第九章
软件优雅降级模型

陈仪香

华东师范大学软件学院可信智能团队TrIG

2024年12月1日

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

第九章 软件优雅降级模型

9.1 软件降级

9.2 优雅降级

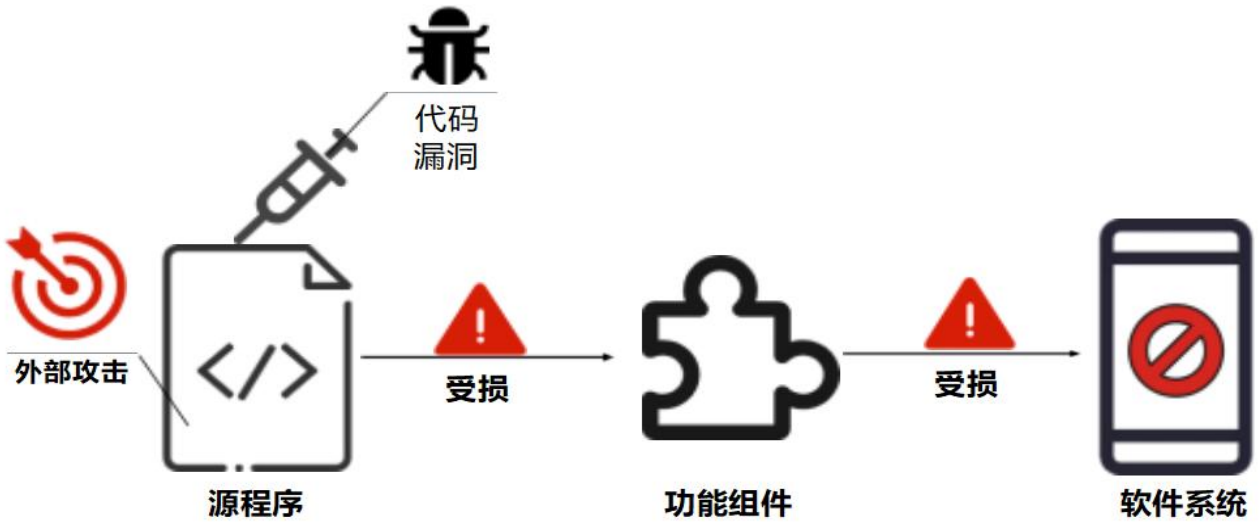
9.3 优雅降级模型

9.4 例子

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

9.1 软件降级

- 大多数软件系统 于业务需求 的增加， 需要开发，以便增加新的功能。
- 业务需求 的不断 庞大、代码变得越来越复杂，



- 如图所示，在对软件组件进行二次开发的时候， 由于开发人员的疏忽从而引入了新的程序漏洞，该程序漏洞易被外部所攻击或 由于软件长时间地运行而逐渐显露出来，从而导致该组件受损，并最终导致软件的可信性发生下降。

9.1 软件降级

为了有效地处理上述情况，在软件系统的开发设计阶段，开发和测试人员需 要考虑软件系统在实际运行中遇到的各种情况，除了添加必要的程序漏洞扫描 减少潜在的程序漏洞以外，还需提高软件系统的容错能力以便于软件在发生故 障或受到攻击后，系统仍能提供一定的服务。

- 对一个软件系统而言，系统的可靠性和稳定性是最重要的可信属性之一，为了提高系统的容错能力来保障系统的 可靠性和稳定性，一些研究人员为软件系统增加了容错机制。
- 常用的容错机制为 恢复块机制（Recovery Block mechanism），该机制通过冗余备份的方式为每个组 件增加备份组件，在组件发生故障时经由备份组件来替代并继续工作，从而帮助 系统继续提供服务。

9.1 软件降级

但该容错机制也存在以下两个缺点：

(1) 开发和维护成本高，由于为系统中的组件设立一个或多个备份组件，增 加了软件系统的开发成本，并使得软件系统的规模变得更加庞大，也增加了软件 的后期维护成本。

(2) 该机制仅针对外部攻击和内部运行故障有效，一旦故障发生原因是某个版本开发阶段遗留在组件源代码内部的程序漏洞所导致的，备份组件也势必存 在漏洞导致系统无法正常运行，此时该方法并不适用。

第九章 软件优雅降级模型

9.1 软件降级

9.2 优雅降级

9.3 优雅降级模型

9.4 例子

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

9.2 软件优雅降级

- 有些学者提出了优雅降级的概念：旨在软件系统受 到攻击或者内部发生故障后，通过牺牲部分的性能或者功能来保证软件继续运行，而非立即停止或崩溃。
- 优雅降级被视作一种特殊的容错机制，不同于传统 容错机制采用冗余备份来提升软件系统的容错能力的方式，
-

9.2 软件优雅降级

- 优雅降级机制包含故障发现机制和故障隔离机制，其中
  - 故障发现机制要求系统能够在故障出现后尽快定位故障的位置，
  - 故障隔离机制要求软件系统能够通过牺牲部分功能或者性能的方式将故障与系统正常运行部分隔离开，以保障系统其它部分正常运行。
- 考虑到容错机制会导致系统开发和维护成本上升以及适用性不广的缺点，针对软件系统采用优雅降级机制提高系统的容错能力是非常有意义的，一个能够实现优雅降级的软件系统，其可靠性和稳定性是值得肯定的。

9.2 软件优雅降级

为了解决系统中的组件在进行二次开发时因开发人员的疏忽而引入程序漏 洞从而导致系统发生故障的情况，本文结合优雅降级具备的两个机制，认为实现 优雅降级模型的系统需具备以下能力：

