计算机科学 2005Vol. 32№. 11

一种软件过程建模语言SPML的设计

陈迎欣 刘 群
(哈尔滨工程大学计算机科学技术学院 哈尔滨 150001)

摘 要 本文提出了一个具有丰富语义、灵活的、可扩展的、形式化的软件过程建模语言SPML。它提供了两个不同抽象层次的描述语言以满足不同用户的需求。高抽象层次的语言SPML/H能够较全面地描述软件过程,具有良好的易用性;低抽象层次的语言SPML/L可以描述规则等,适合描述软件过程的细节。SPML/H还能够被逐步地变换成SPML/L,保证了软件过程模型可以在一个共同的形式化基础上被分析和运作。最后通过一个实例说明过程建模语言的有效性。

关键词 软件过程建模,软件过程,过程建模语言

The Design of a Software Process Modeling Language

CHEN Ying-Xin LIU Qun
(School of Computer Science and Technology, Harbin Engineering University, Harbin 150001)

Abstract It proposed a software process modeling language named SPML which support semantics richness, flexibility, scalability, reuse and formalization. This language provides two levels of abstraction to meet various user requirements. The high level abstraction named SPML/H can be readily to describe the software process form different aspects, and it is easy to be used. The low level language named SPML/L can be used to describe the details of the software process behaviors by specifying more comprehensive semantics such as rules. The SPML/H can be translated into SPML/L which make it possible to analyze and operate software process model on the same base of formalizing description. At last, an example is presented to demonstrate the efficiency of SPML.

Keywords Software process modeling , Software process, Software process modeling language

1 引言

软件和其他工业产品一样,其生产率和质量的高低最终取决于生产产品的过程好坏。通过对软件过程本身的研究,改进软件过程的质量,可以使软件项目的实施更有效,更可预见,从而获得高的生产率和高质量的软件。目前,软件过程的研究日益受到重视,其首要问题是如何表示过程,即建立过程模型;其次是在模型建立之后,如何实施过程,进一步的工作则是研制以过程为中心的CASE环境。贯穿这些问题的核心是过程建模语言,本文侧重对过程建模语言进行了讨论,提出了一个适合用于描述软件过程的语言SPML,它实际上是一个语言簇,包括不同抽象层次的描述语言,其中高抽象层次的过程模型语言SPML/H能够全面地描述软件过程,具有良好的易用性,一般的用户可以采用这种描述方式;低抽象层次的建模语言SPML/L可以描述更加丰富的语义,适合被过程工程师用于描述软件过程的细节。SPML/H可以被变换成SPML/L,保证了过程模型有一个共同形式化的基础。

2 软件过程建模语言的相关研究

过程建模语言PML(Process Modeling Language)作为软件过程技术的基础被许多组织进行了深入的研究。最初人们采用工业界中已有的非形式化过程建模方法描述软件过程,例如IDEFO[1]用自顶向下逐层分解的方式清晰地描述软件过程的功能结构,容易理解和交流,但是它不能描述并发、资源冲突、以及和状态有关的过程行为,也不支持对过程的模拟和运作。

1987年L. Osterwil在第九届国际软件工程会议(ICSE)上提出了"软件过程也是软件"的重要思想[2],引导人们将各种软件技术用于刻画软件过程。此后人们提出了大量基于不同范型的软件过程建模语言,它们主要可以归为以下几类:

1. 基于程序的PML。在某种传统的编程语言的基础上,增加了支持软件过程的特殊元素,可以详细地刻画过程的功能,但是用它建模需要大量的时间。如APPL/A[3]。

2. 基于规则的PML。采用产生式规则描述软件过程,过程中的活动被描述成一条包括前置条件、行为、后置条件的规则。如MSL[4]。

3. 基于图/网的PML。使用网或状态图的形式描述过程模型,能够比较容易地描述过程的功能和行为,但是他们描述过程的能力受所采用的形式化机制限制。如SLANG[5]。

