基于过程模式的软件过程框架

梁义芝[1,2] 王延章[1] 刘云飞[3] 缪旭东[2]

(大连理工大学信息与决策技术研究所 大连116024)[1]

(海军大连舰艇学院作战软件研究中心 大连116018)[2] (海军大连舰艇学院教育技术中心 大连116018)[3]

Process Pattern Based Software Process Framework

LIANG Yi-Zhi[1,2] WANG Yan-Zhang[1] LIU Yun-Fei[3] MIU Xu-Dong[2]

(Institute of Information and Decision Technology, Dalian University of Technology, Dalian 116024)[1]

(Combat Software Research Center, Dalian Naval Academy, Dalian 116018)[2]

(Science and Research Department, Dalian Naval Academy, Dalian 116018)[3]

Abstract Process pattern is an excellent method to express software process knowledge, it can express process knowledge in different granularity and realize the non-gap connection of process knowledge, it supports the reuse and persistent improvement of process knowledge. In this paper, we put forward a process Pattern Based Software Process Framework(PB-SPF), its three layer architecture has realized the high abstraction of software process and the separation of its contents, so it have good reusability and adaptability. We can use it as a base for the research of software process, the building and enactment of process model.

Keywords Software process pattern, Software process knowledge, Software process framework

1 引言

认知学研究表明,人类解决问题的过程是比较复杂的。当面临需要解决的问题时,人们首先会考虑该问题以前是否已经解决了,或者是否能够被抽象成一个与已经解决的某个问题相似的问题,人们惯于在解决实际问题时使用已有的经验,根据经验去认识现有事物,进而选择信息、解释事物,经验是人类的宝贵财富。一个新手通常只能参考已有的例子解决问题,通过研究一个具体问题的解决方法、所使用的特殊策略以及在已解决的问题之间建立联系,可以使经验得到积累,解决问题的能力和效率得到提高。在软件过程技术研究中考虑人的认知特性是一个非常值得研究的课题。

软件过程是一个复杂的、动态变化的、不断改进的过程,不仅与软件技术水平、软件开发方法相关,还与软件组织的企业文化相关,其实施过程会受到许多外界因素的影响。为软件组织和项目建立软件过程模型需要各种软件过程知识与技术,如任务的时序安排、资源分配、产品管理等。软件过程知识与经验的积累以及通过具体过程的实施来完善和改进过程知识对于软件组织提高软件过程能力是非常重要的。

软件过程模式是软件开发过程经验的总结与结构化表示,一个过程模式一般主要包括三部分内容:给定上下文、问题以及问题的解[1]。其中给定上下文(Context)是指产生某一问题的情形;问题(Problem)是指在给定上下文情形下反复出现的待解问题,它包含一系列设计要素(Forces),即解决该问题需考虑的各个方面,如解的部分必须满足的要求、必须考虑的约束条件以及解应该具备的特性等;解(Solution)是指对这一类问题反复验证的成功的解决方案。

软件过程模式描述了软件过程中经过验证的成功的方法和一系列活动,是软件过程知识在不同知识粒度上按模式概念体系的抽象,其目的是尽可能复用成功的软件过程经验以解决新问题并在解决问题的过程中积累新知识、完善已有知识。基于过程模式的软件过程建模有以下优点:

(1)在过程模型粒度的控制上具有良好的伸缩性。不同粒度的过程模型支持不同层次的过程管理和实现人员、支持不同层次的过程任务的实施。

(2)支持过程知识的总结、积累和复用。

(3)支持过程模型的动态生成。

(4)支持过程的持续改进。

2 基于过程模式的软件过程框架(PB-SPF)

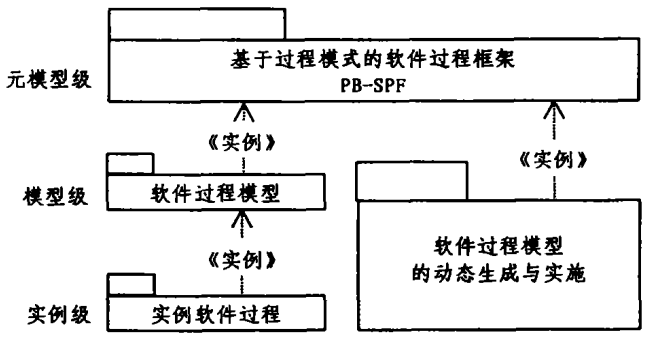


图1 基于过程模式的软件过程框架与软件过程建模与实施关系图

基于过程模式的软件过程框架 PB-SPF 定义了过程知识的描述模型,即包括过程模式在内的过程实体模型及过程实体之间的关系模型;定义了过程知识的使用方法,即作用于过

