

基于语义的工作流过程优化

毋涛,黄宁

WU Tao,HUANG Ning

北京航空航天大学 计算机学院,北京 100083

School of Computer Science and Engineering,Beihang University,Beijing 100083,China

E-mail:wutaoandy@163.com

WU Tao,HUANG Ning.Optimization of workflow process based on semantic.Computer Engineering and Applications, 2008,44(9):43-46.

Abstract: An intelligent workflow management system should have the ability to dynamic analyze and optimize workflow process to improve performance and efficiency.This paper presents a semantic-based workflow optimization method,which uses ontology to describe the concept of the workflow,endsows the workflow model with corresponding dynamic semantic,so as to construct a workflow fact library; by defining a set of classification rules of activity,to classify all of the workflow activities semantically and construct a library of classification rules; by performing these classification rules,to acquire optimized information,and performing corresponding operation according to different activities,so as to optimize the structure of the workflow model.An example is given to illustrate the feasibility and correct of the method.

Key words: workflow process optimization;semantic;ontology;classical rule

摘要:为改善工作流的性能和效率,一个智能化的工作流系统应该具有动态的分析和优化工作流过程的能力。提出了一种基于语义的工作流优化方法,它使用本体概念来描述工作流,利用本体技术的知识表达能力,给工作流模型赋以相应的动态语义,从而构建工作流事实库;通过制定基于本体的活动分类规则,对工作流的所有活动进行语义分类,从而构建分类规则库;通过执行活动分类规则来获取优化信息,针对不同类别的活动执行相应的操作,从而实现对工作流模型的结构优化。最后,通过实例来说明该方法的可行性和正确性。

关键词:工作流过程优化;语义;本体;分类规则

文章编号:1002-8331(2008)09-0043-04 **文献标识码:**A **中图分类号:**TP311

1 引言

工作流过程优化是运用静态的或动态的模型分析方法和手段来分析已经建立的工作流模型的性能,发现模型中可能存在的问题,在此基础上改进和优化工作流模型。优化工作流是在工作流模型的基础上进行的,各种优化算法大多是针对工作流的模型结构和性能参数来进行的,如 Heterd^[1]等人提出了一种基于遗传算法的分配方法,根据用户的特殊需求来获取最佳的分配策略;Van Heel^[2]等人提出了一种边际分配法,并证明了对于某些特殊结构的工作流模型来说,这种分配方法是最优的。在以往的研究中,各种优化算法都是在对工作流信息进行简单判断的基础上来实现的,而没有从语义上来实现对工作流的优化。随着工作流技术的发展,特别是各种基于 XML 的工作流定义语言如 BPEL4WS(Business Process Execution Language for Web Language),XPDL(XML Process Description Language),ebXML 等的出现,从个体间关系的表达上增强了对工作流信息的描述能力,这就给从语义上来对工作流过程进行优化提供了可能性。

目前,本体技术因其适合用于表达领域知识、实现知识重用和知识共享而得到了广泛的研究和应用。将本体技术运用到工作流中已经开始得到广泛的研究,如 Vieira^[3]提出了一种由本体驱动,用于描述工作流、资源、用户间语义关系的工作流系统框架,它主要研究工作流执行的灵活性;Wang^[4]描述了一种用于工作流语义创建、推理及流程重组的基于代理的工作流本体模型,它主要研究代理对动态工作流自动执行的支持。但是,这些研究主要集中在工作流的执行上,并没有考虑对工作流模型的结构进行分析和优化。因此,本文提出了一种基于语义的工作流优化方法,它使用本体概念来描述工作流,通过执行符合推理机制语法的基于本体的过程优化推理规则,对工作流模型进行结构优化。

2 本体和规则描述语言

2.1 OWL-S 本体

目前,一些学者已经开始使用本体来描述工作流,owl-s^[5]就是其中的一种。本文即采用 owl-s 来描述工作流。

作者简介:毋涛(1982-),男,硕士研究生,主要研究领域为电子商务、工作流等;黄宁(1968-),女,博士,副教授,主要研究领域为电子商务、软件工程、语义网和知识工程等。