4. 面向对象的PML。采用了面向对象的分析和设计方法,适合于描述抽象的过程模型,但是不适合描述支持过程运作的细节信息。如E3的建模语言[6]。

5. 基于主体的PML。能形式化地描述和支持合作,允许在过程运作时支持交流、协同、谈判等行为,从而可以帮助过程模型进行演化。如PEACE/PML[7]。

6. 基于时序逻辑的PML。支持过程建模者采用自顶向下的设计方法,从一个对过程的抽象描述开始,逐步地细化到对模型的详细设计,并可以检验细化步骤的正确性。如

陈迎欣 博士;刘 群 教授,博导。

BM[8]。

7. 多范型的 PML。采用多个范型描述软件过程的不同侧面，例如采用编程的方式描述过程的细节，而采用规则描述一般内容。如 SPELL[9]。

在文[10]中对这些 PML 进行了比较，它们有各自的特点，但是没有一个被认为能满足软件过程建模语言的需求。一般而言，基于一个范型的 PML 能够较好地描述软件过程中部分侧面的信息，但是缺乏较全面的描述软件过程的能力，例如不能同时描述软件过程的功能、行为、组织、信息这四个主要视图。多范型的 PML 可以通过综合各范型的优势来弥补这个缺点，但是需要解决各范型之间的信息共享、交换一致性等问题。

3 软件过程建模语言 SPML

软件过程是一个包含不同抽象层次信息的、具有一些特殊性质的复杂实体，在过程生命周期的不同阶段，过程模型对它的建模语言有不同的需求。例如，在过程定义阶段，用户希望 PML 具有直观性和良好的可理解性，并提供合适的抽象；而在分析和运作阶段，PML 需要是形式化的。通过分析软件过程的特点以及现有的 PML 的优缺点，我们认为一个良好的 PML 不仅应该可执行、可分析、支持模型演化，而且应该具有以下重要性质：

丰富、明确的语义 PML 通过结合各种技术来描述软件过程的各个方面，使用户可以全面地掌握软件过程的信息，从而能够更好地度量、控制和改进过程。另外，形式化的 PML 有利于参与过程的人员之间进行交流，避免了对过程理解上的分歧；

易用性 软件过程模型可以从不同的侧面进行描述，不同用户所偏好的描述方式可能是不同的，如某些用户喜欢使用基于活动分解的方式描述软件过程，另一些用户可能更倾向于以人员为中心的方式。因此，PML 要具有向不同用户提供适合的描述方式的能力，应该支持从不同侧面、不同抽象层次的描述方式，以适应不同用户的需求；

灵活性 软件过程模型必须能够适应运作时环境的不确定性，允许当出现模型中未描述的情况时，通过异常处理或人的决策行为选择其他的过程运作路径来完成过程的目标，避免因此造成过程模型的失效。另外，由于软件过程中常常出现人员或活动之间合作完成某些共同的任务，PML 应该支持对共享信息的并发操作，应该提供多视图，并支持用户在过程运作中定制和调整视图；

可扩展性 软件过程领域处于不断发展中，不可能定义一个包罗万象的 PML，而只能进行某些扩展使它适应新的软件开发方法、工具和软件工程要求。另外，由于各组织具有不同的文化背景、处于不同的领域、拥有不同的人员，他们所倾向的建模方法很可能是不相同的，一个良好的 PML 必须能够被扩展和调整，提供多样的描述方式，以适应不同的环境；

互操作性 互操作性体现在两个方面：一是不同组织进行合作时，其过程模型之间交流和共享过程信息的能力；二是过程模型和其他工具的互操作能力，特别是商业化的项目管理工具、配置管理工具、以及组织中已有的软件工程环境；

支持复用和定制 过程模型(或其一部分)应该能够被其他过程或组织复用，以帮助过程工程师充分利用已有的、成熟的过程实践经验，提高过程建模的效率和质量，PML 应该支持从已有的过程模型库中抽取过程的框架、构件、设计模式等可以复用的元素，PML 还应该支持对过程模型的定制。

3.1 参考定义元模型

由于软件过陈技术尚不成熟，现在没有一个标准的软件过程建模语言，这样，当多个组织合作开发项目时，会增加它们交流过程信息的成本。因此，软件过程模型中需要描述一些基本元素，这是不同组织之间合作的一个共同基础，其中，六种主要的过程元素为：活动、产品、角色、人员、工具和演化支持。另外，软件过程模型不仅是这些过程元素的集合，更重要的是它们之间的复杂关系的描述，因此，一个合适的软件过程建模语言需要能够描述这些过程元素和它们之间的关系。本文提出了一个基本的软件过程参考定义元模型，如图 1 所示。

