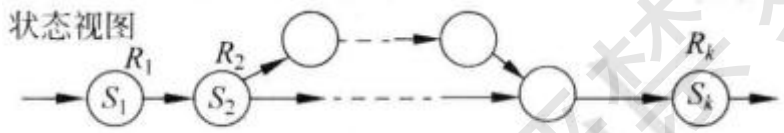
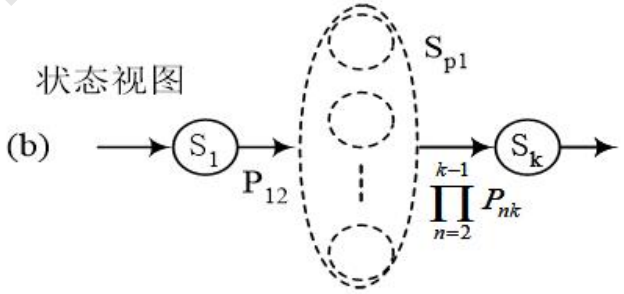
软件体系结构原理、方法与实践

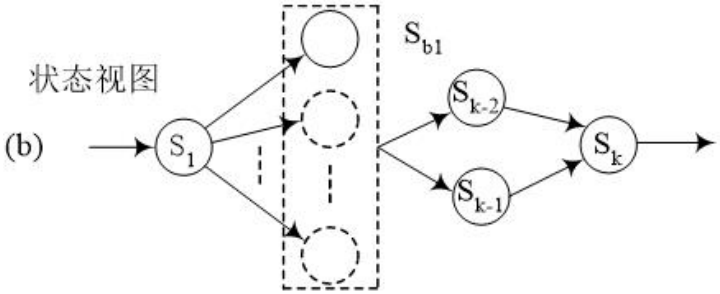
新的开始，新的起点，让我们一起为梦想而努力。

概述

- 在基于构件的可靠性模型中，通过状态图来描述系统的行为。软件系统的可靠性依赖于状态的**执行顺序**和**每一个状态的可靠性**。



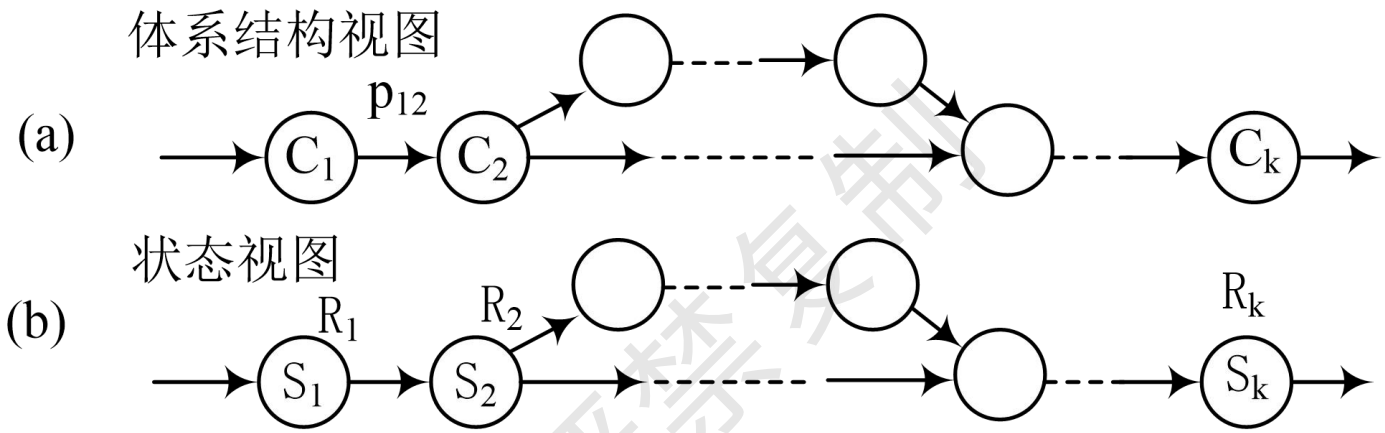




概述

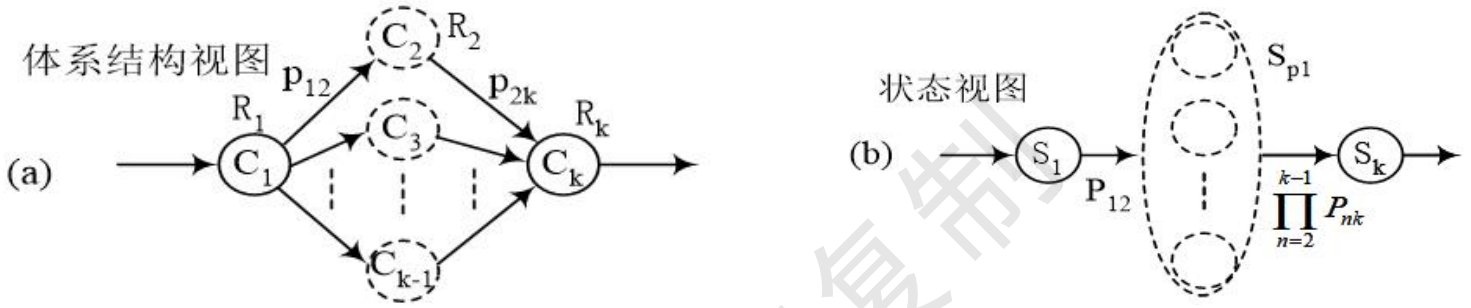
$$
M=\begin{array}{c}
\\
S_1\\
S_2\\
\vdots\\
S_i\\
\vdots\\
S_{n-1}\\
S_n
\end{array}
\begin{array}{c}
\begin{array}{ccccccc} S_1 & S_2 & \cdots & S_i & \cdots & S_{n-1} & S_n \end{array}\\
\begin{bmatrix}
0 & R_1P_{12} & \cdots & R_1P_{1i} & \cdots & R_1P_{1n-1} & R_1P_{1n}\\
R_2P_{21} & 0 & \cdots & R_2P_{2i} & \cdots & R_2P_{2n-1} & R_2P_{2n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