梁义芝 副教授,博士生,主要研究领域为决策支持系统、软件工程。王延章 教授,博士生导师,主要研究领域为管理决策支持系统、经济系统分析、数学规划。刘云飞 高工,主要研究领域为网络技术,多媒体技术,软件工程。缪旭东 副教授,主要研究领域为决策支持系统、软件工程。

程实体模型及其关系模型之上的操作。该框架支持过程知识的表示、集成、维护和重用，支持软件过程静态模型的建立与过程模型的动态生成，是基于过程模式的软件过程建模与实施的基础，图1为用 UML 表示的基于过程模式的软件过程框架与软件过程建模、实施的关系图。

2.1 PB-SPF 的基本思想

简单地说，软件过程是指在软件生存周期中所进行的一系列具有目的性的活动。软件过程中的活动实际是生产软件产品的过程，因此软件过程活动将涉及到为完成一定任务而执行活动的人员、活动需使用的资源以及活动应生产出的产品，我们可以将软件过程描述为在软件生存周期中，承担某些角色的人员，为完成指定任务，使用一定的资源，执行一系列相关活动，生产出软件产品的过程，由此软件过程中的过程实体应包括角色、任务、活动、活动资源和活动产品。图2为用 UML 描述的软件过程实体关系图。

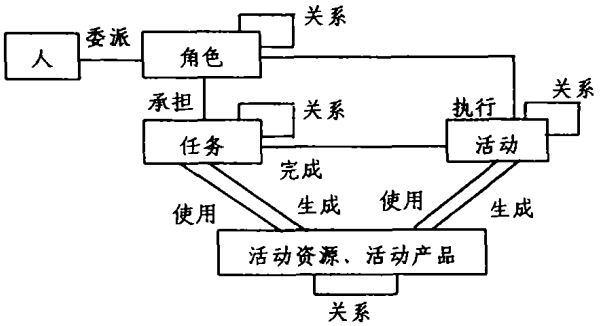


图2 软件过程实体关系图

将不同过程实体之间的关系称为异类实体关系，异类实体关系相对简单，可将其统一为具有一定含义的关联关系，如活动与角色之间为执行(或主体)关系，活动与资源之间为使用(或输入)关系，活动与产品之间为生成(或输出)关系，任何一个过程实体均可通过关联关系找到与其相关的其它过程实体。

将相同过程实体之间的关系称为同类实体关系，同类实体关系相对复杂，如活动可以分解，形成层次嵌套关系，相同层次的活动之间又存在时间偏序关系、并行关系等，这些关系又引起了资源约束等问题。

同类实体关系的复杂性是软件过程复杂性的主要体现，因此软件过程建模的主要任务是清晰、直观地描述同类实体关系。在一个过程模型中描述所有实体的同类实体关系会使模型非常复杂，脉络不清晰，不易于理解和使用，因此，一般是以某一过程实体为主线，即以描述某一过程实体的同类实体关系为主建立过程模型，由此产生了以下过程建模方法[2]：

(1)以角色为中心的建模方法：建模时首先确定软件组织结构中的角色及各角色应承担的任务，然后描述角色之间的关系，形成角色关系网，再通过与角色相关联的异类实体关系找到与各角色有关的其它过程实体，如角色执行的活动、相关产品、资源等，充实角色关系网，建立以角色为中心的过程模型。

(2)以产品为中心的建模方法：建模时首先确定过程应生产出的产品，然后描述产品之间的关系，再通过与产品相关联的异类实体关系找到与各产品有关的其它过程实体，如生产产品的活动、涉及的角色、所需资源等，建立以产品为中心的过程模型。

(3)以活动为中心的建模方法：建模时首先确定过程所涉及的所有活动，然后描述活动之间的关系，形成活动链，再通过与活动相关联的异类实体关系找到与各活动有关的其它过程实体，如执行活动的角色、活动产生的产品、活动需要的资源等，充实活动链，建立以活动为中心的过程模型。

以角色为中心的过程模型能够描述过程组织方面的信息，便于各有关人员明确自己的任务，便于人员管理与控制，一般适用于高层管理；以产品为中心的过程模型便于对过程结点(基线)的控制与管理，一般适用于高层管理；以活动为中心的过程模型能够描述软件过程的工作流程，易于形成对整个软件过程的整体认识，便于不同层次上的过程管理、过程执行、过程监视和过程控制，便于进行过程度量，进而便于过程改进和过程能力提高，另外，对人员的管理可以细化到其完成任务的具体过程，对产品的管理也可细化到生产产品的具体过程，以提高对人员和产品管理的可操作性，因此软件过程建模中对工作流程(即活动及其关系)的研究和描述尤其重要。