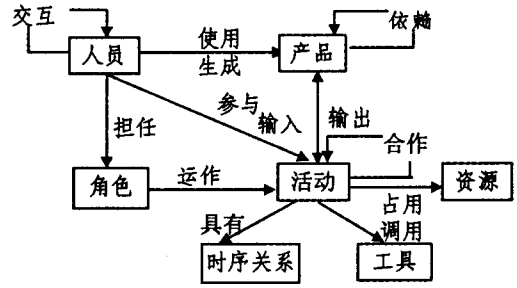

图 1 参考定义元模型

元模型中包括了活动、产品、角色、人员、工具等基本的过程元素，以及通常需要关注的过程元素之间的各种关系，例如过程的细化、活动间的时序关系、产品和活动的输入/输出关系、人员和活动的参与关系、产品之间的依赖、人员之间的交互、以及活动之间的合作等。

3.2 高层的建模语言 SPML/H

高层——用户界面层过程建模语言 SPML/H 为用户提供了过程的可视化描述手段，它是一个可视化的、抽象层次较高的、适合多数用户使用的软件过程建模语言。它提供了丰富的元素以及元素之间的关系，支持参考定义元模型，它支持自顶向下、逐步细化的建模方法。

SPML/H 从整个项目的角的描述软件过程。项目是软件过程建模的基础，受到时间、成本、人员等条件的限制。根据规模，项目可能被划分成多个子项目。项目包括一个工作计划、项目的整体目标、输入和输出产品、可利用的资源、项目的预定时间、工作记录、质量要求、配置、交流方式、子项目的合作方式等信息。参与一个软件项目的人员可以构成多个小组，每个项目小组必须有确定的目标、制定小组成员的角色以明确责任、确定计划、以及确定成员之间以及小组之间的交流方式。SPML/H 较全面地描述项目的各个方面。SPML/H 支持基本的过程元素，如活动、产品、角色、人员、工具。三种基本的关系：泛化(generalization)、聚合(aggregation)、关联(association)被用于建立更抽象的过程元素。SPML/H 允许活动有不同的实例化标记，如一个活动可以被标记为可选的，仅一次的、顺序的或周期的，实例化标记决定了过程模型中的活动实例化的方式。

控制流定义了活动执行的顺序，SPML/H 采用两种方式来灵活地描述控制流：(1)基于规则的方式：它较为灵活，执行活动的顺序可能因为运作状态触发了某些规则而发生变化，能够通过异常处理机制处理某些运作中出现的不确定因素。(2)基于约束的方式：这是一种直观、灵活的描述方式，即某个活动的运作均不能导致系统的运作状态违反任何约束。除此

之外，活动的执行顺序是任意的。过程模型中对一个活动的约束包括对它的时序约束和数据约束。SPML/H 增强了描述活动之间的时序约束关系的能力，提供了一些符合人们的思维方式的时序约束，并赋予了它们可执行的语义，例如：结束-开始(finish-start)：即只有当前一个活动结束后，后一个活动才可以开始；非并发(non-concurrent)：即多个活动可以以任何的顺序一次执行，其中任何活动不能同时执行，等。

过程中的数据流描述了产品在活动之间输入、输出的方式。早期的 PML 只支持同步数据流，即只有当活动结束时，才能将它的输出产品提供给数据流中的后继活动。为了提高活动的并发程度，软件过程应该允许访问尚未完成的活动的中间产品，这就是异步数据流。另外，过程模型中有必要规定活动对于其输入/输出产品的读写权限，因此，SPML/H 提供了四种基本的数据流：只读异步输入(或异步输出)、可读写异步输入、只读同步输入(或同步输出)、可读写同步输入。控制流和数据流的描述方式是面向活动的，当然也可以从产品的角度观察软件过程，SPML/H 通过刻画产品的状态图(如提交状态、评审状态等)和描述产品之间的依赖关系(如设计文档必须符合需求、程序源代码必须按照设计文档编写等)两种方式支持产品建模。