（1）故障定位能力：系统中的部分组件发生故障后，系统能够及时发现故障 并找出故障组件的位置；

（2）故障隔离能力：故障组件能够通过牺牲一定功能或性能来保障其他正常组件的正常运行；

通过上述两个能力，从而使得系统在发生故障后，仍能够正常提供服务，从 而保证系统的可信性。

9.2 软件优雅降级

故障定位：程序漏洞自组件二次开发时被引入以来，随着软件的长时间运行或者外部 攻击的诱发，会导致组件的可信性发生下降进而无法提供正常的功能。

为了减小 该故障对系统造成的损伤并尽可能地降低故障隔离的成本，需要在故障发生时 尽快定位故障组件的位置，以便于后续通过故障处理机制将其从系统中清除出 去。

为了及时并准确地实现故障组件的定位，结合提出的组件可信 度量方法，按照如下5个步骤实现故障发现及故障组件的定位功能 。

通过5个步骤，并结合分层可信计算方法，通过实时监控系统各层可信值变 化情况及组件可信值的变化原因，能够及时并准确地定位故障组件的位置及受 故障影响的组件情况，有助于后续故障隔离机制的实现。

9.2 软件优雅降级

故障定位五步骤

步骤 1: 参照基于组件的分层可信计算模型，实时计算软件各 层的可信值，并建立组件、功能模块、服务、子系统以及系统的可信值与时间的关系；

步骤 2: 通过系统的可信值进行实时监控，根据系统可信值是否下降判断系 统内部是否发生故障；

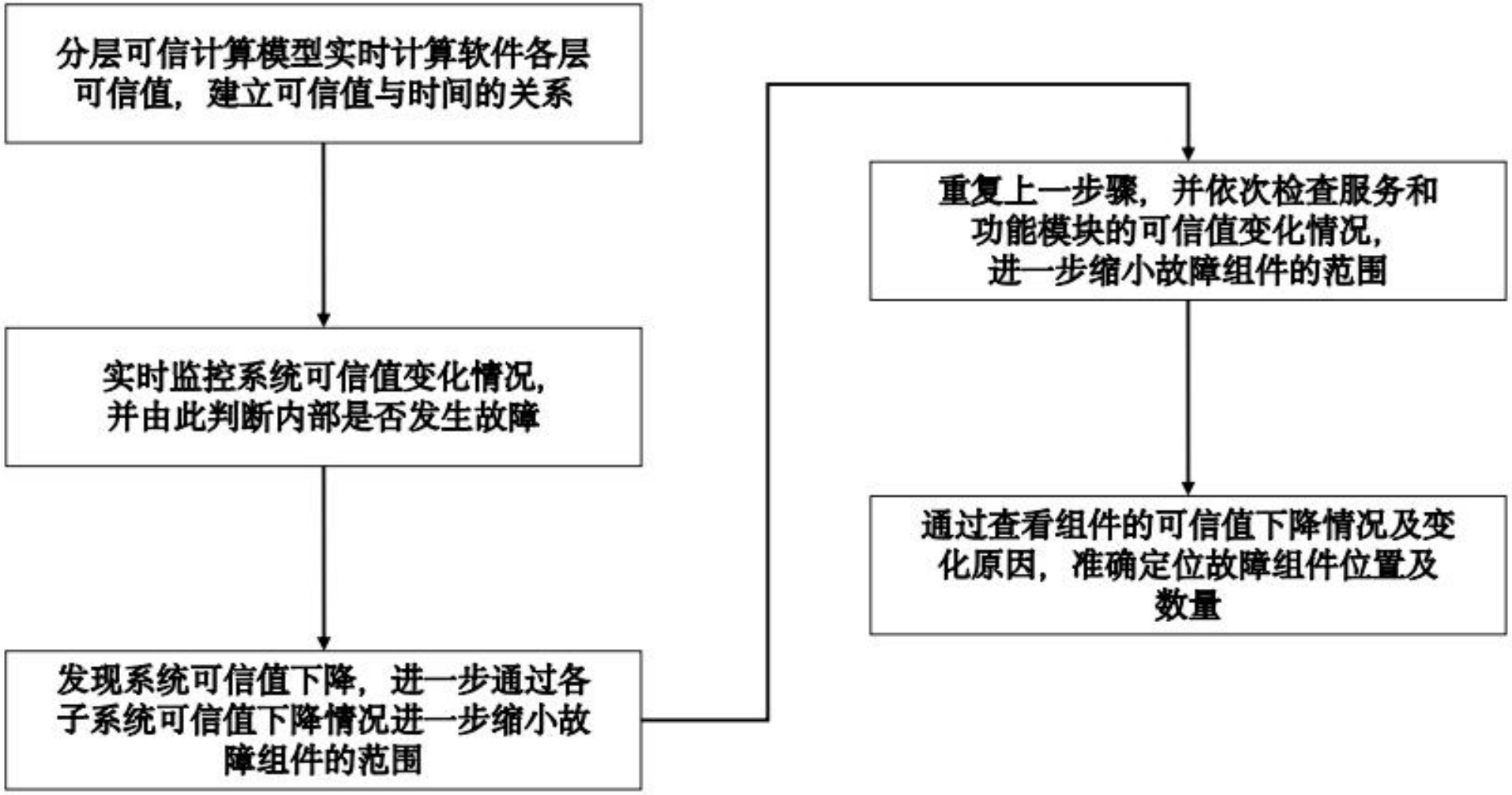
步骤 3: 一旦发现系统可信值发生下降，进一步根据各子系统的可信值变化 情况来判断哪个或者哪些子系统内部存在故障组件；

步骤 4: 重复步骤 3 的过程， 自上而下依次检查各服务和各功能模块的可信 值变化情况用于进一步细化故障的位置；

步骤 5: 最后，通过查看各组件的可信值的下降情况来判断组件是否遭受到 故障传播的影响，并结合组件可信值下降的具体原因进一步判断该组件是否为 故障源。

9.2 软件优雅降级

故障定位五步骤



9.2 软件优雅降级

故障隔离

为了保障软件系统在遭受故障组件的干扰后仍能提供服务，软件系统在自 动识别到故障发生并完成故障组件定位后，应迅速将故障进行隔离，避免该故障 对系统的可信性造成更为严重的影响。

提出了一种针对组件版本的软件降级方法用于隔离故障组件。该方法旨在通过寻找到一个未引入该程序漏洞的历史 版本的组件对故障组件进行及时替换，通过牺牲组件的部分功能和一定兼容性来完成故障的清除，从而保障系统的可信性。

