ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.12, December 2006, pp.2554−2564
DOI: 10.1360/jos172554

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

面向事件处置的信息服务集成调度模型*

罗英伟[1,2+]，刘昕鹏[1]，彭豪博[1]，汪小林[1,2]，许卓群[1]

[1](北京大学 计算机科学技术系,北京 100871)

[2](石河子大学 信息科学技术学院,新疆 石河子 832003)

Information and Services Integrating and Scheduling Model for Event Handling

LUO Ying-Wei[1,2+], LIU Xin-Peng[1], PENG Hao-Bo[1], WANG Xiao-Lin[1,2], XU Zhuo-Qun[1]

[1](Department of Computer Science and Technology, Peking University, Beijing 100871, China)

[2](School of Information Science and Technology, Shihezi University, Shihezi 832003, China)

+ Corresponding author: Phn: +86-10-62763373, Fax: +86-10-62759444, E-mail: lyw@pku.edu.cn, http://gis.pku.edu.cn

Luo YW, Liu XP, Peng HB, Wang XL, Xu ZQ. Information and services integrating and scheduling model for event handling. *Journal of Software*, 2006,17(12):2554−2564. http://www.jos.org.cn/1000-9825/17/2554.htm

Abstract: Around city applications such as E-Government, E-Business, etc, event handling becomes a common activity. When handling an event in a city, people should focus on the task of integrating, conforming and scheduling heterogeneous, distributed resources from multi-domains, as well as sharing knowledge among different organizations. This paper proposes a rule-based distributed resources integrating and scheduling model for event handling. The model has a hierarchical structure with four levels: resource layer, knowledge layer, business layer and representation layer. Detail work on the knowledge layer is explored, which includes the definition of formal business rule, reference to resources in a rule, as well as the design and implementation of the rule system. Finally, an event handling prototype system of unitive alarm taking in for iEMS (integrated emergency management system) is implemented to validate the model.

Key words: event; resource; ontology; rule; integrating and scheduling

摘 要: 以电子政务、电子商务等城市应用为背景,针对这些应用中跨领域事件处置所涉及的异构、分布式资源、服务集成调度以及跨领域知识共享问题,提出了一个面向事件处置的、基于规则的信息服务集成调度模型.该集成调度模型具有 4 个分层结构,包括资源层(resource layer)、知识层(knowledge layer)、业务层(business layer)和表示层(representation layer).重点就知识层形式化业务规则的定义、资源和服务的引用以及规则引擎等给出了详细的设计和实现.最后,对该模型进行了验证,实现了一个城市应急联动应用中的统一接警事件处置原型系统.

关键词: 事件;资源;本体;规则;集成调度

中图法分类号: TP393 文献标识码: A

* Supported by the National Natural Science Foundation of China under Grant No.60203002 (国家自然科学基金); the National Grand Fundamental Research 973 Program of China under Grant No.2006CB701306 (国家重点基础研究发展规划(973)); the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No.20020001015 (国家教育部博士点基金)

Received 2005-10-26; Accepted 2006-01-12

随着信息化在城市建设和人类生活中的进一步深入以及以电子政务、电子商务等为代表的城市应用的广泛普及,事件已经逐步成为一种自然的信息表示和信息处理的概念实体.事件处理因而成为城市生活中一项经常性的事务,也是城市综合信息系统的一项重要任务.

在事件处置过程中,与事件相关的信息是整个事件处置的基础,而将与事件相关的信息和事件处置相关的经验知识及时送到处置者手中,是达成正确、高效的事件处置的关键.当前,城市中部门间合作紧密,在每个事件处置过程中所引用到的信息不可避免地具有跨领域、跨部门的性质,这给事件处理带来了数据存储分布、格式异构等问题,我们需要找到一种自然的、开销小的方式,从多数据源进行信息提取.

本文重点针对城市分布式事件处置的特点和需求,试图构造一个基于规则的分布式资源、服务集成调度模型,通过对分散在异构平台、信息系统的资源、服务进行动态集成,形成面向多领域的、具有统一语义的事件决策信息,以满足高效率事件处置的要求,并在此基础上建立一系列完整的、面向事件的知识体系,以及时、准确、人性化地辅助事件决策.