收稿日期:2007-07-12 **修回日期:**2007-10-23

Owl-s 是基于 Owl 语言的 Web 服务本体, 作为描述 Web 服务的知识本体, Owl-s 的目标是能够对一个 Web 服务进行逻辑描述。其顶层服务本体是由以下元素组成的, 如图 1 所示。

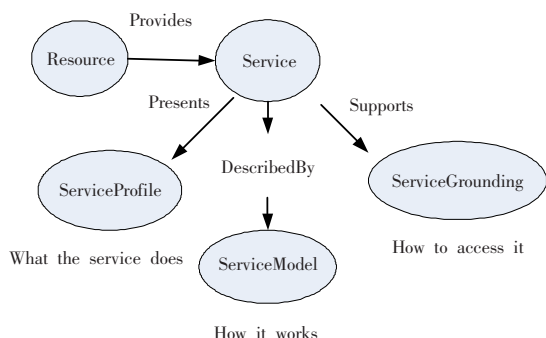


图 1 owl-s 服务本体

Service 类是对一个声明了的 Web 服务的结构化的引用点, 每个 Service 实例将对应一个发布的服务;

ServiceProfile 描述服务做什么, 主要描述服务和提供者信息;

ServiceModel 描述服务如何工作和服务具体的逻辑执行顺序;

ServiceGrounding 描述了如何访问服务, 即描述访问服务的通信协议及其它细节。

在这些元素中, 我们更为关心的是 ServiceModel, 因为它给出了 Web 服务的具体行为细节。ServiceModel 本体主要有两个模块: Process 本体和 ProcessControl 本体, 其中 Process 本体以流程的形式描述了服务中各子过程的执行顺序, ProcessControl 本体主要管理和监控服务流程的执行。目前 Owl Services Coalition 还没有给出 ProcessControl 本体的规范。Process 本体的最基本的实体是过程(Process)。Process 又分为原子过程、简单过程和组合过程。其中原子过程是一个直接可以调用的过程, 它没有子过程, 从服务请求者的角度来看, 原子过程能通过一步完成。简单过程不是一个可直接调用的过程, 但与原子过程类似, 简单过程也能通过一步完成。简单过程用作抽象元素, 它既可以用作原子过程的视图, 又可以用作组合过程的简化表示。组合过程可以分解成其它的过程(包括组合过程和非组合过程), 是通过控制结构将原子过程或组合过程组合起来的一个复杂过程。目前 Owl-S 有 Sequence, Split, Condition, Split+Join, Unordered, Choice, Iterate, If-Then-Else, Repeat-Until 和 Repeat-While 等控制结构构造符。

结合 Owl-s 本体的特性, 从工作流的角度来研究 Owl-s, 可以得到以下的结论, 这也是以 Owl-s 来描述工作流的基础。

(1) 工作流和 Web 服务是同一个功能实体在不同应用领域的不同表现形式, 它们可以采用相同的管理方式。比如, 一个单一的 Web 服务可以看作是一个简单的工作流, 同样, 一个复杂的工作流可以看作是由一个或多个 Web 服务组成的。

(2) 一个服务通过其属性来描述其语义功能, 同时通过数据绑定和控制流信息提供基本的执行信息; 与之对应, 一个工作流通过属性, 控制流和数据流来描述。

(3) Web 服务可以看作是工作流在语义网上的应用, 这有利于工作流组合多种资源和在分布式环境下的执行。

通过以上的描述, 可以得到 Owl-s 中相关概念和工作流中相关元素的映射关系, 如表 1 所示。

表 1 owl-s 和工作流相关概念映射关系表

Owl-s	工作流
service:ServiceModel	工作流顶层类
process:Process	工作流模型
process:AtomicProcess	活动类
process:CompositeProcess	流程类
process:ControlConstruct	工作流控制节点

