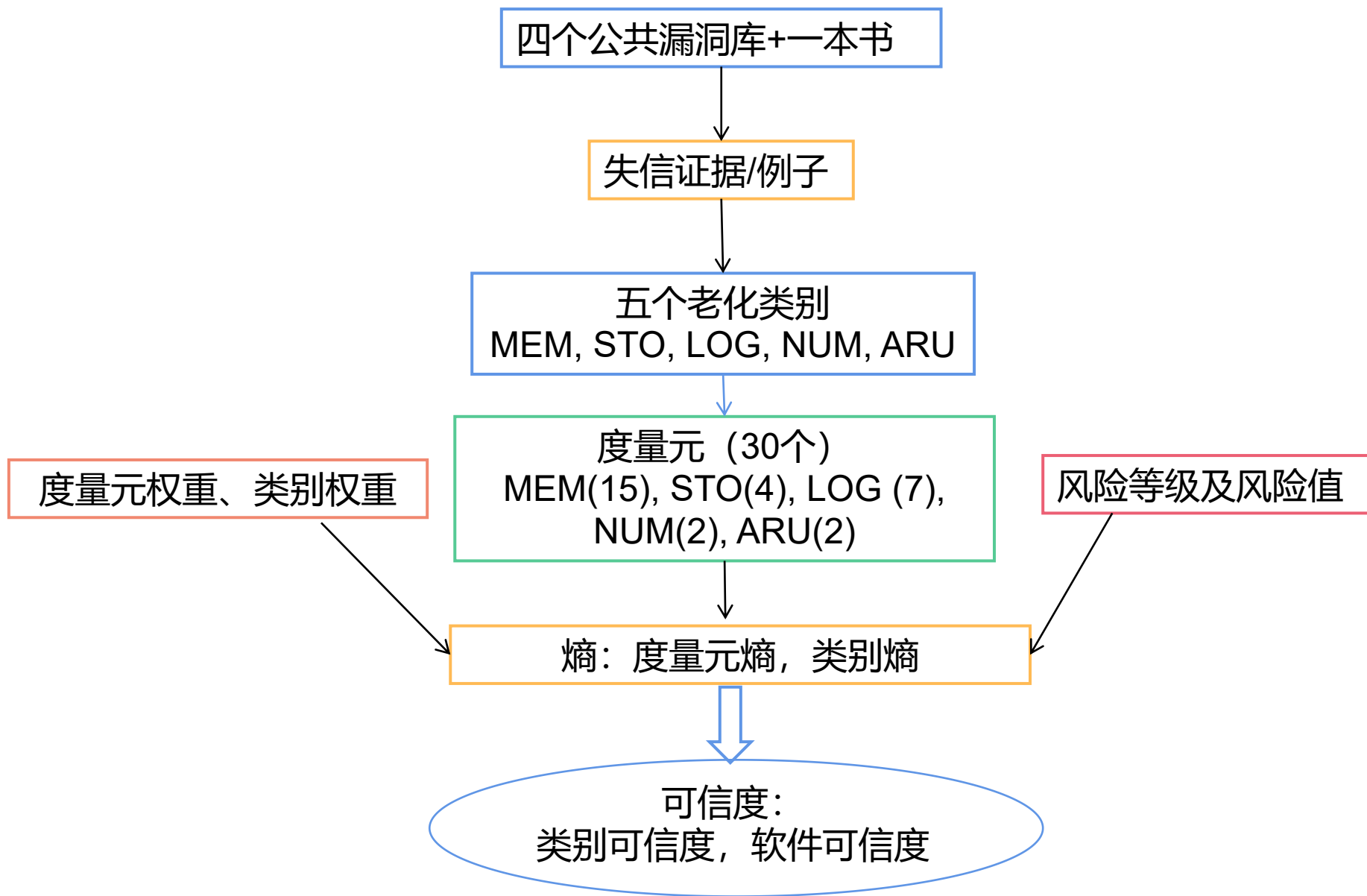
软件可信度计算

将各分类可信度汇总成综合的软件可信度的软件可信性计算模型如下：

$$T(t) = \begin{cases} \prod_{i=1}^n U_i(t)^{\theta_i}, & 1 \leq U_i(t) \leq 10 \\ \sum_{i=1}^n \theta_i = 1, & 0 \leq \theta_i \leq 1 \end{cases}$$