1 相关研究

Web Service 作为一种新型的互操作技术标准,在解决分布式互操作和 Internet 协作方面得到了最为广泛的认可.Web Service 能够以 WSDL 作为标准的接口语言对外发布本地可调用的软件模块,允许由不同的供应商提供的、或者运行在不同的操作系统和平台上的以不同编程语言书写的软件间存在互操作性.运行在 Web Service 纵向框架最顶层的业务流程,通过标准的基于 Web Service 组合得到的服务工作流语言进行描述,从而表达上层应用的业务需求和逻辑[1].

工作流机制能够满足具有稳定工作流程的业务在线组装,并形成多个分布式服务间的流程控制和数据传递[2,3].但在城市的很多应用中,许多事件的处置流程不仅关心分布的服务间的交互,而且对参与到事件处置中的分布式数据本身的在线获取、操作、集成以及分布式数据和服务间的交互也极为关注.在很多情况下,分布式数据是前端事件处置和决策参考的核心,分布式服务主要不是提供工作流程,而是为数据处理所用,以便更友好地提供给应用的用户.工作流及其分布式服务组合机制还不能有效地支持这一点[4].

面向事件的处置需要基于事件自身生命周期的发展、变化为主要线索来组织、管理、调度和浏览相关信息.事件的状态变迁是动态触发的,对于依据流程进行定义和调用的工作流而言也是不合适的.比较好的实现系统行为动态触发的机制是规则系统[5,6].

表达规则的方式很多,如 ECA(event-condition-action,事件-条件-动作)规则,或只有前件和后件组成的 CA(condition-action)规则.规则最初来源于主动数据库,它描述了触发动作的事件和内部条件以及它们和动作之间的依赖关系.在一个规则系统中,相关的多个规则共同作用来指导系统的行为.规则也被用于专家系统中,用来进行相关知识的推理和专家系统知识的更新.目前,规则在分布式信息共享和互操作方面的应用相对较少,但是在使用规则来扩展工作流的语义方面已有工作开展[7].

本文在结合城市应用中事件处置特征的基础上,引入 CA 规则描述事件处置所需的领域知识和业务规则,通过定义规则对事件处置所需的资源、服务的引用方式,以规则的触发机制实现分布式资源、服务的在线集成和互操作过程中的数据、服务基于领域语义指导的有效交互,增强对事件处置的辅助决策能力.

2 基于规则的信息服务集成调度模型设计

2.1 基于规则的信息服务集成调度模型的分层结构

一个完整的基于规则的信息服务集成调度模型可以用图 1 来描述,该模型的建设由外及内可以分为以下 4 个层次(在图中用虚线划分):

(1) 资源层(resource layer).资源层的主要目的是管理面向事件处置的分布式应用所需的各类资源和服务,屏蔽这些资源和服务在分布位置、数据结构、访问方式上的异构性,尽可能地在资源、服务这一粒度上提供完

整的元数据描述,为进一步表达事件处置中所涉及的领域知识的表达做准备.