目前已有多种主要描述工作流程的过程模型，其中通用模型有瀑布模型、螺旋模型等；针对性较强的模型如统一软件开发过程[3]、the V-Modell 97[4]等，即详细定义了软件过程中的主要活动及其关系，又提供了过程模型的描述和应用技术，这些模型各有优点和缺点，我们可以在实际应用中借鉴使用；另外各软件组织积累了大量的过程管理经验并有适合于本组织的基本软件过程，这些都是可以总结、积累、重用和改进的软件过程知识，如何将这些知识以结构化的易于使用的方式表示出来并提供使用、改进知识的方法(操作)是目前软件过程研究中亟需解决的问题，我们认为软件过程模式是解决该问题的一个很好的方法。

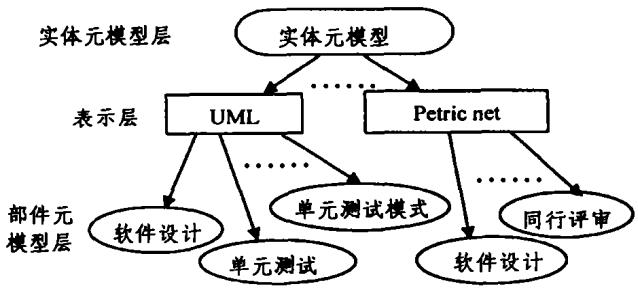


图3 PB-SPF 的三级体系结构

软件过程模式可以表示不同粒度的软件过程知识，可实现知识的无缝链接，便于过程知识的使用与改进，并可实现软件过程模型的复用。为方便过程模式的描述、存贮与使用，便于利用过程模式生成和实施软件过程模型，我们设计并初步实现了基于过程模式的软件过程框架(PB-SPF)，设计 PB-SPF 我们主要考虑以下两方面问题：一是 PB-SPF 要有一个良好的体系结构，便于扩充与维护；二是 PB-SPF 要有坚实的理论基础并具备实际使用价值，为此我们设计了如图3所示的 PB-SPF 的三级体系结构，其中第一级为实体元模型层(Entity meta-model layer)，定义了过程模式的描述模型、存贮结构、使用方法，定义了实体模型、实体关系模型(包括同类和异类实体关系模型)及作用于过程实体上的操作，定义了一些基本约束和一致性准则；第二级为表示层(Representation layer)，定义了实体元模型的表示方法，如 Petri 网、UML 等；第三级为部件元模型层(Component meta-model layer)，部件元模型是用表示层的表示方法描述的实体元模型，如用

UML 描述的继承实体元模型基本属性的软件单元测试元模型类等，其实例是组成过程模型的基本部件。

PB-SPF 的三级体系结构很好地实现了软件过程的抽象及在此基础之上的内容分离，其第一级描述了软件过程的基本内容、基本关系和基本操作，与具体的表示方法无关，具有较强的可重用性和普遍适用性，本文将主要介绍 PB-SPF 中的实体元模型。

2.2 PB-SPF 的实体元模型

PB-SPF 的实体元模型是软件过程实体及其关系的高度抽象。为简化起见，将图2描述的实体关系中的任务用完成该任务的活动替代，从而将软件过程描述为在软件生存周期中，承担某些角色的人员，使用一定的资源，执行一系列相关活动，生产出软件产品的过程；将软件过程中的重要实体活动与实现该活动的过程模式一一对应，形成如图4所示的实体元模型。