2.2 SWRL 规则描述语言和 jess 推理引擎

在众多的规则描述语言和推理引擎中, 本文选择 SWRL^[6] 和 Jess^[7]。

SWRL(Semantic Web rule language)是以语义的方式表示规则的一种语言, 目前 SWRL 是 W3C 规范的一员, 是 W3C 最近提出的一种语义 Web 规则标记语言, 其起点是 DAML 语言系列中 DAML2Rule 或 DAML2L 规则语言的一些初步设想。SWRL 以 hand 表示推论结果, body 表示推论前提的基本形式, 所以 SWRL 可以看作规则和本体论的结合, 这样, 在撰写规则时, 可以直接使用本体论中所描述的关系和类谓词。这也是本文选择 SWRL 作为规则描述语言的主要原因。

Jess(Java Expert Shell System)是基于 Java 语言的 CLISP 推理机。CLISP 是基于产生式的前向推理引擎, 它是一种非常基础性的东西, 许多上层的推理任务, 都要映射到这个推理引擎上来运行。目前, 由于一个推理机可以直接运行 SWRL 规则, 而 jess 推理机可以通过 xslt 将 swrl 转换为 jess 格式, 实现起来较为简单, 所以本文采用 jess 推理机来进行规则推理。

3 基于语义的工作流优化方法

本方法采用基于本体论和 SWRL 为基础的推理方法来实现对工作流的过程优化, 主要通过三步来完成对工作流的过程优化: (1) 构建基于 Owl-s 的工作流本体库; (2) 构建基于 SWRL 的规则库; (3) 推理及优化。

3.1 基于本体的工作流模型

本文将工作流模型当成一个对应的本体知识库框架来考虑, 其目的主要是建立企业工作流活动之间的关联, 透过建立该本体库可以清楚地了解活动与活动之间, 活动与资源之间(参与人员, 操作的数据对象等)和资源与资源之间的关联信息, 通过对 Owl-s 本体库的扩展构建工作流本体库。

3.1.1 工作流模型的本体层次

图 2 显示了本体模型中工作流类对象的层次, 其中 workflowElement 类及其子类是对 Owl-s 本体的扩展。下面主要考虑

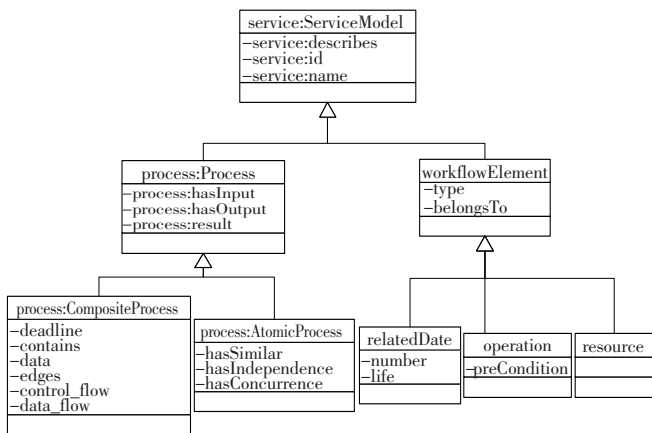


图 2 工作流模型本体层次图

以下三个概念:service:ServiceModel,process:Process 和 workflowElement。

service:ServiceModel 相当于工作流模型中的最基本的类,它包含了子类中的通用属性,如 name, ID 和 description 等。

process:Process 是 Owl-s 中的过程模型,相当于工作流中的工作流模型,它继承了 service:ServiceModel 类,用于工作流中的元素定义;workflowElement 是 service:ServiceModel 的另一个子类,它的子类包括 relatedDate, operation 和 resource 等类,它们用于定义工作流的参与元素。

另外,在 Owl-s 中 process:ControlConstruct 包含了工作流的所有路有类型的节点描述,如类 process:Sequence 相当于顺序结构,类 process:Split 相当于分支结构,类 process:Choice 相当于选择结构等,它可以描述工作流控制流的所有信息。

