基于构件的软件演化波及效应分析

郁 湧[1,2] 王丽霞[3] 赵 娜[1,2]
(云南大学软件学院 昆明 650091)[1] (云南省软件工程重点实验室 昆明 650091)[2]
(云南大学经济学院 昆明 650091)[3]

摘 要 随着新技术的采用和系统环境的变化,构件和软件系统的演化在所难免,演化会影响系统的整体行为。分析了基于构件的软件系统中的耦合性对演化波及效应的影响,对软件系统中构件内部各种依赖关系、构件与连接件之间的各种耦合关系进行矩阵表示,提出一种能够防止波及效应扩大化的软件系统动态演化的波及效应方法,实现了对软件系统的构件和连接件的演化波及效应研究。在基于构件的软件系统的动态演化中,可以根据波及效应的分析获得需要进行重新修改或演化的构件和连接件,从而保证动态演化的一致性和连续性。

关键词 基于构件的软件系统,软件演化,耦合,波及效应
中图法分类号 TP311 **文献标识码** A

Ripple-effect Analysis of Software Evolution Based on Component

YU Yong[1,2] WANG Li-xia[3] ZHAO Na[1,2]
(School of Software, Yunnan University, Kunming 650091, China)[1]
(Key Laboratory of Software Engineering of Yunnan Province, Kunming 650091, China)[2]
(School of Economics, Yunnan University, Kunming 650091, China)[3]

Abstract With the adoption of new technology and the change of the system environment, the evolution of components and software systems is inevitable, and the evolution will affect the overall behavior of the system. This papere analyzed the impact of the coupling on the evolution ripple effect in component-based software systems, and gave the matrix representation of the dependent relationship in component and the various coupling relationship between the components and connections of the software system. Based on matrix shift and calculation, ripple-effect of software evolution was analyzed. And an approach to ripple effect analysis of the dynamic evolution of the software system was presented, which can prevent magnification of the ripple effect. In the dynamic evolution of component-based software system, the related components and connectors can be obtained according to the analysis of the ripple effect, which can ensure the consistency and continuity of the dynamic evolution.

Keywords Software system based on component, Software evolution, Coupling, Ripple-effect

1 引言

演化性与构造性是软件的两个基本特性,软件进行渐变并达到所希望的形态就是软件演化[1]。软件演化由一系列复杂的变化活动组成,且引起变化的原因很多:基础设施的改变、功能需求的增加、高性能算法的发现和技术环境因素的变化等等。因此,对软件变化甚至演化进行理解和控制显得复杂困难。目前对软件演化的研究都流行从宏观层面入手,原因是可以避免过早陷入软件演化研究的复杂细节,可以在结构上确定每一个变化所影响的范围。随着软件功能的变更和环境因素的变化,软件系统的变化如构件的增加、替换、删除等就在所难免,但这种变化蕴含着波及效应,即软件系统中一部分的演化可能会波及到其它部分,因此需要对软件系统中的演化波及效应进行分析。

在演化波及效应方面,早在1978年,Yau就将模块变化的影响在软件模块之间的传播称为"变化传播"[2];Bohner提出了波及效应来描述软件变换的影响,并用可达矩阵的概念对软件变化进行了简单阐述,但没有给出组成软件的要素对软件贡献大小的概念[3,4];Baxter等人提出了通过将设计信息延伸到维护过程中来掌握软件的变化[5];Chiang提出了软件转换的增量式快速原型,并用其来形式化用户需求和规约,进而构造和管理软件的演化[6];Erich等人提出了软件变化的可视化方法[7];Zeng提出通过智能工作流技术来获取软件的易变性[8],但这些方法都偏向于定性的过程分析。在国内,王映辉等人研究建立了基于软件系统的构件-连接件模型,建立了系统关系矩阵和可达矩阵,实现了对软件系统静态和动态演

到稿日期:2013-01-22 返修日期:2013-05-02 本文受国家自然科学基金项目(61262024),云南省科技厅面上项目(2012FB119),云南省软件工程重点实验室面上项目(2012SE305),云南省教育厅科研项目(2011Y120)资助。

郁 湧(1980—),男,博士,讲师,主要研究方向为软件工程、可信软件,E-mail:yuy1219@163.com;王丽霞(1962—),女,硕士,副教授,主要研究方向为软件工程;赵 娜(1982—),女,博士,讲师,主要研究方向为软件工程。