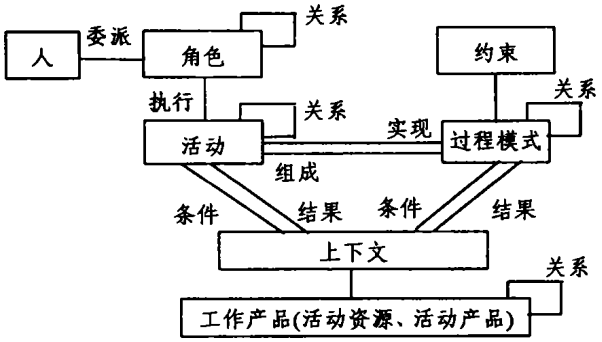


图4 PB-SPF 的实体元模型

过程模式是活动在一定约束条件下执行时应遵循的基本模式，由于 PB-SPF 可以集成多种不同的过程模型以及约束条件的不同，一个活动可以对应多个过程模式；活动描述做什么，过程模式描述应如何做，一个活动可按某一过程模式实现，该过程模式又描述了相互关联的一组子活动，这些子活动又可与相应的过程模式对应，如此循环往复，形成活动与过程模式之间的层次嵌套结构，活动与过程模式的分解层次(知识粒度)视管理层次而定，高层管理知识粒度较大，不必深度分解，具体执行部门则需分解到最底层，到不能分解为止，使过程模型具有较强的可操作性；每个活动以及每个过程模式在初始上下文满足的条件下产生结果上下文，初始上下文是活动及过程模式执行的前提，结果上下文是活动及过程模式执行的结果，活动的上下文和实现该活动的过程模式的上下文必须一致；上下文代表了过程执行的进度或状态，工作产品是最基本的上下文；活动由指定的角色执行，角色由对应的人员承担。

PB-SPF 的实体元模型有两种视图：一种为结构视图(Structure View)，描述过程实体、异类实体关系及操作，对应过程模型的静态部分；一种为行为视图(Behavior View)，描述同类实体关系及操作，对应过程模型的动态部分。下面将分别介绍这两种视图。

2.2.1 PB-SPF 实体元模型的结构视图

定义1(结构视图) Sview::=(Crol,Cact,Cpat,Cwpro; Rra,Rap,Rpa,Rawu,Rawc,Rpwu,Rpwc)为 PB-SPF 实体元模型的结构视图，简称结构视图。

其中：(1)Crol 表示过程角色类；(2)Cact 表示过程活动类；(3)Cpat 表示过程模式类；(4)Cwpro 表示过程工作产品类；(5)Rra 表示过程角色与过程活动的关联关系：执行关系；(6)Rap 表示过程活动与过程模式的关联关系：实现关系；(7)Rpa 表示过程模式与过程活动的关联关系：组成关系；(8)Rawu 表示过程活动与过程工作产品的关联关系：使用关系；(9)Rawc 表示过程活动与过程工作产品的关联关系：生成关系；(10)Rpwu 表示过程模式与过程工作产品的关联关系：使用关系；(11)Rpwc 表示过程模式与过程工作产品的关联关系：生成关系。

关联关系采用 UML 中关联关系的定义和语义。

定义2(约束集) SETconstr::=(CONSTRra,CONSTRap,CONSTRaw)为 Sview 上的约束集，其中的约束分别为：

(1)CONSTRra：

任何角色都必须执行一定的活动，反之任何活动都必须由对应的角色执行，即：

若 SSubCrol 为 Crol 的子类的集合，SSubCact 为 Cact 的子类的集合

则①(∀ SubCrol)(SubCrol∈SSubCrol→(∃ SubCact) SubCact∈SSubCact ∧ (SubCrol Rra SubCact))

②(∀ SubCact)(SubCact∈SSubCact→(∃ SubCrol) SubCrol∈SSubCrol ∧ (SubCact Rra^c SubCrol))

Rra^c 为 Rra 的逆关系

(2)CONSTRap：

活动可与0个或多个过程模式存在实现关系，即：

活动可以与多个过程模式相对应，也可以没有与之对应的过程模式

(3)CONSTRaw：

任何活动都必须使用活动资源，又必须生成活动产品，即：

若 SSubCact 为 Cact 的子类的集合，SSubCwpro 为 Cwpro 的子类的集合，

则(∀ SubCact)(SubCact∈SSubCact→(∃ SubCwpro1, SubCwpro2) ∧ SubCwpro1∈SSubCwpro ∧ SubCwpro2∈SSubCwpro ∧ (SubCact Rawu SubCwpro1) ∧ (SubCact Rawc SubCwpro2))

定义3(一致性准则集) SETconsis::=(CONSISap, CONSISpa)为 Sview 上的一致性准则集，其中的一致性准则分别为：

(1)CONSISap：

存在实现关系的活动和过程模式必须使用相同的活动资源，生成相同的活动产品，即：

若 SSubCact 为 Cact 的子类的集合，SSubCwpro 为 Cwpro 的子类的集合，

SSubCpat 为 Cpat 的子类的集合

则：(∃ SubCact)(∃ SubCpat)(SubCact∈SSubCact ∧ SubCpat∈SSubCpat ∧ (SubCact Rap SubCpat) ∧ (∃ SubCwpro1,SubCwpro2)(SubCwpro1∈SSubCwpro ∧ SubCwpro2∈SSubCwpro ∧ (SubCact Rawu SubCwpro1) ∧ (SubCact Rawc SubCwpro2)))→((SubCpat Rpwu SubCwpro1) ∧ (SubCpat Rpwc SubCwpro2))