3.1.2 工作流本体中的主要类

process:AtomicProcess:该类用于工作流的活动定义,它是工作流流程中最主要的概念,表 2 列出了其对应的主要属性。

表 2 process:AtomicProcess 属性表

属性名	值域	描述
type	xsd:int	活动的类型
state	xsd:int	活动的状态
begin	Expression	开始条件
end	Expression	结束条件
hasParticipant	Participant	用户集合
belongsTo	CompositeProcess	活动所属工作流
requireData	relatedData	活动所处理的数据对象集合
hasPioneers	AtomicProcess	所有前驱活动的集合
hasSuccessor	AtomicProcess	所有后继活动的集合
Operation	operation	对数据对象进行的操作
hasSimilar	List	符合相似规则的活动集合
hasIndependence	List	符合独立规则的活动集合
hasConcurrence	List	符合并发规则的活动集合
hasPAndO	List	前驱(a)及对它所传进来的数据对象所做处理的方法(o)的元组所组成的集合
hasSAndC	List	后继(a)及激发这个后继活动的条件(c)所组成的元组集合

process:CompositeProcess:该类用于工作流的流程定义。一个 process:CompositeProcess 可以包含多个 process:AtomicProcess。除了继承 process:Process 父类的属性,它还添加了 deadline, contains, edges, data, data_flow 和 control_data 等属性,分别用来描述流程预计截止时间,包含的活动节点,组成工作流的边节点,参与的数据,数据流和控制流信息。

process:Participant:该类用于工作流的参与者的定义,它是 Owl-s 的基本类。

expr:Expression:该类用于描述一个逻辑表达式,它是 Owl-s 的基本类。

3.2 工作流优化规则

优化规则的制定主要包括工作流模型合理性判断规则和活动分类规则。前者主要用来约束和检测工作流模型的合理性,后者主要用来对工作流的活动进行分类,从而为下一步的优化提供数据基础。

3.2.1 模型的合理性

模型的合理性规则主要考虑构成工作流过程的各个活动的正确性问题,这部分可以根据 Owl-dl 来表示,即通过本体库中的类与类之间,属性与类之间的关系来表示。如通过对开始

活动类的 hasPioneers 属性添加约束条件 hasPioneers=0 来保证开始活动没有前驱,同理可以保证结束活动没有后继。另外,通过对中间活动类添加约束条件 hasPioneers>=1 和 hasSuccessor>=1 来保证工作流模型中没有冗余活动。

3.2.2 活动分类规则

在本体库中,主要运用到了三类基本的活动,即相似活动,并发活动和独立活动,通过对过程中的所有活动进行分类来实现优化操作,其定义和规则表示如下。

定义 1 (相似活动)如果有两个活动节点,它们处理相同的数据对象,同时有相同的用户集合,而且对相同的数据对象具有相同的权限,对相同的前驱活动和后续活动具有相同的操作,则称这两个活动为相似活动。

相似活动通过活动类的 hasSimilar 属性来表示,相对应的规则为:

- 1.middleActivity(?a) ∧ middleActivity(?b) ∧ process:hasParticipant(?a,?ra) ∧ requireData(?a,?oa) ∧ process:hasParticipant(?b,?rb) ∧ requireData(?b,?ob) ∧ sameAs(?ra,?rb) ∧ sameAs(?oa,?ob)
- 2.hasPAndO(?a,?apo) ∧ hasPAndO(?b,?bpo) ∧ sameAs(?apo,?bpo)
- 3.hasSAndC(?a,?asc) ∧ hasSAndC(?b,?bsc) ∧ sameAs(?asc,?bsc) ∧ first(?asc,?c) ∧ hasPAndO(?c,?ca) ∧ first(?ca,?aa) ∧ sameAs(?ca,?a) ∧ hasPAndO(?c,?cb) ∧ first(?cb,?ab) ∧ sameAs(?cb,?b) ∧ second(?ca,?ma) ∧ second(?cb,?mb) ∧ sameAs(?ma,?mb) → hasSimilar(?a,?b)

定义 2 (独立活动)如果一个节点的触发条件不依赖于它的父亲节点的执行结果,同时它所处理的数据对象并不会被它的父亲节点所改变,则称这个节点为独立节点。

独立活动通过活动类的 hasIndependence 属性来表示,相对应的规则为:

- $$\text{middleActivity}(?a) \wedge \text{middleActivity}(?b) \wedge \text{hasPioneers}(?a,?ap) \wedge \text{sameAs}(?ap,?b) \wedge \text{hasSuccessors}(?b,?bs) \wedge \text{differentFrom}(?bs,?a) \wedge \text{requireData}(?a,?ao) \wedge \text{requireData}(?b,?bo) \wedge \text{sameAs}(?ao,?bo) \rightarrow \text{independent}(?a,?b)$$

定义 3 (并发活动)同一活动的所有后继活动称为等价活动,所有等价活动的后继活动也为等价活动,在等价活动中,如果它们处理同一个数据对象,同时还具有相同的用户集合,则称其为并发活动。

并发活动通过活动类的 hasConcurrence 属性来表示,相对应的规则为:

- 1.middleActivity(?a) ∧ hasPioneers(?a,?ap) ∧ hasSuccessors(?ap,?b) ∧ differentFrom(?a,?b) ∧ process:hasParticipant(?ap,?b) ∧ differentFrom(?a,?b) ∧ process:hasParticipant(?b,?rb) ∧ requireData(?b,?ob) ∧ sameAs(?ra,?rb) ∧ sameAs(?oa,?ob) → hasConcurrence(?a,?b)
- 2.hasConcurrence(?c,?d) ∧ hasSuccessors(?c,?a) ∧ hasSuccessors(?d,?b) ∧ differentFrom(?a,?b) → hasConcurrence(?a,?b)

3.3 规则推理和过程优化

本文采用 jess 推理引擎作为规则推理的平台,工作流本体

库所建立的知识成为规则推理的基础,这些本体知识成为 jess 中的事实,而由 SWRL 所建立起的规则是推理的判断法则。通过 XSLT 将本体知识库和规则文件转换为支持 jess 推理的事实和法则,从而实现推理。通过规则的推理,就将工作流过程的活动进行了分类,同时将结果保存到了本体库中。最后,利用这些分类信息,通过对工作流的相关活动执行合并和分离操作来实现对工作流过程的优化,如针对相似活动和并发活动的合并操作,独立活动的并发操作等。

4 实例分析

下面以某飞机制造厂在制定一个管理计划时的发文流程为例。企业在制定该管理计划时,由公司办公室制定计划草案,在经由相应的审核之后编写具体规程;各部门根据规程,共同制定出一份具体的计划书,并通过公司办公室的整理,将最终的计划书提交审批,最终发布,具体流程如图 3 所示,图中实箭头表示流程的控制流程,虚箭头表示的是数据流程。

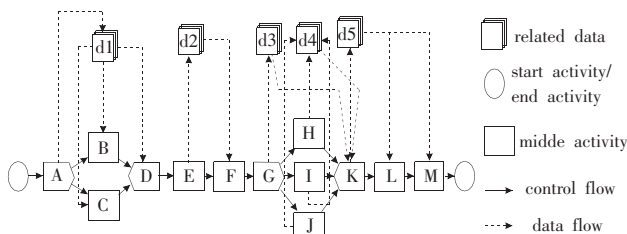


图 3 优化前发文流程

利用工作流本体库构建该流程,建立本体知识库,并以 Owl 文件的形式保存。将知识文件和分类规则文件导入到推理引擎中,推理得到该流程中各个活动节点的分类信息,并保存到相应的属性中,最后,利用分类信息对工作流过程的结构进行优化。在本例中,通过推理得到的分类结果是活动 B 和 C 为并发活动,活动 H、I 和 J 为并发活动,分别对它们进行合并操作得到活动 BC 和 HIJ,同时更新事实库;重新执行规则推理,得到的结果是活动 BC 和 D 为相似活动,活动 HIJ 为独立活动,分别对它们进行合并和并发操作,更新事实库;再次执行规则推理,得到的结果为空,优化结束。优化后的结果如图 4 所示。