R_iP_{i1} & R_iP_{i2} & \cdots & 0 & \cdots & R_iP_{in-1} & R_iP_{in}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
R_iP_{(n-1)1} & R_iP_{(n-1)2} & \cdots & R_iP_{(n-1)i} & \cdots & 0 & R_iP_{(n-1)n}\\
R_nP_{n1} & R_nP_{n2} & \cdots & R_nP_{ni} & \cdots & R_nP_{n(n-1)} & 0
\end{bmatrix}
\end{array}
$$

体系结构的可靠性建模 – 顺序风格模型



$$\begin{cases} M(i,j)=0, S_i\text{不能够到达}S_j \\ M(i,j)=r_i p_{ij}, S_i\text{能够到达}S_j \end{cases}, \ for\ 1\le i,j\le k$$

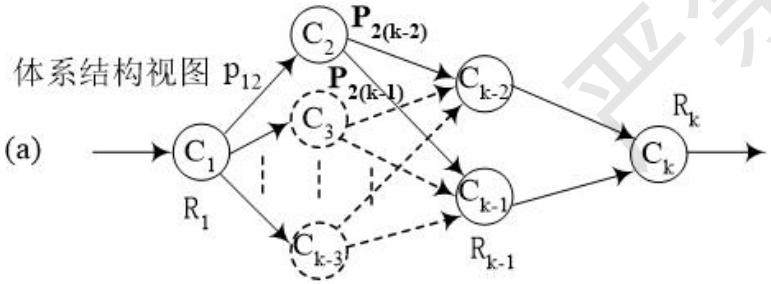
体系结构的可靠性建模 – 并行/管道-过滤器结构风格

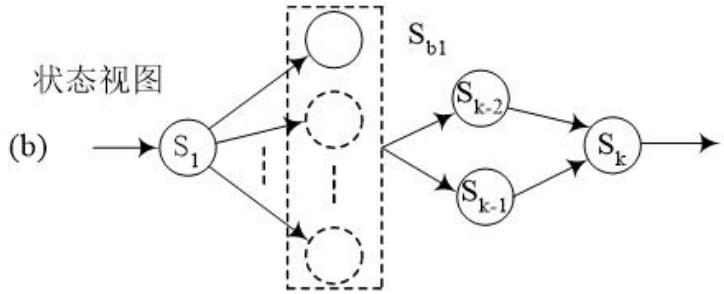


$$\begin{cases} M(i,j) = R_i P_{ij}, S_i \notin S_p \\ M(i,j) = \prod_{Cn\ in\ Si} R_n P_{nj}, S_i \in S_p, \quad for\ \ 1 \le i,j \le |S| \quad and\ \ 1 \le n \le k \\ M(i,j) = 0, S_i \text{不能到达} S_j \end{cases}$$