且

(∃ SubCact)(∃ SubCpat)(SubCact∈SSubCact ∧ SubCpat ∈ SSubCpat ∧ (SubCact Rap SubCpat) ∧ (∃ SubCwpro1, SubCwpro2)(SubCwpro1∈SSubCwpro ∧

SubCwpro2∈SSubCwpro ∧(SubCpat Rpwu SubCwpro1)∧(SubCpat Rpwc SubCwpro2)))→((SubCact Rawu SubCwpro1)∧(SubCact Rawc SubCwpro2))

(2)CONSISpa：

与过程模式存在组成关系的活动组成一个活动集合，集合中第一个执行的活动所使用的活动资源集合一定是该模式的活动资源集合的子集，集合中最后一个执行的活动所生成的活动产品集合一定是该模式的活动产品集合的子集，即：

若 SSubCact 为 Cact 的子类的集合，SSubCwpro 为 Cwpro 的子类的集合，

SSubCpat 为 Cpat 的子类的集合

则①(∃ SubCact)(∃ SubCpat)(SubCact∈SSubCact ∧ SubCpat∈SSubCpat ∧(SubCpat Rpa SubCact)∧(Order(SubCact)=first)∧(∃ SubCwpro1)(SubCwpro1∈SSubCwpro ∧(SubCact Rawu SubCwpro1)))→(SubCpat Rpwu SubCwpro1)

②(∃ SubCact)(∃ SubCpat)(SubCact∈SSubCact ∧ SubCpat∈SSubCpat ∧(SubCpat Rpa SubCact)∧(Order(SubCact)=last)∧(∃ SubCwpro1)(SubCwpro1∈SSubCwpro ∧(SubCact Rawc SubCwpro1)))→(SubCpat Rpwc SubCwpro1)

其中 Order 为求出活动的执行顺序的函数。

2.2.1.1 过程角色类

过程角色类的属性抽象如下：

Rname 角色类名	Rdesc 角色描述	Rtask 承担任务	Ract 执行活动	Rorg 隶属组织	Rsta 角色状态	Rtool 相关工具	Rpro 相关产品
					正在工作、已完成工作等		

过程角色类的操作定义如下：

OrGETact(Rname,Rra) 根据执行关系 Rra 获取角色执行的活动

OrGETwprou(OrGETact(Rname,Rra),Rawu)根据执行关系 Rra 和使用关系 Rawu 获取角色执行活动需使用的资源

OrGETwproc(OrGETact(Rname,Rra),Rawc)根据执行关系 Rra 和生成关系 Rawc 获取角色通过执行活动获得的输出产品

建立以角色为中心的过程模型时，角色之间的关系描述清楚并形成角色关系网后(由 PB-SPF 的行为视图完成)，通过以上操作可获取角色对应的活动、资源和产品以充实角色关系网，即 PB-SPF 支持建立以角色为中心的过程模型。

2.2.1.2 过程活动类

过程活动类的属性抽象如下：

Aname 动类名	Adesc 活动描述	Arol 活动角色	Apat 对应的过程模式集合	Aini 初始上下文	Ares 结果上下文	Asta 活动状态	Atool 相关工具	Atime 时间长度	Ast 起始时间	Aend 终止时间
						挂起、正在执行等				

过程活动类的操作定义如下：

OaGETrol(Aname,Rra) 根据执行关系 Rra 获取执行活动的角色

OaGETpat(Aname,Rap) 根据实现关系 Rap 获取活动对应的过程模式集合

OaGETwprou(Aname,Rawu) 根据使用关系 Rawu 获取活动的输入资源

OaGETwproc(Aname,Rawc) 根据生成关系 Rawc 获取活动的输出产品

建立以活动为中心的过程模型时，活动之间的关系描述清楚并形成活动链后(由 PB-SPF 的行为视图完成)，通过以上操作可获取活动对应的角色、资源和产品以充实活动链，也可获取活动对应的过程模式从而以成功的经验指导活动的执行，即 PB-SPF 支持建立以活动为中心的过程模型。

2.2.1.3 过程模式类

过程模式类的描述模板为：

Pname 过程模式名称

Pcate 过程模式类别：通用过程模式(从通用模型如瀑布模型等提取的过程模式)；相对通用过程模式(如从统一软件开发过程提取的过程模式)；组织过程模式(从软件组织的过程知识提取的过程模式)