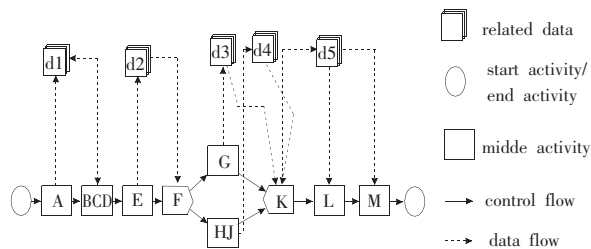


图 4 优化后发文流程

通过比较可以得到,该方法简化了工作流过程,同时也减少了数据的传送次数,在一定程度上,特别是在工作流的结构上实现了对工作流的优化,如表 3 所示。

表 3 优化结果比较

	节点数	数据传送次数	相似集合数	并发集合数	独立集合数
优化前	15	15	2	2	1
优化后	11	11	0	0	0

本文的问题规模主要取决于工作流的三类节点集合的数目。通过实验得到,三类节点的集合数目与优化结果的效率是成正比的。因此,通过基于语义的工作流过程优化,消除了以上三类节点的活动,从结构上优化了工作流过程。

5 总结

本文提出了一种基于语义的工作流过程优化方法,它是建立在本体推理系统的基础上的,通过对已有 Owl-s 本体的语义扩充,实现工作流本体库的构建,利用各种实体间的约束关系来保证工作流模型的合理性,结合特定的推理规则实现对工作流活动的分类,根据这些分类信息执行相应的操作,从而实现对工作流过程的结构优化。通过对实例的分析,可以说明该方法的可行性和正确性。本文的重点在于说明基于本体推理系统的工作流优化方法的可行性,针对不同的应用,可以采用不同的本体描述和规则推理方法来实现对工作流的动态优化。

工作流优化除了结构方面的优化,还有工作流的性能优化。如何结合实际的工作流应用,制定出针对某一特定性能优化规则,实现工作流性能优化,这是下一步的研究方向。

参考文献:

- [1] Heterd N, Binder W, Constantinescu I, et al. Optimal workflow execution in grid environments[C]//2nd International Conference on Grid Service Engineering and Management, September 2005: 19-22.
- [2] Reijers H A. Design and control of workflow processes[M]. Berlin: Springer-Verlag, 2003.
- [3] Vieira T A S C, Casanova M A, Ferrao L G. An ontology-driven architecture for flexible workflow execution[C]//10th Brazilian Symposium on Multimedia and the Web 2th Latin American Web Congress, Ribirao Preto-SP, 2004: 70-77.
- [4] Wang S, Shen W, Hao Q. Agent based workflow ontology for dynamic business process composition[C]//The 9th International Conference on Computer Supported Cooperative Work in Design, Coventry, UK, 2005: 452-457.
- [5] Owl-S.[EB/OL](2004-11).http://www.daml.org/services/Owl-s/1.1/.
- [6] SWRL.[EB/OL](2004-05-21).http://www.w3.org/Submission/SWRL/.
- [7] Jess.[EB/OL](2008-02-21).http://herzberg.ca.sandia.gov/jess/.
- [8] Beco S, Cantalupo B, Giammarino L. Owl-WS: a workflow ontology for dynamic grid service composition[C]//Proceedings of the First International Conference on e-Science and Grid Computing (e-Science'05), Nottingham, UK, 2005: 148-155.
- [9] Perumal S, Mahanti A. A graph-search based algorithm for verifying workflow graphs[C]//Proceedings of the 16th International Workshop on Database and Expert Systems Applications, 2005: 992-996.
- [10] Bi H, Zhao J. Applying propositional logic to workflow verification[J]. Information Technology and Management: Special Issue on Workflow and E-business, 2004, 5(3-4): 293-318.
- [11] Huang Lican, Walker D W, Huang Yan, et al. Dynamic Web service Selection for workflow optimisation[C]//Proceedings of the UK e-Science All Hands Meeting, September 2005.
- [12] Alonso G, Agrawal D, Abbadi A E, et al. Functionality and limitations of current workflow management systems[J]. IEEE Expert, 1997, 2(5).
- [13] 谢玉凤, 杨光信, 史美林. 基于条件化有向图的工作流过程优化[J]. 计算机学报, 2001(7): 729-735.