体系结构的可靠性建模 – 容错结构风格

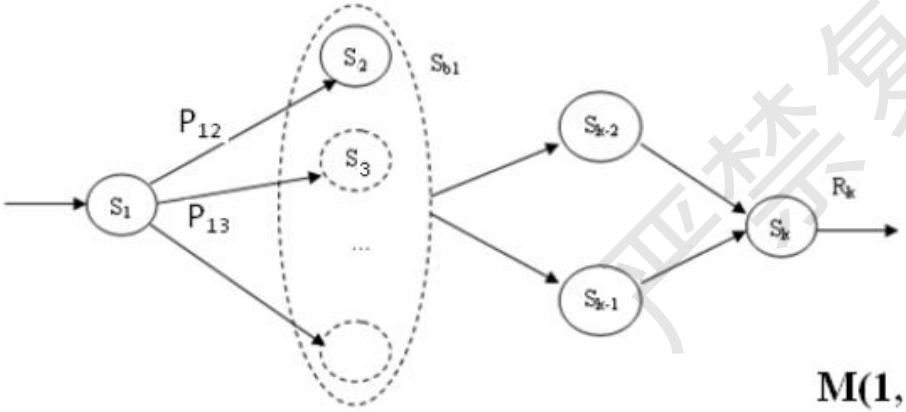
- 容错风格有一个原始构件和一系列备份构件组成，原始构件和备份构件都放在一个并行结构下，当一个构件出错，其它构件可以提供服务，即容错构件聚集是状态S_{b1}内





体系结构的可靠性建模 – 容错结构风格

- 当k=6时，$M(1,\{S_{b1}\})=R_1P_{12}+(1-R_2)R_1P_{13}$
- 当k=7时，$M(1,\{S_{b1}\})=R_1P_{12}+(1-R_2)R_1P_{13}+(1-R_2)(1-R_3)R_1P_{14}$



$$M(1,\{S_{b1}\})=R_1P_{12}\left(1+\sum_{n=3}^{k-3}\left(\prod_{m=2}^{n-1}(1-R_m)\right)\right)$$

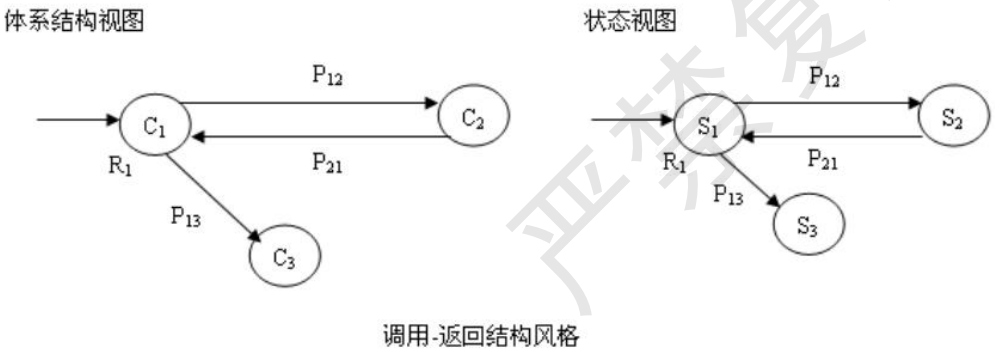
$$M(\{S_{b1}\},\{S_{k-1}\})=R_{b1}P_{2(k-1)}$$

$$M(\{S_{b1}\},\{S_{k-2}\})=R_{b1}P_{2(k-2)}$$

体系结构的可靠性建模 – 调用-返回结构风格

$M(1,3)=R_1P_{13}$ ； $M(2,1)=R_2P_{21}$

由于S_1可以多次迁移到S_2，计算$M(1,2)$时只考虑从S_1到S_2迁移的概率，不考虑S_1的可靠性。



调用-返回结构风格

$$\begin{cases} M(i,j)=R_iP_{ij}, S_i可以到达S_j \\ M(i,j)=Pij, S_i可以到达S_j, S_j是被调用构件, 1\le i,j\le k \\ M(i,j)=0, S_i不能到达S_j \end{cases}$$