9.2 软件优雅降级

故障隔离

由于不清楚程序漏洞是在开发哪个版本时引入的，如果不小心将版本回退 至过老的版本，虽然能够通过移除程序漏洞的方式将故障有效地隔离开，但是也 带来了一个新的问题，

即由于将故障回退到较低版本，该组件的功能损失较多并 且与其他组件之间的兼容性也会发生下降，从而导致系统的功能或者性能发生损失。

与之相反， 如果将版本回退至较新版本有可能会导致未能将程序漏洞完全去除，进而无法达到准确隔离故障的作用。

如何根据软件的实际情况， 以最快的 方法找到准确的版本对组件进行回退成了当下实现故障隔离功能的主要问题。

第九章 软件优雅降级模型

9.1 软件降级

9.2 优雅降级

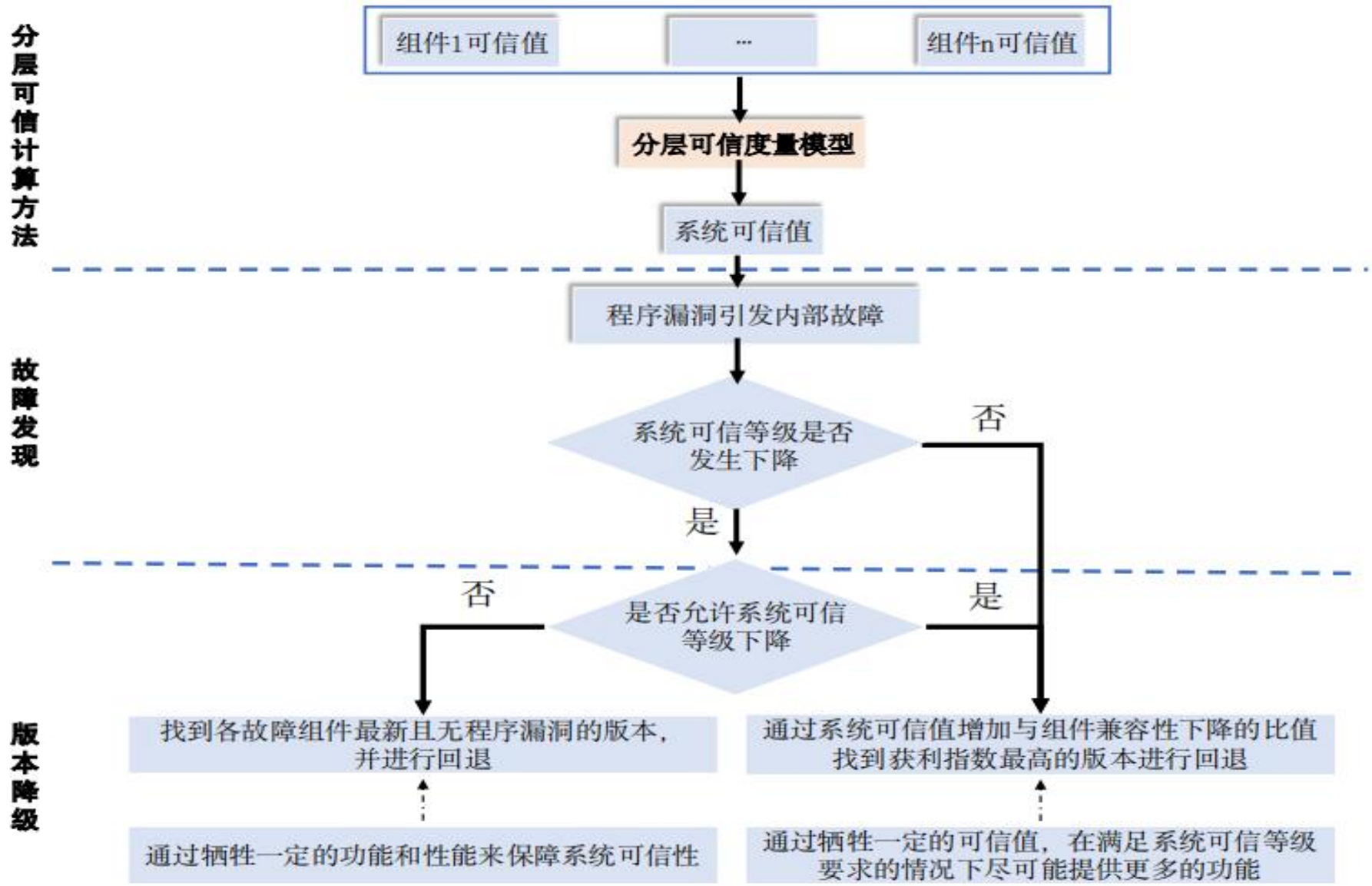
9.3 优雅降级模型

9.4 例子

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

9.3 优雅降级模型

优雅降级模型架构