Pauth 过程模式作者

Pvers 过程模式版本

Palia 过程模式别名

Pkey 过程模式关键字：可以代表模式主要内容和意图的词

Pinte 过程模式意图：描述模式的基本原理、目的、意图

Pprob 针对问题：过程模式可以解决的问题描述

Pforc 设计要素：解的部分必须满足的要求、必须考虑的约束条件、解应具备的特性等

Pactn 活动名称：使用该过程模式提供的策略可以完成的活动，每个模式对应一种活动

Psolu 解：给出在满足设计要素 Pforc 的条件下，可以解决问题 Pprob、完成活动 Pactn 的方法指导和实用建议，具体给出一组子活动列表及子活动之间的关系。

Pinic 初始上下文：描述问题和问题的解重复出现的前提，说明过程模式的适用性，为过程模式施用之前的系统状态

Presc 结果上下文：过程模式应用后产生的结果，为过程模式施用后的系统状态

Pguid 应用指南：给出该过程模式的优缺点，也可记录使用结果评述，便于改进

Pexam 应用实例：成功和失败案例

Prela 相关过程模式：如前任和接任过程模式

过程模式类的操作定义如下：

OpGETRpa(Pname) 根据过程模式 Pname 中的解 Psolu 生成该模式的组成关系 Rpa

OpGETact(Pname,Rpa) 根据组成关系 Rpa 获取组成过程模式的活动的集合

2.2.1.4 活动与过程模式

活动按其复杂程度可分为原子活动(Atomic activity)和复合活动(Composite activity),原子活动不可再分解,可通过工具或人工完成,复合活动可再分解为原子活动或复合活动。过程模式只用于复合活动,下述的活动一般指复合活动。

活动可按过程模式的指导进行,为活动选择过程模式时一般应首先用操作 AGETpat 从过程模式库找出与该活动有实现关系的所有过程模式,然后从这些过程模式中找出设计要素满足当前状态或约束的过程模式作为候选模式,对候选模式进行比较找出适合本软件组织或项目的过程模式,该过程模式将描述为完成该活动应该执行的一系列子活动及子活动的执行顺序,通过操作 PGETact 可获得这些子活动,如果子活动是复合活动,可再按上述过程找出该子活动的过程模式,如此反复,形成活动与活动、模式与模式、活动与模式的层次嵌套关系,实现过程知识的逐步细化,通过对细化程度的控制,可生成不同粒度的过程模型。所以说 PB-SPF 支持大粒度的过程模型生成,支持软件过程中的管理问题,如资源分配、进度计划、过程监控等;同时大粒度模型在需要时也能够得到细化,生成粒度小的模型,支持软件过程中的技术性问题,如活动的详细安排等。图5为活动－模式映射实例。

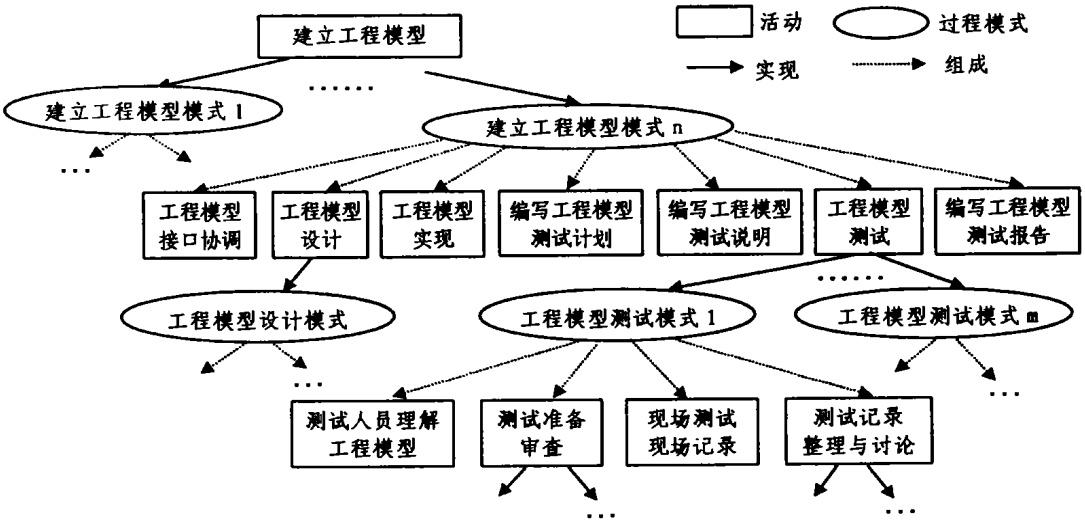


图5 活动－模式映射实例

限于篇幅,省略对工作产品的描述。

2.2.2 PB-SPF 实体元模型的行为视图 由角色行为视图、活动行为视图、产品行为视图三个视图组成,分别定义了角色之间、活动之间、产品之间可能存在的基本关系和性质。

定义4(行为视图) PB-SPF 实体元模型的行为视图为三元组 Bview ::= (BRview, BAview, BWPview),其中 BRview 为角色行为视图,BAview 为活动行为视图,BWPview 为产品行为视图。