SPML/H 为元过程中的各种角色定义了常用的视图，在一个软件过程中，通过不同种类的视图，项目成员可以获得个人的工作日程清单，项目管理员可以观察和描述项目计划，还可以从不同角度监控过程的运作，如从产品角度观察生产和消费它的活动，从活动角度观察相关活动的执行时序和合作关系等。SPML/H 允许过程建模者定制合适的视图，包含过程模型中部分过程元素及它们之间的部分关系，便于用户采用希望的方式理解和描述过程模型。

3.3 低层建模语言 SPML/L

低层——过程实施层建模语言 SPML/L 为计算机提供了过程的形式化定义手段，可以描述由多个可并发运行的对象构成的系统，并可以直接描述对系统的全局行为的约束，SPML/L 中面向对象的描述方式提供了丰富的描述能力，并且有利于从已有的过程模型中抽取并复用构件。同时，它提出了用模式来描述多个对象的操作在执行时需要满足的约束，用户可以直观地了解过程的整体行为，并且过程的行为在一定程度上(不违反模式的约束)是可变的，从而可较好地支持过程的动态性。

SPML/L 用对象描述所有软件过程涉及的实体，如活动、产品、角色、人员等。对象封装了一些数据、对这些数据的操作和操作的规则，对象之间通过发送和接收事件进行交互。例如：产品对象的主要属性有：产品名、产品状态等，主要操作有：读和写操作；活动对象的属性有：活动名、活动状态、负责人、优先级、参与角色、工作量等，主要操作有：启动、挂起、继续、结束、中止、提交产品、获取产品等。建模语言提供了一些基本的数据类型(如字符串、数值型、集合型等)，在此基础上用户可以通过三种内置构造方式(聚合、泛化、关联)自定义过程元素的类型，过程模型中的元素则是这些类型的一个实例化的对象。

(1)聚合："a consist $\{a_1, a_2, \cdots, a_n\}$"表示类 a 包含了类 $a_1, a_2, \cdots, a_n$。

(2)泛化："a is $b\{\cdots\}$"表示类 a 继承了类 b，并定义了一些新的属性、操作和规则。

(3)关联："a $\{\cdots$ attr_b:$b\cdots\}$"表示类 a 定义的某个属性 attr_b 的类型是 b。

为了便于分析，规定数据类型的范围是有限的，以保证过程模型在运作时可能的状态是有限的。过程建模语言中的对象采用事件-条件-行为来描述规则，其基本形式为：On event IF condition DO action，表示当某个事件 event 发生时，如果满足指定的条件 condition，则执行操作 action。事件是过程运作中动态产生的，包括事件名以及一些参数。事件可以是其它对象的操作请求，过程运作状态的变化，或者时钟事件，也可以是用户自定义的。系统中发生的事件被广播到所有的对象，它们需要察看自己的规则集合并判断是否需要处理这个事件。规则中的条件用谓词描述，它只能引用对象中的数据以及这个事件的参数，通过计算谓词的值可以决定是否执行某个操作。对象中的操作是一个例程，可以计算并修改本对象中的数据，也可以发送事件和其他对象进行交互和协作。

软件过程具有动态性和不确定性，一般的 PML 提供的例外处理语句不能灵活地反映这个特点。本文提出用模式描述软件过程中对全局行为的约束，能够很好地体现软件过程的动态性。模式约束就是在过程运作中，一个对象执行某个操作时，必须不和过程模型中任何模式产生冲突。过程实施层的建模语言提供了丰富的构造复杂模式的符号：顺序(；)、并发(‖)、或(∨)、非(¬)、可选([])、迭代(+)、异或(⊕)、循环(*)。根据基于规则和基于约束的不同，可以将活动之间的时序关系变换成模式约束的表示形式，如表 1 所示。

表 1 部分时序关系和模式约束的对应

	活动的时序关系	对应含义	对应的模式约束
活动 A B	Finish-start	A 完成后，开始执行 B	(A. commit ; [B. start])*
	Start-start	A 开始执行后，开始执行 B	(A. start ; [B. start])*
	Start-finish	A 开始执行后，B 执行完成	(A. start ; [B. commit])*
	Finish-finish	A 完成后，B 执行完成	(A. commit ; [B. commit])*
	After-expect	A 完成后，希望 B 开始执行	(A. commit ; B. start)*
	After-prohibit	A 完成后，禁止执行 B	(A. commit ; ¬B. start)*
	While-prohibit	在 A 期间，禁止执行 B	(A. start ; ¬B. start ; (A. commit⊕A. abort)B. commit))*