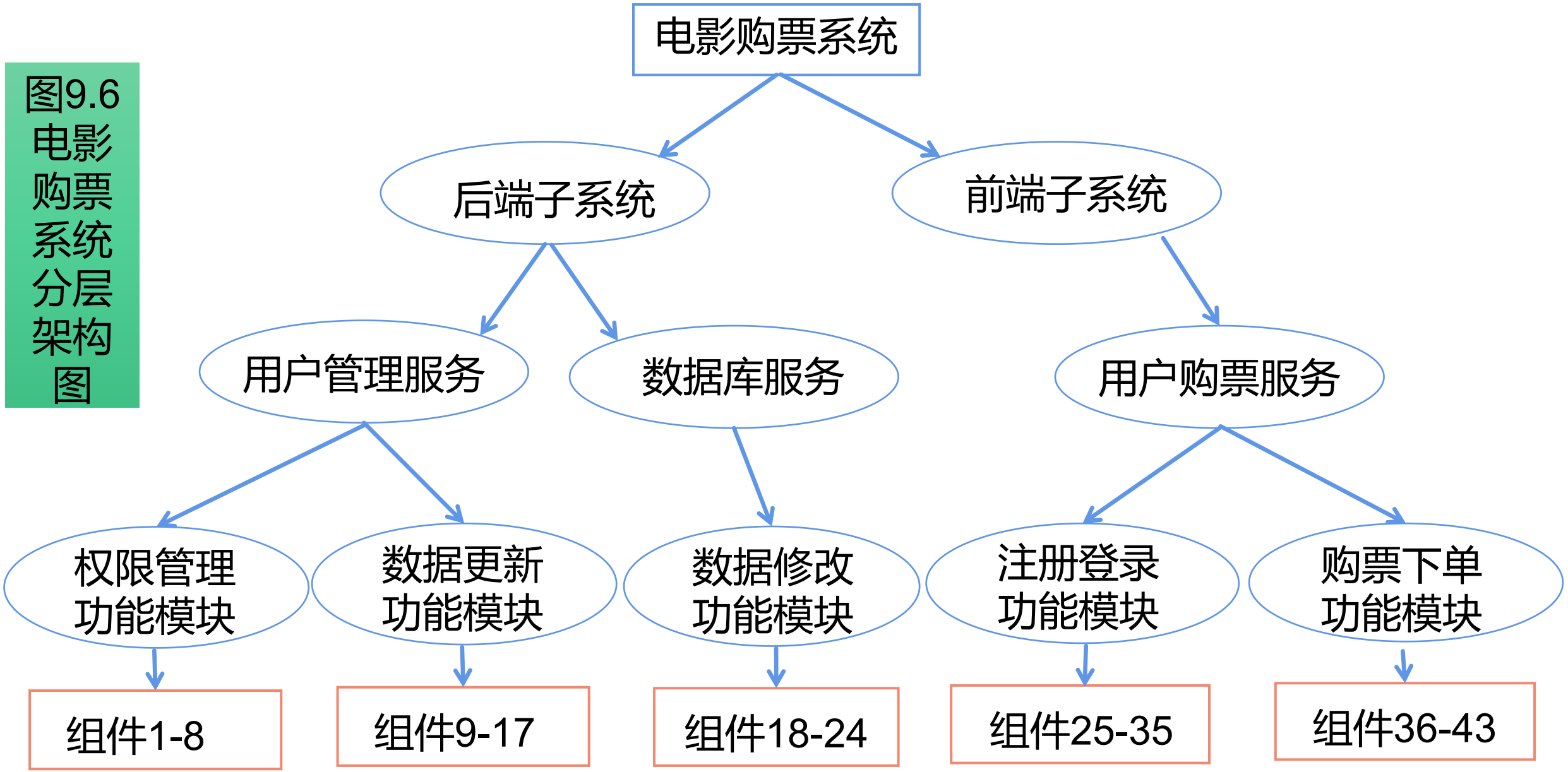
9.3 优雅降级模型

选取 Github 上一个 电影购票软件系统为例计算其可信度，该系统囊括订单确认组件、用户注册组件和订单支付组件等 43 个组件，为了方便描述，本节使用符号 $cp_1 \sim cp_{43}$ 来代表 这 43 个组件。然后根据系统的功能设计说明书和业务说明书按照业务逻辑将其进行分层拆解，得到该软件系统的分层结果如图9.6所示，

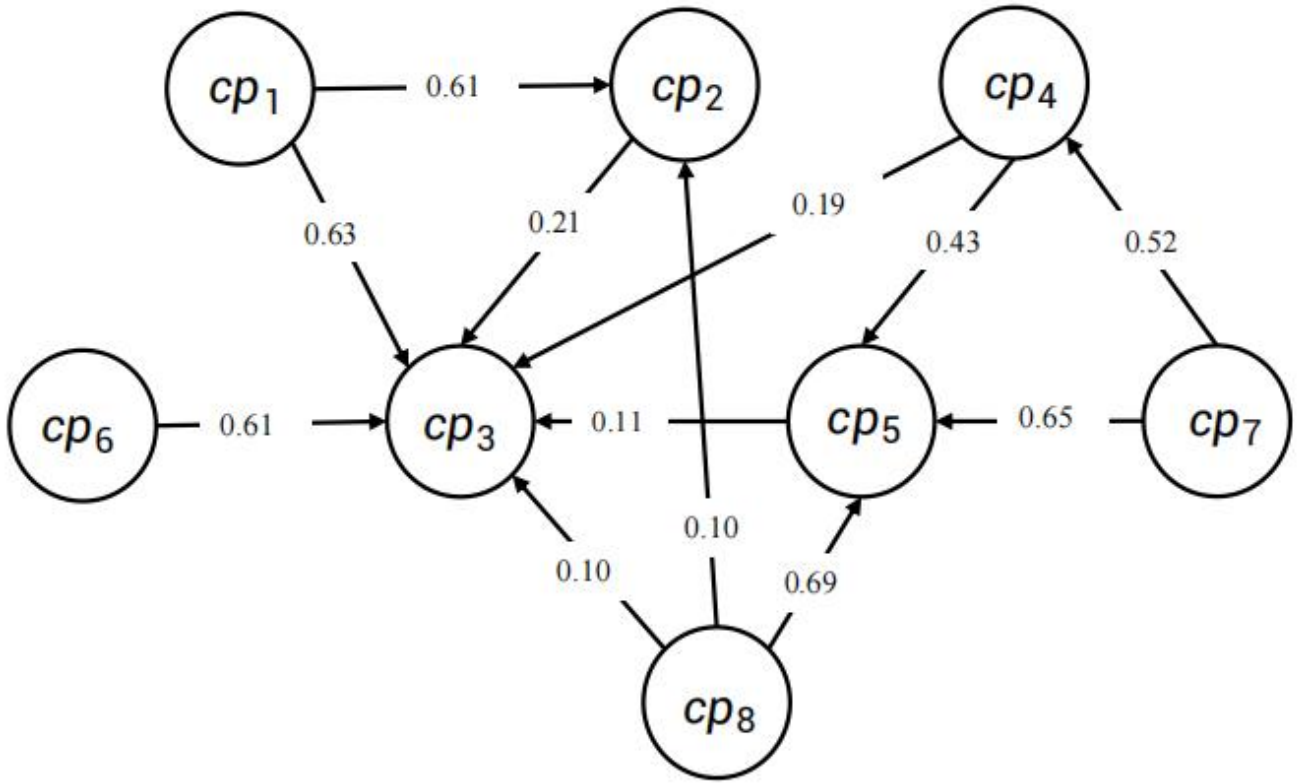
其中组件 $cp_1 \sim cp_8$ 属于 权限管理功能模块，组件 $cp_9 \sim cp_{17}$ 属于数据更新功能模块，组件 $cp_{18} \sim cp_{24}$ 属于数据修改功能模块，组件 $cp_{25} \sim cp_{35}$ 属于注册/登录功能模块，组件$cp_{36} \sim cp_{43}$ 属于购票下单功能模块。