体系结构的可靠性建模 – 系统的可靠性模型化步骤

- 通过系统的详细说明书，确定系统所采用的体系结构风格。
- 把每一种体系结构风格转换成状态视图，并计算状态视图中每一个状态的可靠性及其相应的迁移概率。
- 通过整个系统的体系结构视图，把所有的状态视图集成为一个整体状态视图。
- 通过整体状态视图构造系统的迁移矩阵，并计算系统的可靠性。

多选题 1分

设置

下列说法正确的是：

A 软件系统的可靠性依赖于状态的执行顺序和每一个状态的可靠性

B 状态图是一个无向图，一个状态表示一个构件的执行

C 系统的可靠性通过迁移矩阵来计算，M(i, j)表示能够从状态S_i到状态S_j成功迁移的概率

D 在顺序结构风格中，如果状态S_i到状态S_j是可达的，则从状态S_i迁移到S_j成功的概率$M(i\ j)=R_i*P_{i\ j}$

E 在调用返回风格中由于S_1可以多次迁移到S_2，计算M(1, 2)时只考虑从S_1到S_2迁移的概率，不考虑S_1的可靠性

提交

单选题 1分

设置

对于系统的可靠性模型化步骤排序正确的是：
1.把每一种体系结构风格转换成状态视图，并计算状态视图中每一个状态的可靠性及其相应的迁移概率。
2.通过整体状态视图构造系统的迁移矩阵，并计算系统的可靠性。
3.通过系统的详细说明书，确定系统所采用的体系结构风格。
4.通过整个系统的体系结构视图，把所有的状态视图集成为一个整体状态视图。

A 1,2,3,4

B 1,3,4,2

C 3,1,4,2

D 3,2,4,1

提交

软件体系结构的风险分析 – 动态方法

- **动态方法：**用来评估执行中的软件体系结构的动态耦合度和动态复杂度。
- **动态耦合度：**用来度量在特定的执行场景中，两个相互连接的构件或连接件之间的活跃程度。
- **动态复杂度：**用来测量特定构件在给定场景下的动态行为。

软件体系结构的风险分析 – 构件依赖图方法

- **构件依赖图（CDG）**：用于在体系结构级进行可靠性分析的概率模型。它是控制流图的一个扩展。
- CDG图是一个有向图，它描述了构件、构件的可靠性，连接件、连接件的可靠性和迁移概率。
- CDG是从场景中开发出来的，通常用UML序列图对场景进行建模。

软件体系结构的风险分析 – 构件依赖图方法

- CDG=< N，E，s, t >,N 是节点集，E是边集，s和t分别是初始点和终点。
- 例如：N={n},E={e}, n=<C_i,RC_i,EC_i>其中C_i表示第i个构件，RC_i表示C_i的可靠性，EC_i表示C_i的平均执行时间。