化的定量分析[9]；王银坤等基于测试系统演化开发的军用ATS开发方法，针对测试系统演化开发活动中的波及效应提出了一种基于构件模型的方法[10]；黄翰等以软件体系结构作为软件的蓝图和支撑，设计了复合信息矩阵模型作为软件体系结构演化波及效应分析的新工具[11]。

尽管已经存在一些关于波及效应的研究方法，但是仅仅基于构件之间关系的波及效应分析方法还有很多不足，容易导致波及效应分析的扩大化。本文介绍了基于构件的软件系统中的耦合性对演化波及效应的影响，对基于构件的软件系统中各种耦合关系进行矩阵表示；并在分析软件系统中构件内部各种依赖关系和不同构件之间的耦合性对演化波及效应的影响的基础之上，提出一种能够防止波及效应扩大化的软件系统动态演化的波及效应方法。在基于构件的软件系统的动态演化中，可以根据波及效应的分析获得需要进行重新修改或演化的构件和连接件，从而保证动态演化的一致性和连续性。

2 耦合性对演化波及效应的影响

随着新技术的采用和系统环境的变化，构件演化会影响系统的整体行为，使系统产生不一致的情况。这时，必须分析某一构件演化对系统中其它构件的影响，以便于对其它构件进行相应的调整，从而确保系统的一致性。

构件通过它们之间的相互连接发生交互，要分析构件演化对系统的影响，就必须根据相互连接的构件形成的网络来分析构件之间的耦合关系。由于构件间的耦合性可能不具有传递性，因此不能直接依靠对构件的耦合邻接矩阵相乘的方式来分析构件演化的波及效应。

但由构件的交互模型知，构件的输入接口可能有赖于它自身的输出接口，而输出接口又依赖于其它构件的输入接口。这种依赖关系具有传递性，因此当一个构件演化时，可以通过构件中接口之间的依赖关系和构件之间的耦合关系闭包来找出可能受影响的所有构件及对应的构件接口。然后，对受影响的各个构件进行相应的调整，即可将系统恢复到一致状态。

基于构件的软件系统波及效应分析是指由于修改软件系统中的部分构件而导致对软件系统其它部分产生的影响的分析。

3 基于构件的软件中各耦合关系的矩阵表示

3.1 构件和软件系统的代数表示

构件是特定应用领域的软件系统中具有一定规模、相对独立、可替换的重用单元，它具有较稳定的组成模式，在确定的系统中，完成一项确定的、可区分的功能，并遵从和提供一套接口以及这些接口的实现[12]。

在软件系统中，一个构件应该包括接口和实现。其中接口部分定义了构件所提供的功能并规范了功能的使用方法；而实现部分包括了构件所能提供的一系列相关操作。

因此，构件和软件系统可以定义如下：

定义 1 构件 C 是软件系统中承担一定功能的数据或计算单元。构件是一个三元组 $C=\langle Interface, Imp, Spec\rangle$，其中：

(1) $Interface=IP\cup OP$ 是构件接口的集合，IP 代表构件的输入接口，OP 代表构件的输出接口；

(2) Imp 是构件的实现部分，它是由一系列操作 $t_1, t_2, \cdots, t_n$ 组成的，每个操作完成一定的功能；

(3) $Spec$ 是构件的内部规约，主要用来描述构件中接口与实现之间、构件实现的各操作之间的约束关系。

在构件中，一个构件可拥有多个输入接口和输出接口，每个输入接口刻画一定的实现，外界可以通过输入接口请求构件的服务；构件输出接口是描述构件对外界的请求，当构件完成某一功能时，可能需要其它构件的协作，输出接口便指明了这种对外关系。

Imp 表示构件的实现体，是构件的具体实现部分。它是构件接口对应的操作的序列的集合，描述构件 C 能完成的计算或数据功能，各操作之间既可能顺序执行，也可能根据一定规则选择、循环或并行执行。操作 t 的执行需要一定的条件，当条件满足时，相关的操作就执行；同时一个操作执行后，它会使其它操作的条件成立。