9.3 优雅降级模型

图9.6 电影购票系统分层架构图



9.3 优雅降级模型



使用 Structure 101 for Java 静态分析工具对该系统进行静态分析得到 43 个组件自身被调用的情况以及组件之间的调用关系和调用次数。

9.3 优雅降级模型

电影购票系统中各组件依赖关系及权重

Edges	E1 ,2	E1 ,3	E2 ,3	E4 ,3	E4 ,5	E5 ,3	E6 ,3	E7 ,4	E7 ,5	E8 ,2
Weight	0.61	0.63	0.21	0.19	0.43	0.11	0.61	0.52	0.65	0.1
Edges	E8 ,3	E8 ,5	E9 , 16	E10 , 16	E12 , 17	E13 , 10	E13 , 14	E14 , 16	E15 , 10	E15 , 14
Weight	0.1	0.69	0.2	0.39	0.23	0.31	0.57	0.3	0.38	0.43
Edges	E15 , 16	E15 , 17	E17 , 10	E18 ,21	E18 ,24	E18 ,25	E18 ,26	E19 ,21	E21 ,24	E22 , 19
Weight	0.45	0.03	0.4	0.28	0.6	0.6	0.54	0.37	0.46	0.68
Edges	E22 ,24	E23 , 19	E23 ,20	E23 ,24	E25 ,35	E26 ,25	E26 ,27	E26 ,30	E27 ,29	E27 ,30
Weight	0.61	0.31	0.47	0.49	0.57	0.25	0.5	0.6	0.56	0.38
Edges	E27 ,32	E28 ,25	E31 ,29E	E33 ,27	E33 ,35	E34 ,27	E35 ,30	E37 ,36	E37 ,39	E37 ,41
Weight	0.4	0.23	0.19	0.22	0.44	0.28	0.2	0.15	0.27	0.41
Edges	E37 ,42	E38 ,37	E38 ,41	E39 ,36	E40 ,36	E40 ,39	E24 ,28	E28 ,25	E29 ,36	E27 ,24
Weight	0.33	0.27	0.51	0.49	0.23	0.24	0.54	0.56	0.04	0.27
Edges	E20 ,24	E21 ,20	E28 ,32	E29 ,25	E29 ,30	E40 ,42	E43 ,36	E43 ,38		
Weight	0.27	0.4	0.39	0.36	0.12	0.39	0.84	0.3		

9.3 优雅降级模型

电影购票系统中各组件可信度、重要度及关键组件选取结果

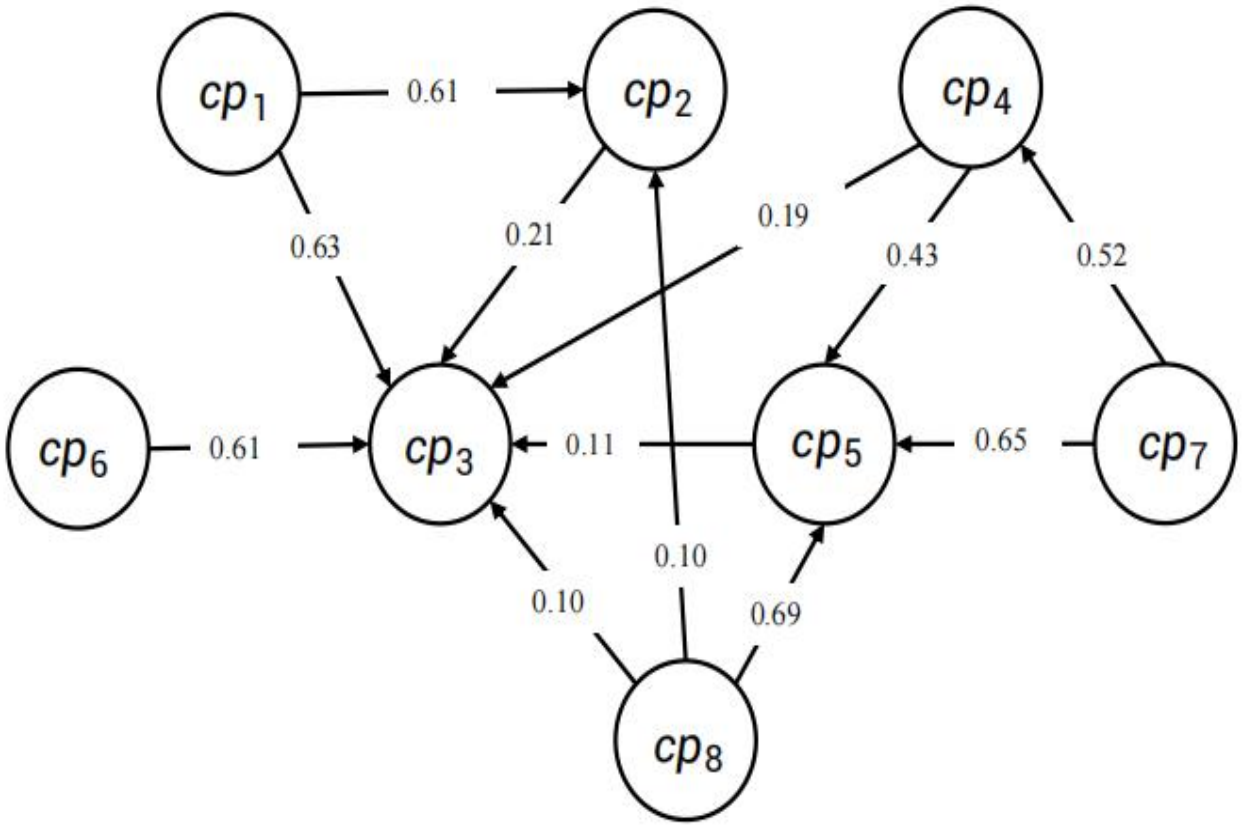
组件	cp 1	cp2	cp3	cp4	cp5	cp6	cp7	cp8	cp9	cp 10	cp 11
可信度	9.15	9.77	9.07	9.36	9.04	9.84	9.05	9.61	9.91	9.23	9.74
重要度	5.37	3.82	3.8	3.39	2.41	2.61	4.65	5.53	5.22	3.19	5.19
是否为关键组件	是	否	否	否	否	否	否	是	是	否	否
组件	cp 12	cp 13	cp 14	cp 15	cp 16	cp 17	cp 18	cp 19	cp20	cp21	cp22
可信度	9.33	9.41	9.06	9.33	9.86	9.28	9.14	9.67	9.78	9.02	9.78
重要度	3.87	5.76	5.95	5.79	4.5	3.89	9.3	6.16	4.59	4.94	6.18
是否为关键组件	否	是	是	是	否	否	是	是	否	否	是
组件	cp23	cp24	cp25	cp26	cp27	cp28	cp29	cp30	cp31	cp32	cp33
可信度	9.86	9.14	9.9	9.53	9.25	9.56	9.74	9.07	9.94	9.51	9.79
重要度	6.94	6.6	4.68	4.45	3.65	3.34	2.62	3.8	3.82	4.5	5.52
是否为关键组件	是	是	否	否	否	否	否	否	否	否	是
组件	cp34	cp35	cp36	cp37	cp38	cp39	cp40	cp41	cp42	cp43	
可信度	9.46	9.9	9.07	9.44	9.8	9.67	9.34	9.47	9.84	9.9	
重要度	3.14	4.52	3.8	3.46	4.24	3.93	6.34	5.2	3.1	4.69	
是否为关键组件	否	否	否	否	否	否	是	是	否	否	