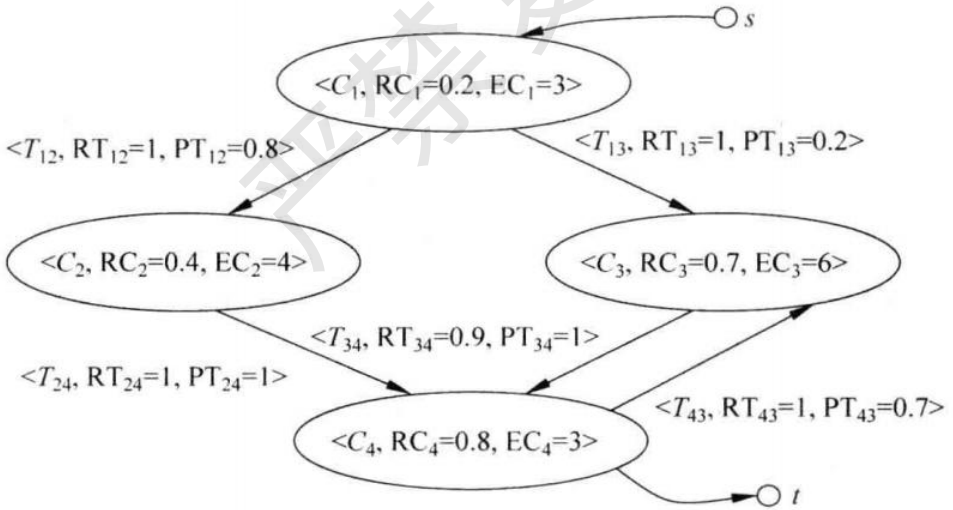


图 10-5 一个简单的 CDG

软件体系结构的风险分析 – 构件依赖图方法

- 其中PS_k是场景S_k的执行概率，|S|是场景的总数，$Time(C_i)$是构件C_i的执行时间，$Time(C_i)$可通过序列图中构件C_i在其生命线上的活跃时间总数进行估计，C_i是场景S_k中的构件;第i个构件的平均执行时间EC_i为场景执行概率乘以构件活跃时间之和，公式如下：

$$EC_i = \sum_{k=1}^{|S|} PS_k * Time(C_i)_{C_i \text{ in } Sk}$$

软件体系结构的风险分析 – 构件依赖图方法

- e=<T_{ij},RT_{ij},PT_{ij}>,其中Tij是从节点ni到nj的迁移，RT_{ij}是该迁移的可靠性，PT_{ij}是该迁移的概率，PT_{ij}可通过下式获得：

$$PT_{ij} = \sum_{k=1}^{|S|} PS_k \times \left(\frac{|\text{Interact}(C_i, C_j)|}{|\text{Interact}(C_i, C_l)|_{l=1,\cdots,n}} \right)$$

$C_i, C_j, C_l, (j, l \neq i)$都是场景 S_k 中的构件，同样，PS_k 是场景 S_k 的执行概率，$|S|$是场景的总数，N 是应用系统中的构件数量，$|\text{Interact}(C_i, C_j)|$是构件 C_i 和构件 C_j 在场景 S_k 中的交互次数。

软件体系结构的风险分析 – 步骤

(1) **体系结构建模**：用ADL对体系结构进行建模。

(2) **复杂性分析**：通过模拟方法对构件和连接件进行复杂性分析。

(3) **严重性分析**：通过 FMEA (failure mode and effectanalysis)和模拟运行对构件和连接件失效执行严重性分析。

(4) **开发启发式风险因子**：通过复杂性和严重性因素，为体系结构中的每一个构件和连接件计算其启发式风险因子。

(5) **构件依赖图**:建立用于风险评估的CDG。

(6) **评估和分析**：通过图论中的算法执行风险评估和分析。

软件体系结构的风险分析 – 步骤

1. 采用体系结构描述语言ADL对体系结构进行建模

- 软件的风险可以和系统运行时的失效性相关联，因此 软件系统的风险可以通过对单个构件的行为和系统的动态行为进行分析和评估。
- 选择的ADL必须对构件之间的交互和单个构件的行为提供支持。
- 可以使用UML的状态图描述单个构件的行为，使用UML的序列图来描述构件之间的交互。

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(1) 构件的复杂性--程序流图的环路复杂性度量

- McCabe的基于程序流图的环路复杂性度量：VG 二 e – n + 2
其中，e是图的边数，n是图的结点数。

- 可以基于此方法为每个构件开发复杂性因子，对每个场景统计Sk中的每个构件C_i的执行路径的环路复杂性，这类度量称为互操作复杂性，用$cpx_k(C_i)$表示。

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(1) 构件的复杂性--复合状态的环路复杂性

- 复合状态的环路复杂性：可以通过统计所有迁移片段的环 路复杂性计算，包括起始点出口代码片断和终结点的入口 代码片断。

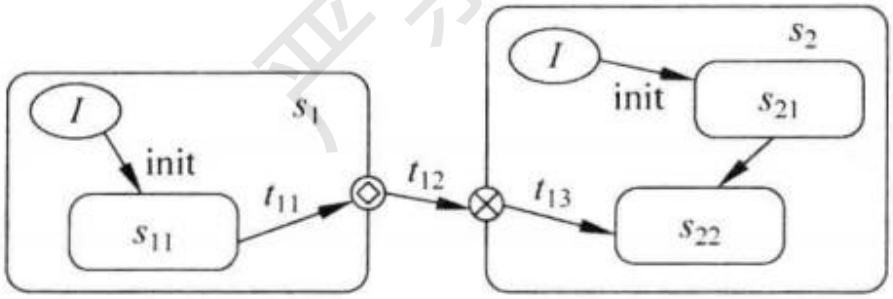


图 10-6 两个复合状态之间的迁移图

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(1) 构件的复杂性--复合状态的环路复杂性

- 从状态S_{11}到状态S_{22}的一个迁移的VG如下，其中，VG_x是出 口编码片断的复杂性，VG_a是活动代码片断的复杂性，VG_e 是入口代码片断的复杂性。