4 软件过程语言 SPML 的支持环境

SPML 的支持环境如图 2 所示，它支持 SPML，支持过程生命周期的主要部分，包括建模、分析、运作、度量和演化，能够有效地帮助用户进行项目管理。

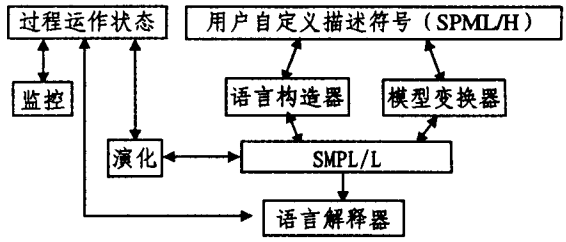

图 2 SPML 的支持环境

(1)语言解释器：语言解释器的主要功能是按照过程模型的信息，根据用户和 SPML 支持环境交互时的请求进行软件过程的运作，包括改变过程运作的状态、向用户输出信息等。

语言解释器的构成以及各构件之间的关系见图 3。

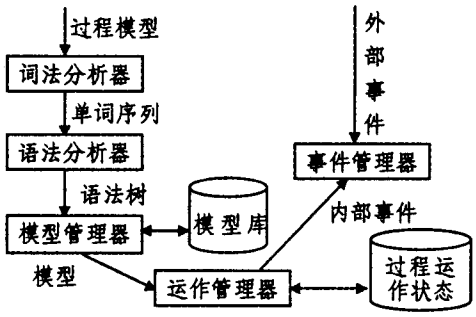

图 3　语言解释器的结构

词法分析器：读取 SPML/L 描述的过程模型，将它转换成单词序列；语法分析器：按照 SPML/L 的语法规则将单词序列组成语法树，检查语法的正确性等；模型管理器：将语法树存储在过程管理库中，它是过程模型的语法表示；事件管理器：对于外部事件以及过程执行过程中各对象产生的事件，事件管理器按照事件发生的先后顺序要求运作管理器处理这些事件；运作管理器：读取过程模型库中的模型信息，根据环境中出现的事件、过程模型以及过程模型的状态，执行过程模型中隐含的相应操作，并改变过程运作的状态。

词法分析器、语法分析器和模型管理器实现比较简单，这里不多阐述，下面介绍事件管理器和运作管理器的实现。在事件管理器和运作管理器中有 getobjectvalue()和 getpatternstate()两个函数用来获取过程运作状态。SPML/L 描述的模型包括多个对象和模式约束，因此过程的运作的状态由各对象的属性集合以及模式约束的当前状态集合组成，对象属性的值可以用一个三元组来存储：<objectname，attrname，value>，分别表示对象名、属性名以及属性值。模式约束的当前状态可以存放在一个整型数组中。Onevent()处理环境中出现的某个事件，它由事件管理器调用，执行过程如下：

1)找出过程模型中处理这个事件的规则，得到规则中操作和条件信息。

2)调用 callcondition()方法，确定运作状态是否符合执行此规则所需的条件，如果满足则转 3)，否则退出。

3)执行 preexec()方法，检查执行规则中的操作是否会与模式约束相冲突，即是否有模式约束的状态进入"陷阱"状态，如果没有冲突则转 4)，否则退出。

4)执行 executeaction()方法，执行过程模型中相应操作的语义，这个方法会修改某些对象属性的值。

5)调用 postaction()方法，修改各个约束模式当前状态的集合。

(2)模型变换器的功能是将 SPML/H 变换成 SPML/L 的形式，再输入上述的 SPML/L 语言解释器。这样做有两点好处：其一，可以在更小的粒度上分析、运作、演化过程模型；其二，不仅可以支持预定义的 SPML/H，也可以支持用户自定义的描述符号。将 SPML/H 变换成 SPML/L 的步骤如下：