本文只介绍活动行为视图。

定义5(活动行为视图) BAview::=(Cact,Cpat;RAaggre,RAorder)为 PB-SPF 实体元模型行为视图中的活动行为视图,简称活动行为视图。

其中:

(1)Cact 表示过程活动类(定义见2.2.1.2);(2)Cpat 表示过程模式类(定义见2.2.1.3);(3)RAaggre 表示活动类之间的聚合关系;(4)RAorder 表示活动类之间典型的执行顺序关系集合。

2.2.2.1 活动类之间的聚合关系

复合活动可再分解为原子活动或复合活动,构成活动类之间的聚合关系。活动类之间的聚合关系和过程模式与活动之间的组成关系是一一对应的,即活动的分解是通过活动所对应的过程模式实现的,在建立过程模型的过程中,为活动选定了过程模式,也就确定了与该活动有聚合关系的活动的集合。对应图5有如图6所示的用 UML 表示的活动聚合关系图。

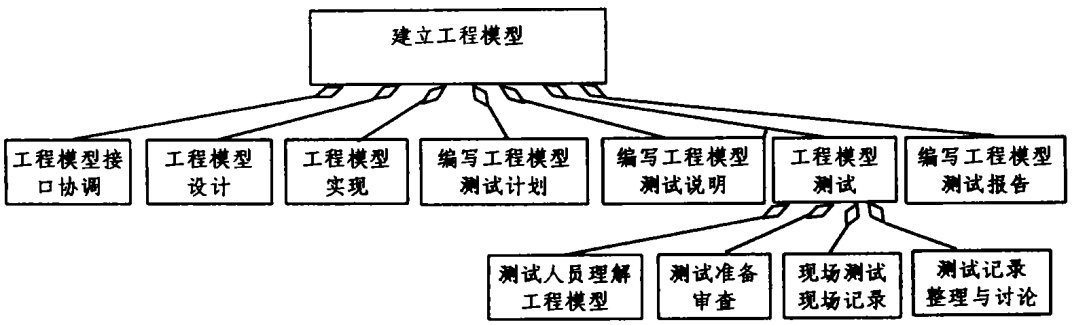
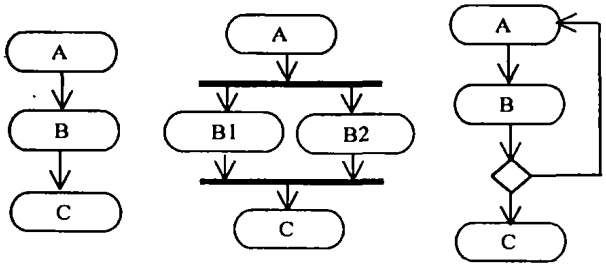


图6 活动聚合关系示意图

2.2.2.1 活动类之间典型的执行顺序关系

目前大多数的过程模型描述了三种典型的活动顺序关系:顺序执行关系、并行执行关系、循环执行关系,对应于三种典型的通用软件开发模型:瀑布模型、并行开发模型、螺旋模

型。目前的过程模型表示方法(如 Petric Net、UML)均可方便地表示这三种关系,图7是 UML 表示的三种典型的活动执行顺序关系。



①顺序关系　②并行关系　③循环关系

图7　三种典型的活动执行顺序关系

在实际应用中我们发现,有些活动的执行顺序关系用这三种典型关系及其组合难以表达清楚,如我们开发的战术软件是一种嵌入式的实时系统,与其所嵌入的外部环境(即指控系统)有多种接口,受外部限制较多,建立工程模型时(见图6)执行的活动如工程模型接口协调、工程模型设计、工程模型实现之间既不是纯粹的并行关系又不是纯粹的循环关系,其实际的执行关系是:接口协调开始后即可启动工程模型设计,工程模型设计启动后即可开始工程模式实现活动;只有接口协调完成后工程模型设计活动才能完成,只有工程模式设计活动结束后工程模型实现活动才可结束。我们将这种执行关系作为并行关系的一种扩展,称其为条件并行关系。