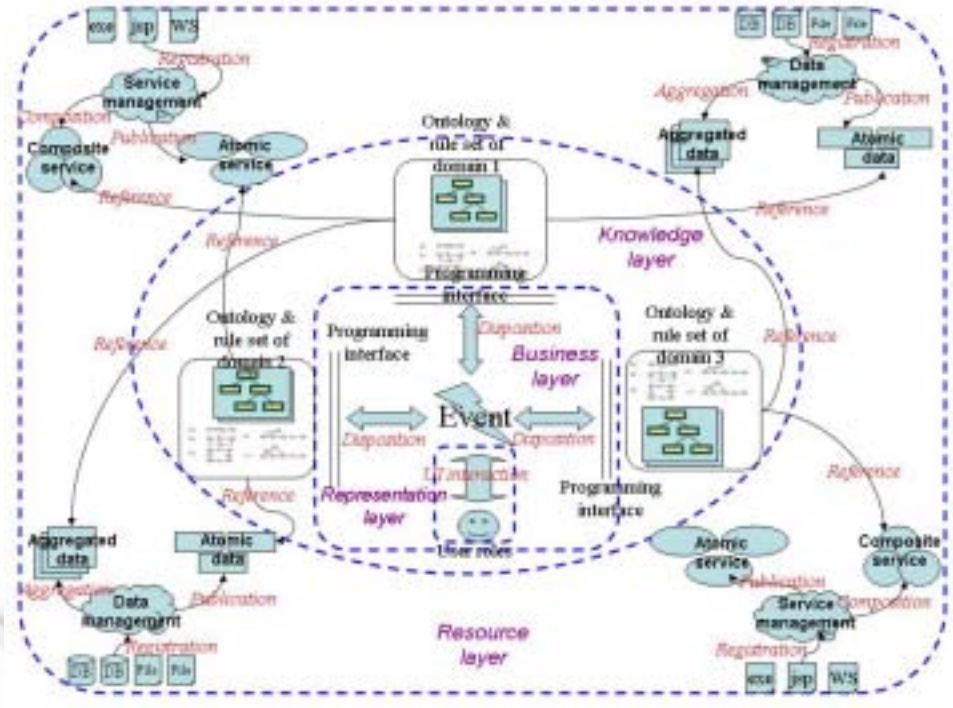

Fig.1 Architecture of rule-based information and services integrating and scheduling model

图 1 基于规则的信息服务集成调度模型的分层结构

从一般的表现形式上看,资源存储在数据库或者文件中,但是仅仅通过数据库的表结构或者文件名还不足以说明这些资源的用途[8,9],需要在参与到事件处置的多个节点上进行数据管理,以提供对数据更为丰富的描述.我们称这样的节点为"数据中心".当资源通过数据中心完成注册后,相关的元数据描述也必须同时被撰写和发布出来.数据中心除了可以依据这些元数据描述对外发布具有自描述能力的原子数据以外,有时也应上层业务的需求使用一些在应用背景中才有意义的更大粒度的数据.这些数据可以通过某些方式由多个原子数据生成.我们称这样的数据为"聚合数据"[10].

类似地,对于服务而言,它们往往以标准 Web Service 或者其他形式的可调用实体(如 EXE 文件、JSP 页面、DLL 文件、遗产系统服务接口等)出现.除了和调用相关的接口信息外,使用这些服务同样需要更为丰富的元数据进行描述.我们把类似服务管理能力的节点称作"服务中心".服务中心将能够注册以上各种服务实例,并对外发布具有自描述能力的原子服务.同样,上层业务考虑到某些应用需求也会经常使用通过原子服务组成的工作流,因而,服务中心还可以发布由此形成的服务组合.

(2) 知识层(knowledge layer).知识层主要为参与到分布式业务流程中的各个领域的概念建立本体模型,用领域内统一的词汇描述领域知识.本体中将引用由资源层规整得到的原子数据、聚合数据、原子服务和服务组合.本体对这些资源层要素的引用主要基于资源层元数据描述中相关数据和服务的名称、属性,而不涉及它们的数据格式、分布位置和访问方式,后者由资源层完成抽象.知识层的建设专注于对领域语义的阐明.

考虑到事件处置的特点,除了为各个领域建立相关的本体模型以外,我们将事件各个阶段处置中的经验知识抽象为 CA 规则,这样的规则称为"业务规则".规则的词汇集来源于各个领域的本体,从而对规则的调用间接形成了对资源层的资源、服务的引用.由于事件处置的经验被抽象出来形成规则,系统具有更大的灵活性.而基于规则的调用机制可以完成规则的语法表达、语义解释、分析执行与规则所表达的业务逻辑之间的松耦合,使得新旧规则之间的替换成为可能,有利于在较大范围、较长时间内维持分布式事件处置体系的可用性.