软件系统包括构件、构件之间的交互关系、约束、构件和连接件构成的拓扑结构、设计原则与指导方针，它定义了构件、连接件类型和一系列关于它们如何组合的限制规则，可以把软件系统描述如下。

定义 2 软件系统 $S=\langle CN, LN, D, G, A_L, A_G\rangle$ 是一个六元组，其中：

(1) $CN=\{C_1, C_2, C_3, \cdots, C_n\}$ 为软件系统中构件的集合，其中 $C_i(i=1,2,\cdots,n)$ 表示软件系统 S 中的第 i 个构件；

(2) $LN=\{L_1, L_2, L_3, \cdots, L_m\}$ 为软件系统中连接件的集合；

(3) $D\subseteq S_1\cup S_2\cup\cdots\cup S_n$ 为非空有限类型集，称为软件系统中连接件所能传递的信息类型集合；

(4) $G\subseteq(S.OP\times LN)\cup(LN\times S.IP)$，可以表示软件系统中构件与连接件之间的耦合关系；其中 $S.IP=C_1.IP\cup C_2.IP\cup\cdots\cup C_n.IP$、$S.OP=C_1.OP\cup C_2.OP\cup\cdots\cup C_n.OP$ 分别表示系统中所有构件的输入、输出接口集的并集；

(5) $A_L: LN\rightarrow 2^D$，是 LN 到 D 的多集上的映射，$A_L(L)$ 表示连接件 L 所能传递的信息的类型集；

(6) $A_G: G\rightarrow D_{MS}$，是 G 到 D 的多重集上的映射，表示构件接口与连接件之间能传递的信息类型的组合，且满足若 $\forall g\in G, A_G(g)\in(A_L(LN(g)))_{MS}, A_G(g)\in(A_P(P(g)))_{MS}$，其中 $LN(g)$ 表示弧 g 所对应的连接件，$P(g)$ 表示弧 g 所对应的接口。

在软件系统中连接件 L 只起传递作用，而不对其传递的信息实施任何的操作。

3.2 软件系统耦合关系分析

在软件系统 $S=\langle CN, LN, D, G, A_L, A_G\rangle$ 中，构件的输入接口与输出接口之间，构件与连接件之间，甚至构件与构件之间并不是相互毫不相关的，它们之间存在着一定的耦合或者依赖的关系。

定义 3 对于软件系统 $S=\langle CN, LN, D, G, A_L, A_G\rangle$ 中的构件 C 的输出接口 op 和连接件 L，若满足条件：

$\langle op, L\rangle\in G, A_G(\langle op, L\rangle)\in(A_P(op))_{MS}$

则称构件 C 的输出接口 op 正向直接耦合于连接件 L。

定义 4 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C 的输出接口 op 和连接件 L，若输出接口 op 正向直接耦合于连接件 L，则称连接件 L 反向直接耦合于构件 C 的输出接口 op。

定义 5 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的任意构件 C 与连接件 L，若存在构件 C 中的输出接口 op 满足 op 正向直接耦合于连接件 L 的条件，则称构件 C 正向直接耦合于连接件 L。

定义 6 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的任意构件 C 与连接件 L，若构件 C 正向直接耦合于连接件 L，则称连接件 L 反向直接耦合于构件 C。

定义 7 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C 的输入接口 ip 和连接件 L，若满足条件：

$\langle L,ip\rangle\in G$，且 $A_G(\langle L,ip\rangle)\in(A_P(ip))_{MS}$

则称连接件 L 正向直接耦合于构件 C 的输入接口 ip。

定义 8 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C 的输入接口 ip 和连接件 L，若连接件 L 正向直接耦合于构件 C 的输入接口 ip，则称构件 C 的输入接口 ip 反向直接耦合于连接件 L。

定义 9 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的任意构件 C 与连接件 L，若存在构件 C 中的输入接口 ip 满足连接件 L 正向直接耦合于 ip 的条件，则称连接件 L 正向直接耦合于构件 C。

定义 10 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的任意构件 C 与连接件 L，若满足连接件 L 正向直接耦合于构件 C 的条件，则称构件 C 反向直接耦合于连接件 L。

定义 11 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C_1 的输出接口 op、构件 C_2 的输入接口 ip 和连接件 L，若满足条件：