9.3 优雅降级模型

cp_i 表示为组件，$R_{i,j}$ 表示组件 cp_i 和组件cp_j 之间存在直接依赖关系集合，
$r_{i,j}$表示cp_i 和cp_j的关系，

$w_{i,j}$ 表示组件cp_i与cp_j之间依赖边的权重。

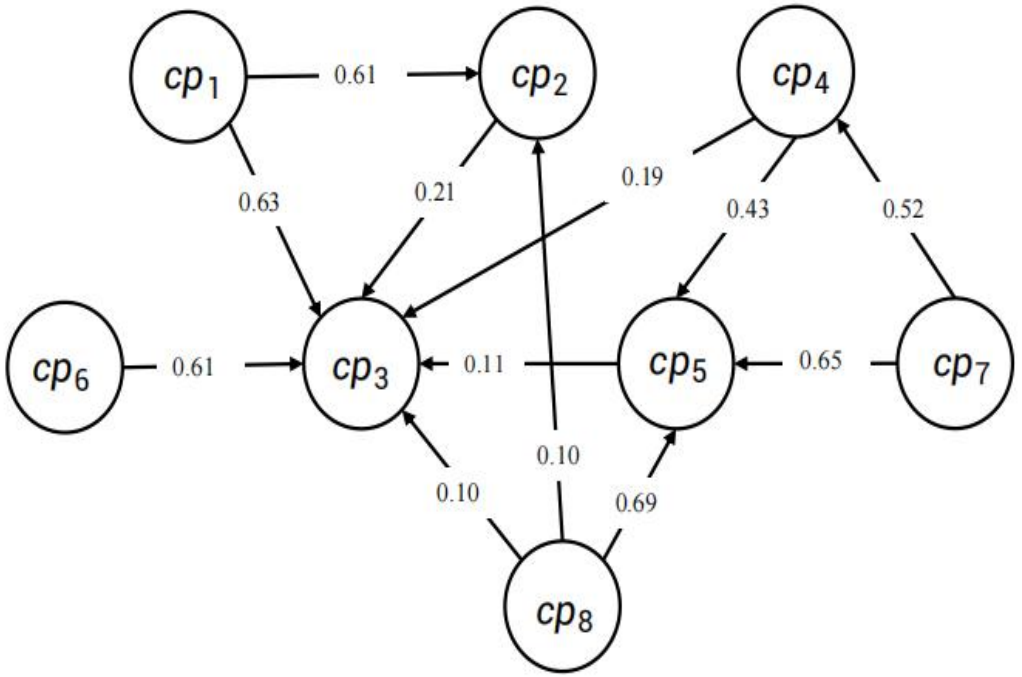
outdegree 表示组件节点cp_i的出度数，用于代表该组件节点直接影响的组件节点数目之和。