其中， θ_i ($i=1, \dots, n$) 为各类别的权重。

总结



例子（风险值）



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

[illegible]

计算四个类别的熵和可信度，以及软件可信度

	时间	MEM	STO	LOG	NUM	ARU
熵	t=0	0	0	0	0	0
	t=10	0.746	0.872	0.852	0.867	0.943
可信度	t=0	10	10	10	10	10
	t=10	4.743	4.180	4.266	4.202	3.893

时间	软件可信度
t=0	10
t=10	4.482

作业（风险值）



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

[illegible]

作业：编程或使用课程提供的Python



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

输入：类别，以及类别中每个度量元的含例子个数，

输出：在时刻 $t=0$ 和 $t=10$ 情况下，四个类别的熵和可信度，以及软件可信度。

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

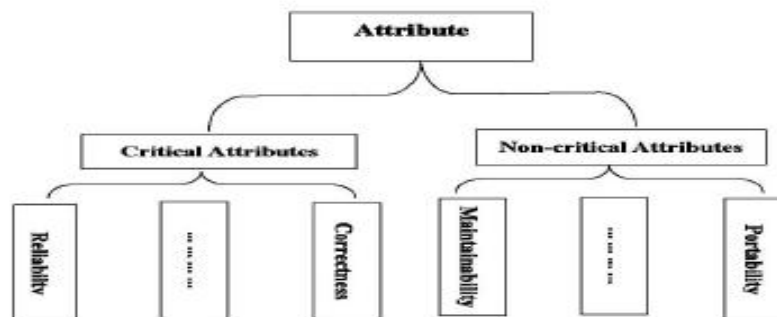
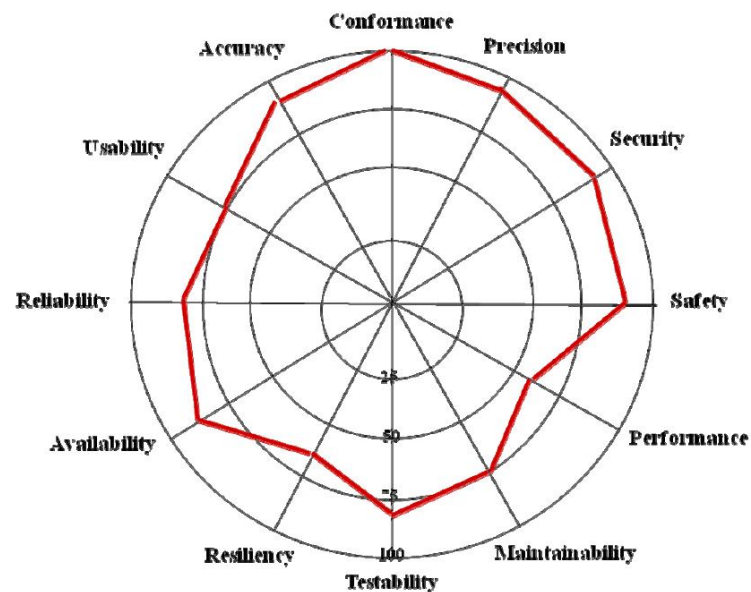
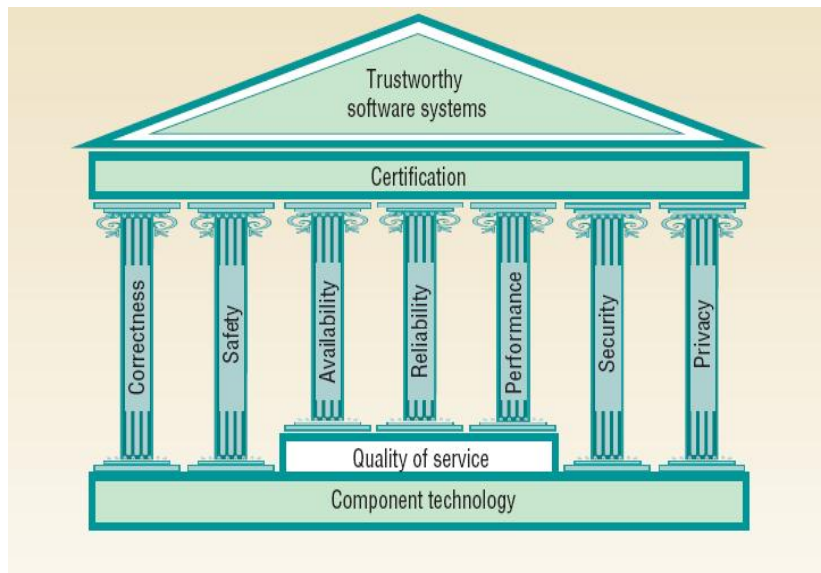