(1)$\langle op,L\rangle\in G,\langle L,ip\rangle\in G$；

(2)输出接口 op 正向直接耦合于连接件 L；

(3)连接件 L 正向直接耦合于构件 C 的输入接口 ip。

则称构件 C_1 的输出接口 op 通过连接件 L 正向直接耦合于构件 C_2 的输入接口 ip，简称输出接口 op 正向直接耦合于输入接口 ip。

定义 12 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C_1 和构件 C_2，若存在构件 C_1 的输出接口 op 和构件 C_2 的输入接口 ip 以及连接件 L 使得输出接口 op 通过连接件 L 正向直接耦合于输入接口 ip，则称构件 C_1 正向直接耦合于构件 C_2。

定义 13 对于软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C_1 和构件 C_2，若构件 C_1 正向直接耦合于构件 C_2，则称构件 C_2 反向直接耦合于构件 C_1。

3.3 软件系统耦合关系的矩阵表示

因此，为了便于演化波及效应的分析，我们可以把它们之间的这些关系用相应的矩阵进行表示。

若在软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中有 n 个构件、m 个连接件，它们分别为构件 $C_1,C_2,\cdots,C_n$ 和连接件 $L_1,L_2,\cdots,L_m$；且对于构件 $C_i(i=1,2,\cdots,n)$，它的输入接口和输出接口分别为 $ip_{i1},ip_{i2},\cdots,ip_{iI_i}$ 和 $op_{i1},op_{i2},\cdots,op_{iO_i}$，其中 I_i 和 O_i 分别表示构件 C_i 的输入接口和输出接口的个数，则软件系统中构件与连接件之间、构件与构件之间和各构件中输入接口与输出接口之间耦合和依赖关系的相应矩阵表示如下。

定义 14 软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中构件与连接件之间的(正向)直接耦合关系邻接阵 $CN\text{-}LN$ 可表示为：

$$CN\text{-}LN=(c_{ij})_{n\times m}$$

式中，c_{ij} 表示软件系统 S 中的第 i 个构件 C_i 与第 j 个连接件 L_j 之间的直接耦合关系($i=1,2,\cdots,n;j=1,2,\cdots,m$)；且有

$$c_{ij}=\begin{cases}1, & \text{当 } C_i \text{ 与 } L_j \text{ 之间存在正向直接耦合关系}\\ 0, & \text{当 } C_i \text{ 与 } L_j \text{ 之间不存在正向直接耦合关系}\end{cases}$$

定义 15 软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中连接件与构件之间的(正向)直接耦合关系邻接矩阵 $LN\text{-}CN$ 可表示为：

$$LN\text{-}CN=(c_{ij})_{m\times n}$$

式中，c_{ij} 表示软件系统 S 中的第 i 个连接件 L_i 与第 j 个构件 C_j 之间的直接耦合关系($i=1,2,\cdots m;j=1,2,\cdots,n$)；且有

$$c_{ij}=\begin{cases}1, & \text{当 } L_i \text{ 与 } C_j \text{ 之间存在正向直接耦合关系}\\ 0, & \text{当 } L_i \text{ 与 } C_j \text{ 之间不存在正向直接耦合关系}\end{cases}$$

定义 16 软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中构件与构件之间的(正向)直接耦合关系邻接矩阵 $CN\text{-}CN$ 可表示为：

$$CN\text{-}CN=(c_{ij})_{n\times n}$$

式中，c_{ij} 表示软件系统 S 中的第 i 个构件 C_i 与第 j 个构件 C_j 之间的(正向)直接耦合关系($i=1,2,\cdots,n$)；且有

$$c_{ij}=\begin{cases}1, & \text{当 } C_i \text{ 与 } C_j \text{ 之间存在正向直接耦合关系}\\ 0, & \text{当 } C_i \text{ 与 } C_j \text{ 之间不存在正向直接耦合关系}\end{cases}$$

从构件与连接件的耦合性可以知道，$CN\text{-}LN$ 与 $LN\text{-}CN$ 之间并不是转置的关系。

从软件系统中构件与连接件之间的直接耦合关系的性质可以知道，由于构件的输入接口与输出接口之间不一定存在依赖关系，因此软件系统中的构件之间的耦合性也不一定具有传递关系，即：