1)调用 getargn()、getexpression()、getprocedure()和 getlocation()方法得到变换此描述符号的规则信息。

2)调用 callexpression()方法计算是否采用此变换规则，如果采用则转 3)，否则转 1)。

3)调用 replaceprocedure()将规则中的 procedure 参数替换成实际的数据。

4)根据 location 信息将变换后的 procedure 放入整个过程模型对应的 SPML/L 程序的合适位置。

(3)语言构造器，其功能主要是向规则库中添加相应的规则，以构造自定义描述符号。

(4)演化可以增加、修改、细化和删除过程元素和它们的关系。主要的演化规则和相应的处理操作包括两方面：其一，修改过程、过程元素和其关系属性，如修改项目、过程和活动属性：修改所有执行中的活动实例的属性，通知相关的项目成员；其二，增加、细化和删除过程元素和它们的关系，如增加一个活动：通知参与运作这个活动的人员；删除一个人员：会导致和所有其它关系的元素被删除，并通知项目负责人和其本人。

(5)过程运作状态：可以从活动、产品、人员等多个角度观察一个正在运作或已经完成的过程，从中了解过程运作的各种信息。如项目进度，包括对可能延期的、可按期进行的、按期完成的、延期完成的各种活动的数量统计，可以按优先级、工作小组、人员、时间段等方式分类统计；各种活动的状态；各种产品的状态、各人员的工作量分布等。

(6)监控：向项目负责人和其他人员提供了从多角度观察过程运作状态的能力，使它们能够跟踪并发现潜在的风险，并通过控制和调整过程运作消除可能出现的问题。提供了控制某个子过程或项目的方法，如开始、结束、中止、挂起某个子过程或整个项目的运作，强制相关人员做某个工作，或不允许他/她做某个工作。

5　一个例子

下面举一个简单的例子，说明 SPML 在软件过程建模中的应用，此例子从产品的角度刻画软件过程模型。在详细设计阶段，此阶段有三类产品，它们的结构如图 4。

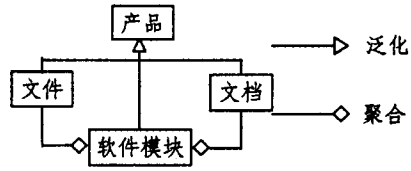

图 4　产品及其关系图

一些文档和文件组成了软件模块，这些产品类型可以有不同的状态集合和状态图，描述了产品在某些操作后的状态变化。如图 5 描述了文档产品的状态图，表示一个文档可以被多次提交，提交后需要通过评审。

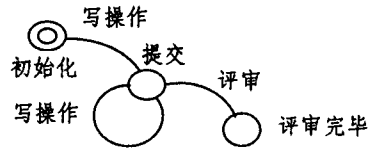

图 5　文档产品状态图

对应于上面的产品及其关系图和文档产品状态图的 SPML/L 描述如下：

```
file is product;
document is product{
        state(initial, submitted, reviewed);
on read_event do read;
on write_event if state = initial or state = submitted
do write;
on review_event if state = submitted do review;
review (){state = reviewed;}
        }
module is product consist{file, document}
```

结束语 本文提出的软件过程建模语言 SPML 从两个层次描述软件过程，具有丰富的语义、易用性、灵活性、可扩展性、互操作性和支持复用等特点。高层建模语言 SPML/H 提供了高度抽象的描述方法以便于一般用户使用，支持从多个视图进行建模，能够全面地描述软件过程的主要方面。低层建模语言 SPML/L 提供了更加丰富的语义描述能力，采用对象和模式来描述对软件过程中全局行为的约束，这种方式能很好地适应软件过程的动态性和不确定性，充分考虑了软件过程中人的因素。SPML/H 还能够被逐步地变换成 SPML/L，保证了软件过程模型可以在一个共同的形式化基础上被分析和运作。实践证明 SPML 能够很好地支持不同层次和需求的过程建模。

参 考 文 献

1 DEFO Report. FIPS Publication 183. Computer Systems Laboratory of NIST, 1993

2 Osterwil L. Software process are software too. In: proc. of the 9th intl. conf. on software engineer, 1987

3 Sutton S M, Heimbigner Jr., D, Osterweil L J. APPL/A: A language for software-process programming. ACM Trans. on software engineering and methodology, 1995, 4(3): 186～221

4 Kaiser G E. MARVEL 3.1: A multi-user software development environment. In: Proc. of the Intl. Symposium on Logic programming, Vancouver, Canada, 1993

5 Bandinelli S, Fuggetta A, Ghezzi C, et al. SPADE: A Environment software process modeling and Technology. Research Studies Press Ltd, 1994. 223～227