Fig. 1. The Hierarchical Model for Trustworthy Metrics of Software

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

度量计算模型

$$\begin{cases} \alpha_1 + \alpha_2 + \cdots + \alpha_n = \sum_{i=1}^n \alpha_i = 1 & 0 \leq \alpha_i \leq 1 \\ T = y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_n^{\alpha_n} = \prod_{i=1}^n y_i^{\alpha_i} & 0 \leq y_i \leq 1 \end{cases}$$

T 为软件可信度, $0 \leq T \leq 1$

y_i 为第 i 个属性的可信度

α_i 为第 i 个属性的权重值

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

每个可信属性的分布函数是什么？

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

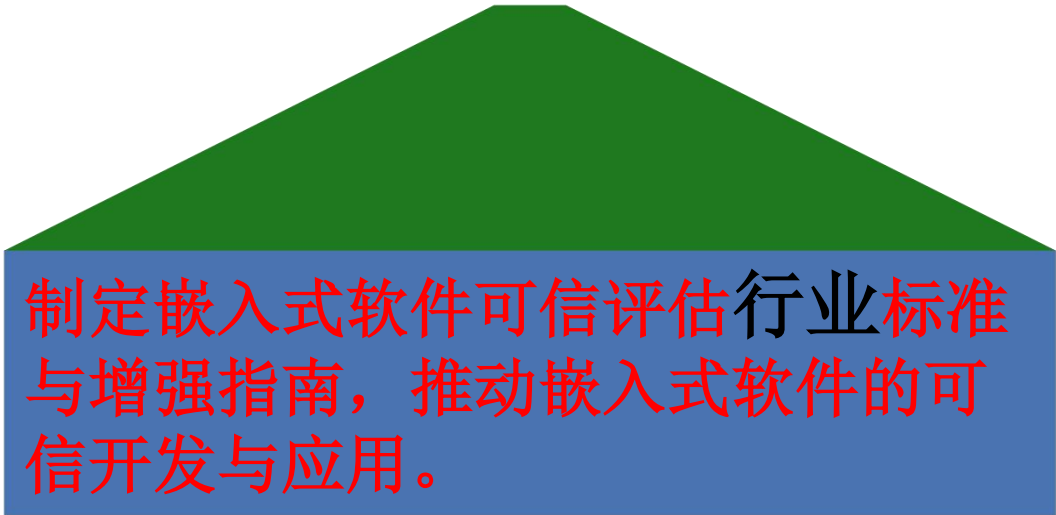
嵌入式软件在运行**10**个小时后，其可信等级为**IV**的概率是多少？

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group



制定嵌入式软件可信评估行业标准
与增强指南，推动嵌入式软件的可
信开发与应用。

问题与发展前景



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group



智测软件，品质飙升啦！



华东师范大学软件学院可信智能团队

Trustworthy Intelligence Group

谢 谢