矩阵 $CN\text{-}CN\neq CN\text{-}LN\times LN\text{-}CN$。

我们知道，在软件系统 S 中，由于构件之间的耦合关系不具有传递性，因此要分析基于耦合关系的软件系统演化波及效应，必须要考虑构件的输入接口和输出接口之间的依赖关系。

定义 17 软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的构件 C_i 的输入接口与输出接口之间的依赖关系矩阵 C_{ii} 表示为：

$$C_{ii}=(c_{kl})_{I_i\times O_i}$$

式中，c_{kl} 表示构件 C_i 的第 k 个输入接口 ip_{ik} 与第 l 个输出接口 op_{il} 之间的依赖关系($k=1,2,\cdots,I_i;l=1,2,\cdots,O_i$)；且有

$$c_{kl}=\begin{cases}1, & \text{当构件 } C_i \text{ 的输入接口 } ip_{ik} \text{ 与输出接口 } op_{il}\\ & \text{之间存在依赖关系}\\ 0, & \text{当构件 } C_i \text{ 的输入接口 } ip_{ik} \text{ 与输出接口 } op_{il}\\ & \text{之间不存在依赖关系}\end{cases}$$

定义 18 软件系统 $S=\langle CN,LN,D,G,A_L,A_G\rangle$ 中的第 i 个构件 C_i 的输出接口集与第 j 个构件 C_j 的输入接口集之间的直接耦合关系矩阵 C_iC_j 表示为：$C_iC_j=(c_{kl})_{O_i\times I_j}$，其中 c_{kl} 表示构件 C_i 的第 k 个输出接口 op_k 与构件 C_j 的第 l 个输入接口 ip_{il} 之间的直接耦合关系（$k=1,2,\cdots,O_i$；$l=1,2,\cdots,I_j$）；且有

$$c_{kl}=\begin{cases}1, & \text{当构件 } C_i \text{ 的输出接口 } op_k \text{ 与构件 } C_j \text{ 的输入接口 } ip_l \text{ 之间存在直接耦合关系}\\ 0, & \text{当构件 } C_i \text{ 的输出接口 } op_k \text{ 与构件 } C_j \text{ 的输入接口 } ip_l \text{ 之间不存在直接耦合关系}\end{cases}$$

对于矩阵 C_iC_j，如果 $i=j$，且 $C_iC_j\neq 0$，则表示构件 C_i 存在自耦合。

定义 19 软件系统 S 中的所有构件的输入接口与输出接口之间的依赖关系矩阵 R_1 可表示为：

$$R_1=\begin{vmatrix}C_{11} & O_{12} & \cdots & O_{1n}\\ C_{21} & O_{22} & \cdots & O_{2n}\\ \cdots & \cdots & \cdots & \cdots\\ C_{n1} & O_{n2} & \cdots & O_m\end{vmatrix}$$

其中矩阵 R_1 中的对角线上的矩阵 $C_{ii}=(c_{kl})_{I_i\times O_i}$（$i=1,2,\cdots,n$）分别是构件 C_i（$i=1,2,\cdots,n$）的输入接口与输出接口之间的依赖关系矩阵，而非对角线上的矩阵 O_{ij}（$i,j=1,2,\cdots,n$）是对应的零矩阵。

定义 20 软件系统 S 中的所有构件的输出接口集与输入接口集之间的直接耦合关系矩阵 R_2 可表示为：

$$R_2=\begin{vmatrix}C_1C_1 & C_1C_2 & \cdots & C_1C_n\\ C_2C_1 & C_2C_2 & \cdots & C_2C_n\\ \cdots & \cdots & \cdots & \cdots\\ C_nC_1 & C_nC_2 & \cdots & C_nC_n\end{vmatrix}$$

其中 R_2 中的每个 $C_iC_j=(c_{mn})_{O_i\times I_j}$（$i,j=1,2,\cdots,n$）分别是构件 C_i 的输出接口与构件 C_j 的输入接口之间的直接耦合关系矩阵；如果 $i=j$，且 $C_iC_j\neq 0$，则表示构件 C_i 存在自耦合。