$$VG_x(s_{11}) + VG_a(t_{11}) + VG_a(t_{12}) + VG_e(s_1) + VG_a(t_{13}) + VG_e(s_{22})$$

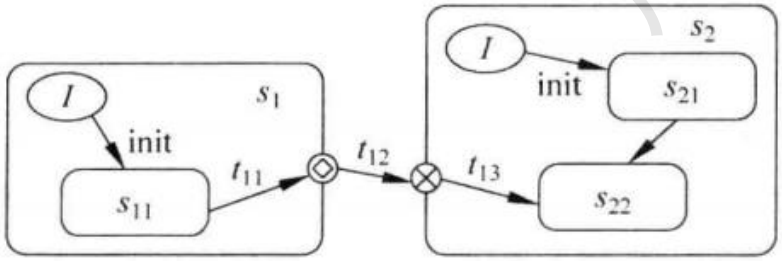


图 10-6 两个复合状态之间的迁移图

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(1) 构件的复杂性--构件的操作复杂性

- 使用场景概率PS_k能够通过下式统计每个构件的操作复杂性 度量，其中|S|表示场景数。

$$\mathrm{cpx}(C_i) = \sum_{k=1}^{|S|} \mathrm{PS}_k \times \mathrm{cpx}_k(C_i)$$

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(2) 连接件的复杂性

- 可以使用动态耦合法进行分析：动态耦合中包括输出动态耦合和输入动态耦合。假设$EC_k(C_i, C_j)$是与构件C_j相关的构件C_i的输出耦合，它等于从构件C_i到构件C_j所发的消息数在场景S_k的执行期内所交换的消息总数的百分比，即：

$$EC_k(C_i, C_j) = \frac{|\{M_k(C_i, C_j) \mid C_i, C_j \in A \wedge C_i \neq C_j\}|}{MT_k} \times 100$$

$M(C_i, C_j)$是在场景Sk执行期间，从构件C_i到构件C_j所发送的消息集；MT_k 是在场景S_k执行期间，所有构件进行的消息交换总数。

软件体系结构的风险分析 – 步骤

2. 通过模拟方法执行复杂性分析

(2) 连接件的复杂性

- EC(C_j, C_j)度量公式：
  其中：|S|是场景总数，PS_k是场景S_k的执行概率。

$$EC(C_i, C_j) = \sum_{k=1}^{|S|} PS_k \times EC_k(C_i, C_j)$$

软件体系结构的风险分析 – 步骤

3. 通过FMEA和模拟运行执行严重性分析

- 构件的复杂性并不是估计失效风险的完整方法，有些构件的复杂度很低，但它们却充当一个主要的安全角色，如果失效可能导致一个灾难性的后果。
- 通过每一个潜在的失效方式的级别和失效方式的后果进行严重性分析。
- FMEA技术是一个用于描述系统可能的失效方式和识别失效后果的系统方法。

软件体系结构的风险分析 – 步骤

3. 通过FMEA和模拟运行执行严重性分析

(1) 失效方式的识别：为简单起见，只考虑下面一些失效分析技术：

- **单个构件的失效方式**:在体系结构描述模型中，每一个 构件用一个状态图来描述，它是一个基于状态的构件行 为描述。在识别单个构件的失效方式中，仅考虑功能性 故障分析和基于状态的故障分析。
- **单个连接件的失效方式**:在体系结构模拟模型中，为了 识别单个连接件的失效方式，仅考虑在消息参数与消息参数之间的接口错误失效误差。

软件体系结构的风险分析 – 步骤

3. 通过FMEA和模拟运行执行严重性分析

(2) 严重性分级：本节通过以下方式对严重性进行分级。

- 灾难性的:一个错误可能导致整个系统的失败或毁灭。
- 危急的:一个错误可能导致严重的损坏、主要性能的损 坏、主要系统的损坏，或者主要产品的失败。
- 边际性的:一个错误对性能、系统产生一个较小的损坏， 或推迟产品的完成日期。
- 较小的:一个错误并不产生任何的损坏，但需要不定时 地进行维护和修理。