9.3 优雅降级模型

$\max_{r_{i,j} \in R_{i,j}} \{w_{i,j}\}$ 表示从组件cp_i出发的最大边的权重，用于代表该组件节点的最大直接传播影响；

$\sum_{r_{i,j} \in R_{i,j}} \{w_{i,j}\}$ 表示从组件cp_i出发的所有边的权重之和，用于代表该组件节点的传播影响之和；

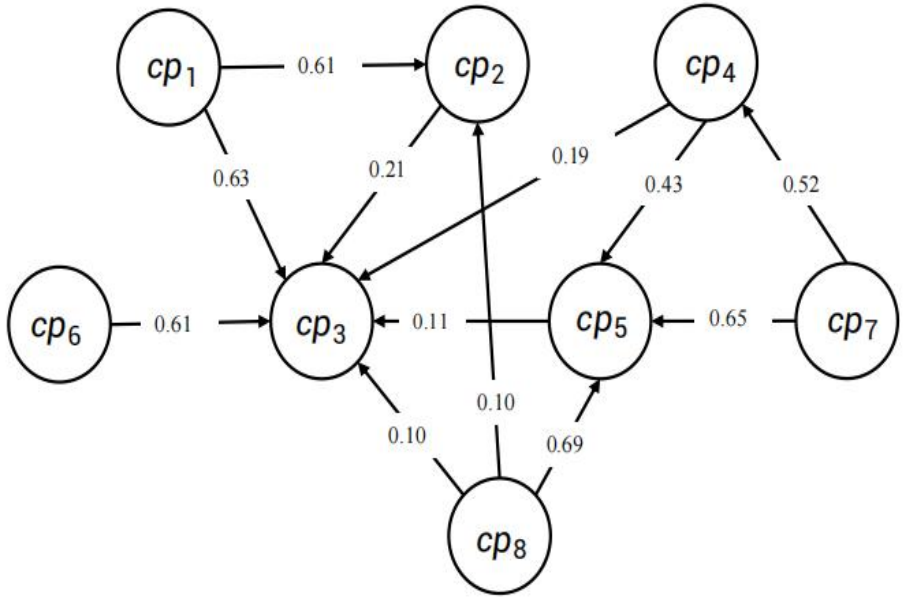


$$\mathrm{IMout}(cp_i) = \max_{r_{i,j} \in R_{i,j}} \{w_{i,j}\} \times \left(\sum_{r_{i,j} \in R_{i,j}} w_{i,j} \right) \times \mathrm{outdegree}$$

9.3 优雅降级模型

其中，$S(cp_i)$ 表示组件节点cp_i 的自身重要度，
$frq(cp_i)$ 表示组件节点cp_i 被系统调用的频数，
n 表示节点的数目。

$$S(cp_i) = \frac{frg_i}{\sum_{j=1}^{n} frg_j} \times 10$$



9.3 优雅降级模型

$$\mathrm{IM}(cp_i) = p \times \mathrm{IMout}(cp_i) + (1-p) \times S(cp_i)$$

$IM(cp_i)$表示组件cpi的重要程度，由该组件的出度重要度$IMout(cp_i)$，组件自身重要度$S(cp_i)$组成。

9.3 优雅降级模型

基于组件版本回退的软件降级模型

为了有效找到合适的组件版本用于组件回退，本文考虑到在进行组件版本 回退时所造成的版本兼容性下降问题，并将此视作版本回退的成本，并建立了 基于组件版本号的成本函数。而关于组件的版本号，开发人员通常遵循由 Tom Preston-Werner 建立的语义化版本控制规范，采用 Major.Minor.Patch 格式对组件 的版本号进行命名，如公式9.1所示。

$$V = X.Y.Z \tag{9.1}$$

其中 V 表示组件的版本，X 表示组件的主版本号，Y 表示组件的次版本号，Z 表示组件的修订号。

9.3 优雅降级模型

参照语义化版本控制规范的定义，

- 主版本号通常表示为不兼容性的 API 修 改后版本，当大版本号发生改变时，兼容性发生大幅度下降，甚至导致组件之间 不兼容。

- 次版本号表示为组件功能数的改变后的版本，当组件增加了新的功能 时，次版本号应随之递增，当次版本号发生改变时，不同组件之间的兼容性会发 生一定幅度的下降。

- 修订号表示为组件对内部 BUG 修复版本，当组件修复了自 身内部的 BUG 时，修订号会随之递增，修订号的变更不会导致组件之间的兼容 性发生变化。

9.3 优雅降级模型

- 虑到组件的大版本号和次版本号发生改变会导致组件之间的兼容 性下降，而修订号的改变不会导致组件之间的兼容性发生下降，一旦将故障组件 进行版本回退，势必会导致该组件与直接依赖于它的其他组件间的兼容性发生 下降。
- 为了方便描述， 使用兼容度来指代组件之间的兼容性，并设置兼容度 的取值范围为 [1,10] 代表不同程度的兼容性，其中兼容度为 10 表示组件之间的的兼容性较高，兼容度为 1 表示组件之间的兼容性较低。
- 为了计算组件版本回退 导致的兼容度下降程度，我们假设在软件的设计与开发阶段，在理想状态下各组 件之间的初始兼容度应当为 10 ，并将组件间的兼容度的下降之和视作单个故障 组件的回退成本，并建立各组件回退成本与组件版本之间的关系如下式所示

9.3 优雅降级模型

$$Cost(i)=\alpha_i\times\sum_{r_{i.j}\in R_{i,j}}\left(\alpha_j\times\frac{\Delta X_i}{X_i}(10-|X_i-X_j|)+\alpha_j\times\frac{\Delta Y_i}{Y_i}(10-\beta|Y_i-Y_j|)\right)$$

$$\Delta X_i=|X_i^*-X_i|,\quad \Delta Y_i=|Y_i^*-Y_i|$$

- Cost(i) 代表故障组件 cp_i 进行版本回退时所带来的回退成本，
- $R_{i,j}$ 表示组件 cp_j 与组件 cp_i 存在直接依赖关系集合，$r_{i,j}$ 表示组件 cp_j 与组件 cp_i 之间的关系，用$r_{i,j}\in R_{i,j}$ 表示所有直接依赖于组件 cp_i 的其他组件的集合；
- ΔXi 和 ΔY_i 表示组件 cp_i 进行版本回退后的主版本号及次版本号与未进行版本回退的初始版本号之间的差值，
- X_i *和 Y_i* 则分别表示组件 cp_i 版本回退前的主版本号和次版本号；
- β 表示次版本号对兼容度的影响程度，取值范围为 (0,1)；
- α_i 和 α_j 分别表示组件 cp_i 和 cp_j 所对应的重要性权重，可由公式9.3得到。