4 演化波及效应分析

构件与连接件是基于构件的软件系统中的两大类构成部分，因此，对于基于构件的软件系统演化波及效应分析必须综合考虑这两方面的内容，才能得出较为全面合理的结论。

4.1 叉和与叉乘

在对软件系统中各关系矩阵进行转换前有必要先给出两个运算符：叉和⊕和叉乘⊗的定义。

定义 21 数的叉和⊕

若 x、$y=0$ 或 1，则 $x\oplus y$ 的值定义如下：

$$x\oplus y=\begin{cases}0, & \text{当 } x=y=0\\ 1, & \text{其它}\end{cases}$$

定义 22 向量的⊕

若 X、Y 分别是 n 维向量 $\langle x_1,x_2,\cdots,x_n\rangle$、$\langle y_1,y_2,\cdots,y_n\rangle$，且 $x_i,y_j=1$ 或 $0(i,j=1,2,\cdots,n)$，则向量 $X\oplus Y$ 定义如下：

$$X\oplus Y=\langle x_1\oplus y_1,x_2\oplus y_2,\cdots,x_n\oplus y_n\rangle$$

矩阵的⊕与向量的⊕的原理一致，故在此省略。

定义 23 数的叉乘⊗

若 x、$y=0$ 或 1，则 $x\otimes y$ 的值定义如下：

$$x\otimes y=\begin{cases}0, & \text{当 } x=y=1\\ 1, & \text{其它}\end{cases}$$

定义 24 向量的叉乘⊗

若 X、Y 分别是 n 维向量 $\langle x_1,x_2,\cdots,x_n\rangle$、$\langle y_1,y_2,\cdots,y_n\rangle$，且 $x_i,y_j=1$ 或 $0(i,j=1,2,\cdots,n)$，则向量 X 与向量 Y 的转置向量 Y' 的叉乘 $X\otimes Y'$ 定义如下：

$$X\otimes Y'=(x_1\otimes y_1)\otimes(x_2\otimes y_2)\otimes\cdots\otimes(x_n\otimes y_n)$$

当两个矩阵叉乘⊗时，其中对应的向量的叉乘原理与定义 24 相一致，故在此省略。

4.2 演化波及效应

由于构件间的耦合性可以根据构件内输入接口与输出接口之间的依赖关系和构件间输出接口与输入接口之间的直接耦合关系推导出，且它们之间的关系具有传递性，因此可以对矩阵 R 不断地求叉乘，然后对各个叉乘的结果求叉和，从而可以得到相应系统的演化波及效应矩阵。

定义 25 令矩阵 $R=R_1\otimes R_2$，则 R 表示软件系统中所有构件之间输入接口的耦合关系矩阵。

在矩阵 R 的叉乘⊗中，我们令 $R^2=R\otimes R$，$R^3=R\otimes R\otimes R$，$\cdots$，$R^n=R\otimes R\otimes\cdots\otimes R$（即 n 个 R 求⊗乘），$\cdots$。

软件系统中所有构件之间输入接口的耦合关系具有传递性，于是，其传递闭包所对应的矩阵 $R^+=R\oplus R^2\oplus R^3\oplus\cdots$。

对耦合关系的传递闭包所对应的关系矩阵 R^+ 按照各构件的输入接口数大小的形式进行分块，可以把 R^+ 分为 n^2 个分块矩阵，如下所示：

$$R^+=\begin{vmatrix}R_{11} & R_{12} & \cdots & R_{1n}\\ R_{21} & R_{22} & \cdots & R_{2n}\\ \cdots & \cdots & \cdots & \cdots\\ R_{n1} & R_{n2} & \cdots & R_m\end{vmatrix}$$

其中各个分块矩阵 R_{ij}（$i,j=1,2,\cdots,n$）为 $I_i\times I_j$ 的矩阵，令

$$R_{ij}=\begin{vmatrix}r_{11} & r_{12} & \cdots & r_{1I_j}\\ r_{21} & r_{22} & \cdots & r_{2I_j}\\ \cdots & \cdots & \cdots & \cdots\\ r_{I_i1} & r_{I_i2} & \cdots & r_{I_iI_j}\end{vmatrix}$$