软件体系结构的风险分析 – 步骤

4. 为构件和连接件开发启发式风险因子

- 主要是通过复杂性和严重性因素，为体系结构中的每 一个构件和连接件计算其启发式风险因子。
- 体系结构的每一个构件的启发式风险因子可以通过下式计算：

$$hrf_i = cpx_i \times svrty_i$$

$cpx_i \in [0, 1]$ 是第i个构件的动态复杂性
$svrty_i \in [0, 1]$ 是第i个构件的严重性级别

软件体系结构的风险分析 – 步骤

4. 为构件和连接件开发启发式风险因子

- 体系结构的每一个连接件的启发式风险因子可以通过下式计算：

$$hrf_{ij} = cpx_{ij} \times svrty_{ij}$$

$cpx_{ij} \in [0, 1]$是第i个构件到第j个构件之间的连接件的动态偶合度，可以通过$EC(C_i, C_j)$计算；
$svrty_{ij} \in [0,1]$是第i个构件到第j个构件之间的连接件的严重级别。

软件体系结构的风险分析 – 步骤

5. 建立用于风险评估的CDG

➢ CDG模型可以通过下述步骤构造：

(1) 通过估计每一个场景执行的频率，来估计每个场景执行的概率；

(2) 对每一个场景中的构件通过模拟报告来记录每一个构件的执行时间；

(3) 通过场景的使用概率和场景之间的迁移概率来计算构件之间的迁移概率；

(4) 通过模拟，为每个构件、连接件估计复杂性因子和严重性指标，
通过综合复杂性因子和严重性指标获取每个构件、连接 件的风险因子。

软件体系结构的风险分析 – 步骤

6. 通过图论中的算法执行风险评估和分析

- 可靠性风险分析算法：
  - 体系结构的风险因子可以通过聚合单个构件和连接件的风险因子来获得。

多选题 1分

设置

软件体系结构的风险分析说法正确的是：

A 风险评估应该是能够通过一个定量的方法对软件产品属性进行度量

B 体系结构动态风险评估方法用来评估执行中的软件体系结构的动态耦合度和动态复杂度

C 动态复杂度：用来度量在特定的执行场景中，两个相互连接的构件或连接件之间的活跃程度

D 构件依赖图（CDG）是从场景中开发出来的一个有向图，用于在体系结构级进行可靠性分析的概率模型，它是控制流图的一个扩展

E 动态耦合度：用来测量特定构件在给定场景下的动态行为

提交

基于体系结构描述的软件测试– 概述

软件体系结构的测试是用于检查软件设计的适用性，而不考虑软件的实现代码，所以基于实现的程序测试对软件体系结构测试并不适用

基于体系结构描述的软件测试 – 测试内容

- 与传统的软件测试一样，基于体系结构的软件测试也需要研究测试内容、测试准则、测试用例、测试充分性及测试方法等问题。
- 体系结构的描述必须满足一个基本的要求，即体系结构描述的各个部分必须相互一致，不能彼此冲突，因为体系结构主要关注系统的结构和组装，如果参与组装的各个部分之间彼此冲突，那么由此组装、细化和实现的系统一定不能工作。因此，体系结构的分析和测试主要考虑构件端口行为与连接件约束是否一致、兼容，单元间的消息是否一致、可达，相关端口是否可连接，体系结构风格是否可满足。
- 为了保证测试的充分性，必须对关联构件、连接件的所有端口行为和约束进行测试，即所有接口应该是连接的，数据流、控制流和命令应该是可达的，且在并发时应该保证无死锁发生。

基于体系结构描述的软件测试 – 测试准则

测试应覆盖所有的构件及各个构件的接口、各个连接件的接口、构件之间的直接连接、构件之间的间接连接

基于体系结构描述的软件测试 – 测试需求和测试用例的生成

- 构件或连接件内部消息的传递路径。
- 构件或连接件内部端口的执行顺序路径。
- 构件之间到连接件或连接件到构件的消息传递路径。
- 构件之间的直接连接路径。
- 构件之间的间接连接路径。
- 所有构件的连接路径。