6 Jaccheri M L, Picco G P, Lago P. Eliciting Software Process Models with the E3 Language. ACM Transa. on software Engineering and methodology, 1998, 7(4): 368～410

7 Arbaoui S, Oquendo F. PEACE: Goal-Oriented logic-Based-Formalism for Process modeling. In: Software process modeling and Technology. Research studies press Ltd, 1994

8 Bruynooghe R H, Greenwood R M, Robertson I, et al. RADM: Towards a Total Process Modeling System. In: Software process modeling and Technology. Research studies press Ltd, 1994

9 Conradi R, Haguseth M, Liu C. Planing Support for cooperating transactions in epos. In: Proc. CAISE'94, 1994. 2～13

10 Dernjame I C, Kaba B A, Wastell D. Software Process: principles, methodology and technology. LNCS 1500, springer verlag, 1998

中国计算机学会电子政务与办公自动化专委会

第三届全国 Web 信息系统及其应用学术会议（WISA2006）

征文通知

Web 在人类信息的存储和交换过程中发挥日益重要的作用．适用于网络平台、动态特性的 Web 技术层出不穷，提高办公效率、节约资源消耗和扩大信息共享的 Web 应用和服务蓬勃发展．但随着 Web 规模的不断膨胀，Web 上数据资源以爆炸性的趋势飞速增长，而且信息的结构和内容越来越复杂，使得管理、查询和使用 Web 信息变得愈加困难．

全国 Web 信息系统及其应用会议（WISA）是中国计算机学会电子政务与办公自动化专委会主办的系列会议．首届会议 WISA2004 于 2004 年 10 月在武汉圆满召开，会议共收到应征论文 368 篇，其中 54 篇论文在《武汉大学学报（英文）》（EI 源刊）作为正刊专辑发表（已经全部被 EI 收录）．第二届会议 WISA2005 于 2005 年 9 月在沈阳召开，会议共收到应征论文 606 篇，录用 239 篇，其中 62 篇论文在《武汉大学学报（英文）》（EI 源刊）作为正刊专辑发表，118 篇在《计算机科学》发表，59 篇在由清华大学出版社出版的会议论文集《Web 信息系统与技术》发表．

WISA2006 将于 2006 年 10 月在南京召开．会议将继续这一良好的传统，在 Web 技术、信息系统、电子政务与办公自动化等方面进行深入广泛的学术交流．会议论文集仍将分两部分出版，录用论文中将选择 60 篇左右高水平论文以英文方式继续由《武汉大学学报（英文）》（EI 源刊）正刊专辑出版，中文论文集将由著名计算机核心期刊《计算机科学》专刊和中央级出版社出版．会议期间除进行会议论文交流外，还将邀请著名学者作特邀报告．本次会议仍将评选大会优秀学生论文．

一、征文范围（包括但不限于）

Web 信息挖掘与检索；语义 Web 与智能 Web；Web 站点逆向工程与维护技术；Web 测试与 Web 应用的质量保证；Web 与网格计算；多媒体数据管理；Web 与数据库技术；工作流模型；XML 与半结构化数据管理；组件与中间件技术；Web 信息系统环境与基础；代理技术及信息管理；Web 应用框架和体系结构；自动文本索引与分类技术；Web 与信息系统安全性；决策支持与分析技术；Web 信息系统开发工具；电子政务与电子商务框架及应用；Web 系统度量与分析技术；电子政务与办公自动化发展现状与趋势

二、来稿要求

1. 本次会议只接受 Email 投稿．

2. 中英文稿均可，一般不超过 6000 字，为了便于出版论文集，来稿必须附中英文摘要、关键词、资助基金与主要参考文献，注明作者及主要联系人姓名、工作单位、详细通信地址（包括 Email 地址）与作者简介．稿件要求采用 WORD 或 PDF 格式．

三、联系信息

投稿地址：东北大学信息科学与工程学院 王国仁（wanggr@mail.neu.edu.cn）

会务情况：东南大学计算机科学与工程系 徐宝文 许蕾（xlei@seu.edu.cn）

大会网站：http://www.neu.edu.cn/wisa2006/

四、重要日期 征文截止：2006 年 3 月 25 日　录用通知发出：2006 年 4 月 15 日　正式论文提交：2006 年 4 月 30 日