则矩阵 R^+ 就是系统 S 的演化波及效应矩阵；而矩阵 R_{ij}（$i,j=1,2,\cdots,n$）就是构件 C_i 的演化对构件 C_j 的波及效应矩阵，其演化波及影响关系如下：当矩阵 R_{ij} 中的元素全部为 0 时，表示构件 C_i 的演化对构件 C_j 没有产生任何波及效应；若矩阵 R_{ij} 中的元素 $r_{kl}=1$，表示构件 C_i 的输入接口 ip_{ik} 及对应实现部分的演化对构件 C_j 的输入接口 ip_{jl} 及对应实现部分产生波及效应。若矩阵 R_{ij} 中的元素为 1 的个数大于 1，则说明构件 C_i 的演化对构件 C_j 产生的波及效应是多重的，从而我们可以对其波及效应进行度量。

4.3 演化对软件系统中构件的波及效应

令 I_i 维单位向量 $e_i=(1,1,\cdots,1)$，即 e_i 中 1 的个数为 I_i；向量 e_i 的转置向量为 $\overline{e_i}$；令 I_j 维单位向量 $e_j=(1,1,\cdots,$

1)，即 e_j 中 1 的个数为 I_j；向量 e_j 的转置向量为 $\overline{e_j}$。

定义 26 对于矩阵 R_{ij} 的第 k 行向量 $(r_{k1}, r_{k2}, \cdots, r_{kl_j})$，令 $q=(r_{k1}, r_{k2}, \cdots, r_{kl_j}) \times \overline{e_j}$，则构件 C_i 的输入接口 ip_{ik} 及对应实现部分的演化对构件 C_j 的基于输入接口的波及效应率 $A_{pk}=q/I_j$。

A_{pk} 表示构件 C_i 的输入接口 ip_{ik} 及对应实现部分的演化对构件 C_j 的输入接口的波及影响的程度。A_{pk} 的值越大，说明影响的程度越大。

定义 27 令向量 $m_i=e_i \otimes R_{ij}$，则 $q=m_i \times \overline{e_j}$，则构件 C_i 的演化对构件 C_j 的基于输入接口的波及效应率 $A_{C_{ij}}=q/I_j$。

$A_{C_{ij}}$ 表示构件 C_i 的演化对构件 C_j 的输入接口的整体波及影响的程度。$A_{C_{ij}}$ 的值越大，说明影响的程度越大。若 $A_{C_{ij}}=0$，说明构件 C_i 的演化对构件 C_j 没有任何波及影响；若 $A_{Cij}=1$，说明构件 C_i 的演化对构件 C_j 的每部分都有影响。

同时，通过可达矩阵非常容易界定某一构件变化所影响的其他构件。进而，当某一组构件发生变化时，可以圈定被影响或波及到的其他构件的范围。当构件 C_i 演化时，矩阵 R_{ij} $(j=1,2,\cdots,n)$ 中具有非 0 元素的矩阵个数就是构件 C_i 演化对系统 S 中其他构件影响的个数。我们可以按照大小对其进行排序，其大小代表相应构件演化对系统的影响程度。当然，这个数值越大，该构件对软件系统的影响就越大。

4.4 演化对软件系统中连接件的波及效应

从软件系统的原理可知，构件之间的连接是通过连接件来实现的。我们可以把连接件看成是构件之间进行交互的桥梁，因此可以在系统中的构件之间演化波及效应的基础之上得出构件的演化对连接件的波及效应。

定义 28 构件输出接口与连接件的耦合关系矩阵

软件系统 S 中构件 C_i 的输出接口与连接件之间的耦合关系矩阵 C_iLN 可表示为：

$$C_iLN=(c_{kl})_{O_i \times m}$$

式中，c_{kl} 表示构件 C_i 的输出接口 op_{ik} 与连接件 L_l 之间的直接耦合关系 $(k=1,2,\cdots,O_i; l=1,2,\cdots,m)$；并且

$$c_{kl}=\begin{cases}1, & \text{当 } C_i \text{ 的输出接口 } op_{ik} \text{ 与连接件 } L_l \\ & \text{之间存在直接耦合关系} \\ 0, & \text{当 } C_i \text{ 的输出接口 } op_{ik} \text{ 与连接件 } L_l \\ & \text{之间不存在直接耦合关系}\end{cases}$$