9.3 优雅降级模型

$$\alpha_i = \frac{IM(cp_i)}{\sum_{k=1}^{n} IM(cp_k)} \quad (9.3)$$

组件的权重α_i 由组件的重要度$IM(cp_i)$计算

$$\mathrm{IM}(cp_i) = p \times \mathrm{IMout}(cp_i) + (1-p) \times S(cp_i)$$

$IM(cp_i)$表示组件cpi的重要程度，由该组件的出度重要度$IMout(cp_i)$，组件自身重要度$S(cp_i)$组成。

第九章 软件优雅降级模型

9.1 软件降级

9.2 优雅降级

9.3 优雅降级模型

9.4 例子

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

9.4 例子

为了验证优雅降级模型的可行性，同样选取电影购票软件作为实例。为了模拟优雅降级的实验环境，从该电影购票软件系统的 43 个组件中随机选择其中 10 个组件作为故障组件，并随机地对这些组件的某个版本进行人为故障注入，通过上述方式模拟该软件系统中部分组件由于二次开发引入了程序 漏洞的情况。

9.4 例子

假设各故障组件中的程序漏洞均已导致组件的可信值发生下降，此时可通过基于分层可信度量模型所开发的软件系统分层可信度量工具对各组件可信度的实时监控，可根据组件可信度的下降情况及下降原因完成故障组件的定位。

如下图所示，监控到用户验证组件内部出现故障导致可信证据“接口 被正确调用的比例”对应的度量元由 A 下降为 C，从而导致组件的可信度从 9.35 下降至 7.165。

查看各组件状态

名字	变化发生的日期	变化发生的原因	可信值
用户验证组件		用户验证组件由于自身内部发生故障导致可信证据"接口被正确调用的比例"对应的度量元由'A'下降为'C'，组件的可信度由9.3500下降至7.1650	7.165

9.4 例子

通过对所有组件可信度变化的实时监控，可定位得到 10 个故障组件的结果 如表9.1所示。

9.4 例子

表 9.1 故障组件定位结果

序号	1	2	3	4	5
组件名	用户验证组件	用户管理组件	异常响应组件	隐私设置组件	数据验证组件
当前版本	V1.6.2	V1.4.2	V1.8.1	V1.10.0	V1.5.2
序号	6	7	8	9	10
组件名	日志记录组件	用户注册组件	密码重置组件	订单确认组件	电影搜索组件
当前版本	V1.3.0	V1.5.0	V1.1.2	V1.8.2	V1.10.0

9.4 例子

待定位到所有故障组件后，为了保证能在较短时间内找到故障组件的最优回退版本，这里采用基于二分搜索的组件回退版本搜索算法所示的过程， 通过二分搜索算法的思想，通过不断缩小搜索区间直到找寻到各故障组件的最优回退版本。

同样以用户验证组件为例，使用二分搜索算法求解该组件的最优回退版本，其中用户验证组件存在 8 个版本，经过 3次区间缩小及重新度量对应版本的组件可信度，最终确定用户验证组件的最佳回退版本为 V1.3.0。

9.4 例子

表 9.2 采用组件回退版本搜索算法所得四元组信息

组件名	用户验证组件	用户管理组件	异常响应组件	隐私设置组件	数据验证组件
最佳回退版本	V1.3.0	V1.2.2	V1.6.0	V1.8.0	V1.5.0
回退成本 ($\times 10^{-3}$)	5.3	4.3	4.6	3.4	9.4
系统可信度上升情况	0.133	0.034	0.044	0.024	0.058
组件名	日志记录组件	用户注册组件	密码重置组件	订单确认组件	电影搜索组件
最佳回退版本	V1.2.1	V1.4.0	V1.1.0	V1.7.1	V1.6.0
回退成本 (×10−3)	4.4	6.1	1.5	4.7	2.5
系统可信度上升情况	0.059	0.028	0.014	0.164	0.046

9.4 例子

在得到如表9.2所示各故障组件的四元组信息后，结合该电影购票软件系统 的可信要求选择对应的故障组件选择算法，通过选中其中部分故障组件进行版本回退以满足系统的可信要求。

通过比对系统的可信等级要求与系统实际可信度，发现系统的可信等级较未发生故障前发生下降。假设软件系统不允许系统可信等级发生下降，此时需通过算法9.2基于分支定界法的故障组件选取算法来搜寻最优的故障组件选择方案。

9.4 例子

在得到如表9.2所示各故障组件的四元组信息后，结合该电影购票软件系统 的可信要求选择对应的故障组件选择算法，通过选中其中部分故障组件进行版本回退以满足系统的可信要求。

通过比对系统的可信等级要求与系统实际可信度，发现系统的可信等级较未发生故障前发生下降。假设软件系统不允许系统可信等级发生下降，此时需通过基于分支定界法的故障组件选取算法来搜寻最优的故障组件选择方案。

接入分层可信度量工具中以便于能实时计算系统和子系 统的可信值用于判断组件选取方案是否满足系统可信要求，最后得到如图9.5所示结果。

9.4 例子

图 9.5 基于分支定界法的故障组件选取算法结果展示

优雅降级方案

系统名字	可信等级变化情况
电影购票系统	该系统可信等级最高为: 5,当前可信等级为: 4 , 可信等级发生下降.

是否允许系统的可信等级发生下降

否

确定提交

系统名字	故障组件选择方案
电影购票系统	请选择以下组件进行版本回退: 用户验证组件: 将其回退至V1.3.0, 所需成本5.3*10^(-3); 隐私设置组件: 将其回退至V1.8.0, 所需成本3.4*10^(-3); 订单确认组件: 将其回退至V1.7.1, 所需成本4.7*10^(-3); 电影搜索组件: 将其回退至V1.6.0, 所需成本2.5*10^(-3), 此时系统可信等级将满足5级可信, 花费总成本最少为1.59*10^(-2);

9.4 例子

待发现系统的可信等级自 5 级可信下降至 4 级可信，且系统不允许可信等级发生下降，此时点击提交按钮，系统开始搜寻满足要求的故障 组件选取方案，

当程序计算完毕后，选取四个组件进行版本回退，分别为用户验证组件、隐私设置组件、订单确认组件以及电影搜索组件。

根据四个组件所对 应的四元组中的最佳回退版本及回退成本信息可知，此时需最少花费回退成本 1.59×10^{-2} 方能保证系统满足 5 级可信。

作业

1. 软件优雅降级的目的是什么？
2. 优雅降级模型架构是什么？
3. 当软件系统发生故障后，可信等级已经发生了下降，但不允许可信等级发生下降的情况下如何选择组件的版本？