总结并抽象典型的活动执行顺序关系,赋予其严格的定义、良好的语义及可视化表示,可将其作为最基本的过程模式,我们称其为元过程模式,元过程模式的描述模板为:

MPname　元过程模式名称。按其所代表的典型的活动执行顺序关系取名,如顺序元过程模式。

MPauth　元过程模式作者

MPvers　元过程模式版本

MPprob　针对问题类型:元过程模式可以解决的问题类型描述

MPsolu　解:典型活动顺序关系的定义、语义及其可视化表示

由此,我们说 PB-SPF 实现了过程知识的二级重用,一级重用是重用元过程模式,复合生成过程模式,二级重用是重用过程模式,复合生成过程模型。PB-SPF 中所有的过程模式均是仅仅由元过程模式经过一定的操作复合生成的,保证了过程模式具有严格的语义及良好的结构,从而保证了重用过程模式复合生成的过程模型同样具有严格的语义和良好的结构。

活动之间的执行顺序关系繁琐复杂,活动分解的层次越低,过程模型的粒度越小,活动顺序关系越复杂,PB-SPF 中将活动执行顺序关系分散在过程模式中描述,使得模型粒度伸缩自如,既可统观全局,又可深入局部,适应不同层次需求。

总结与展望　本文介绍了基于过程模式的软件过程框架 PB-SPF,重点介绍了框架中的实体元模型,可以说 PB-SPF 是基于过程模式的软件过程建模和实施的基础,实体元模型是 PB-SPF 的基础;实体元模型是关于软件过程知识的知识,部件元模型是具体的软件过程知识,软件过程知识的使用即是软件过程模型的建立和实施过程。

目前我们的主要工作:

(1)进一步研究和完善 PB-SPF 的实体元模型,为 PB-SPF 建立坚实的理论和方法基础;

(2)在利用 PB-SPF 建立过程模型的实践中不断改进 PB-SPF;

(3)研究基于过程模式的软件过程度量和改进方法。

参 考 文 献

1 梁义芝,王延章,赵晓哲,缪旭东. 软件领域中的模式研究. 计算机科学,2003,30(3)

2 李健,邵维忠,杨芙清. 软件过程建模方法分类概述. 计算机应用与软件,1996,13(2)

3 周伯生,冯学民,樊东平(译). 统一软件开发过程. 机械工业出版社,2002

4 schel M Dr, Wiemers M. Das V-Modell 97. Oldenbourg. 1999

5 Gamma E,Helm R,Johnson R,Vllissides J. Design Patterns:Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995

6 Ambler S. Process Patterns:Building Large-Scale Systems Using Object Technology. Cambridge University Press,1998

7 Strrle H. Describing Process Patterns with UML. In:Proc. of the 8th European Workshop on Software Process Technology. Witten, Germany:Springer-Verlab Berlin Heidelberg, 2001. 173～181

8 Cugola G. Tolerating deviations in process support systems via flexible enactment of process models. IEEE Transactions on software egineering,1998,24(11)

9 Gnatz M,Marschall F,Popp G,Rausch A,Schwerin W. Towards a Living Software Development Process based on Process Patterns. In:Proc. of the 8th European Workshop on Software Process Technology. Witten, Germany: Springer-Verlab Berlin Heidelberg,2001. 182～202

10 Ribó J M, Franch X. Building Expressive and Flexible Process Models Using a UML-Based Approach. In:Proc. of the 8th European Workshop on Software Process Technology. Witten, Germany:Springer-Verlab Berlin Heidelberg, 2001. 152～172

11 Ribó J M, Franch X. Searching for Expressiveness,Modularity, Flexibility and Standarisation in software Process Modeling. In: proc. of the Brazilian Symposium on Software Engineering. Joao Pessoa,Brazil,2002. 59～276

12 Bergner K,Rausch A,Sihling M,Vilbig A. Putting the Parts Together:concepts,description techniques,and development process for componentware. In:Proc. of the 33rd Hawali intl. conf. on system sciences,2000

13 Morisio M, Tully C, Ezran M. Diversity in Reuse Processes. IEEE Software,July/Aug. 2002

14 Blanco M,Gutiérrez P,Satriani G. SPI Patterns:Learning from Experience. IEEE software,May/June,2001. 28～35

15 Palshikar G K. Applying formal specifications to real-wrold software deveopment. IEEE Software,Nov. /Dec. 2001

16 Ostolaza E,Quintano N,Satriani G. Patterns to Adopt Knowledge Based Solutions to Software Processes Management Problems. http://www.esi.es/Patterns/pdf

17 Lida H. Pattern-Oriented Approach to Software Process Evolution. http://www.reseachindex.org/