定义 29 令 $R_{C_iLN}=C_i \otimes C_iLN \otimes R_{i1} \otimes C_1C_1 \otimes C_1LN \otimes R_{i2} \otimes C_2C_2 \otimes C_2LN \otimes \cdots \otimes R_{in} \otimes C_nC_n \otimes C_nLN$，则 R_{C_iLN} 为 $I_i \times m$ 阶的矩阵：

$$R_{C_iLN}=\begin{vmatrix} r_{11} & r_{12} & \cdots & r_{1m} \\ r_{21} & r_{22} & \cdots & r_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ r_{I_i1} & r_{I_i2} & \cdots & r_{I_im} \end{vmatrix}$$

它描述了构件 C_i 的演化对软件系统中连接件的波及影响，其描述如下：

矩阵 R_{C_iLN} 中的元素全为 0，则说明构件 C_i 的演化对软件系统中的任何连接件都没有波及影响；在矩阵 R_{C_iLN} 中，若第 l 列 $(l=1,2,\cdots,m)$ 中存在值为 1 的元素，则说明构件 C_i 的演化对连接件 L_l 可能有波及影响，否则构件 C_i 的演化对连接件 L_l 无影响。若元素 $r_{kl}=1(k=1,2,\cdots,I_i; l=1,2,\cdots,m)$，则说明构件 C_i 的输入接口 ip_k 及对应实现部分的演化对连接件 L_l 有波及影响。

结束语 随着新技术的采用和系统环境的变化，构件和软件系统的演化在所难免。在基于构件的软件系统中，构件演化会影响系统的整体行为，使系统产生不一致的情况。构件通过它们之间的相互连接发生交互，要分析构件演化对系统的影响，就必须根据相互连接的构件形成的网络来分析构件之间的耦合关系。同时，必须分析构件演化对系统中其它构件的影响，以便于对其它构件进行相应的调整，从而确保系统的一致性。本文分析了软件系统中的耦合性对演化波及效应的影响，对软件系统中各种耦合关系进行矩阵表示，并在此基础之上分析了软件系统中演化对构件和连接件的波及效应。在基于构件的软件系统的动态演化中，可以根据波及效应的分析获得需要重新修改或演化的构件和连接件，从而保证动态演化的一致性和连续性。

参 考 文 献

[1] 杨芙清. 软件工程技术发展思索[J]. 软件学报，2005，16(1)：1-7

[2] Yau S S，Collofello J S，McGregor T M. Ripple effect analysis of software maintenance[C]//Proc. of the Computer Software and Applications Conf. (COMPSAC'78). Piscataway：IEEE Computer Society Press，1978：60-65

[3] Ryder B G，Tip F. Change impact analysis for object-oriented programs[C]//Proc. of 2001 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering. New York：ACM Press，2001：46-53

[4] Bohner S A. Software change impacts：An evolving perspective [C]//Proc. of the Int'l Conf. of Software Maintenance(ICSM 2002). Washington：IEEE，2002：263-272

[5] Baxter I D，Pidgeon C W. Software change through design maintenance[C]//Proc. of the Int'l Conf. of Software Maintenance. Washington：IEEE，1997：250-259

[6] Chiang C C，Urban J E. Incremental elicitation and formalization of user requirements through rapid prototyping via software transformations[C]//Proc. of the 20th Int'l Computer Software and Applications Conf. (COMPSAC'96). Washington：IEEE，1996：240-245

[7] Erich S G，Graves T L，Karr A F，et al. Visualizing software changes[J]. IEEE Trans. on Software Engineering，2002，28(4)：396-412

[8] Zeng D D，Zhao J L. Achieving software flexibility via intelligent workflow techniques[C]//Proc. of the 35th Annual Hawaii Int'l Conf. on System Sciences(HICSS-35 2002). Washington：IEEE，2002：606-615

[9] 王映辉，张世琨，刘瑜，等. 基于可达矩阵的软件系统演化波及效应分析[J]. 软件学报，2004，15(8)：1107-1115

[10] 王银坤，肖明清，王学奇. 构件模型的测试系统演化开发波及效应分析[J]. 空军工程大学学报：自然科学版，2008，19(2)：60-63

[11] 黄翰，郝志峰，陈明，等. 基于复合信息矩阵的软件体系结构演化波及效应分析[J]. 计算机科学，2007，34(2)：260-263

[12] 王志坚，费玉奎，娄渊清. 软件构件技术及其应用[M]. 北京：科学出版社，2005