(3) 业务层(business layer).知识层中领域本体和规则的制定为面向业务开发相关业务系统提供了丰富的语义库.由于本体和规则得到了规范的表达,模型有可能基于它们提供面向本体、规则的编程接口,从而为访问这些语义信息提供入口.基于这些编程接口,面向事件的开发人员可以专注于一个具体事件处置的业务逻辑进行设计、开发,并在业务流程中需要使用相关语义信息进行决策分析时,调用这些接口.

(4) 表示层(representation layer).事件具有鲜明的阶段性,在不同的阶段处置方式不同:有的只需要进行后台的数据分析;而有的需要将信息多样化显示;还有的甚至需要和用户进行交互完成相关处置.表示层将在业务层之上进行相关的用户界面设计.在本文开始部分对面向事件处置特征的分析中提到:面对不同角色的用户,事件处置需要展现的内容不同,展现的方式也不同.将用户界面从业务逻辑中分离出来,就可以提供实现业务层单一处置流程、资源层单一信息存储下,表示层用户界面多样化的可能性.

2.2 业务规则表达和基于规则的分布式资源、服务集成调度

基于规则的信息服务集成调度模型的核心是知识层.在知识层建设展开之前,模型在资源层的数据中心和服务中心已经通过数据元数据和服务元数据描述完成了对与分布式事件处置相关的资源和服务的封装.

数据中心发布的数据项将以

$$DataType(DataAttribute_1,\ldots,DataAttribute_n)$$

的格式给出.数据中心对注册到其中并加以维护的各种数据实例都赋予一个逻辑上的数据类型 *DataType*,同属于一个逻辑数据类型的数据实例拥有相同的属性集合$\{DataAttribute_i\}$.对数据中心数据的引用,只需要指定数据实例的名称 *DataName* 和引用的属性集合,而由数据中心依据元数据定位该数据的逻辑类型 *DataType*.

服务中心发布的服务将以

$$ServiceName(ServiceArgument_1,\ldots,ServiceArgument_n):ServiceReturnValue$$

的格式给出.服务中心对注册到其中并加以维护的各种服务都赋予一个逻辑上的ID,该ID对应到描述该服务的接口细节,包括服务名称 *ServiceName*、服务参数列表$\langle ServiceArgument_i\rangle$和服务的返回值 *ServiceReturnValue*.其中,服务的参数和返回值在逻辑上有 3 种来源:常数值、数据中心的数据和其他服务的返回值.

2.2.1 业务规则的定义及表达

资源层发布的资源、服务为使用规范化方法表达业务规则提供了基础.每条业务规则由其前件、后件组成.前件是一个布尔表达式,它反映了规则执行的条件.其内容从语法上看,是通过对数据的属性取值、词汇间的语义关系以及服务的返回值进行条件约束,表达本条规则满足执行条件时业务环境的状态;从语义上看,则定义了该业务规则所描述的事件处置经验或者领域知识适用的前提.后件是一个执行动作或者动作序列,它反映了规则执行的动作.其内容从语法上看,体现为模型资源层的一个服务或者服务流;从语义上看,则定义了该业务规则所希望表达的领域知识或者事件处置经验建议的处置行为和手段.以下为规则的规范化表达文法:

Rule=if *Condition* then *Action* 1
Condition=*BoolItem*|*BoolExpr* 2
BoolItem=*BoolConst*|[*NOT*]*BasicItem* 3
BoolConst=True|False 4
BasicItem=*RelationItem*|*BoolOper* 5
RelationItem=*AtomicItem Relation AtomicItem* 6
AtomicItem=*Term*|*Service*|*Const* 7
Term=*TermName*[.*Aspect*] 8
TermName=*Metadata*::*DataName*|*CriticalRange*::*DataName*|*Ontology*::*Resource* 9
Aspect=*Metadata*::*DataArrtibute*|*CriticalRange*::*DataArrtibute*|*Ontology*::*Property* 10
Relation=>|=|<|>=|<=|<>|*Ontology*::*Property* 11
BoolExpr=*BoolItem BoolOper BoolItem* 12
BoolOper=AND|OR 13
Action=*Service*|*ServiceFlow* 14
Service=*Metaservice*::*ServiceName*(*Params*) 15
Params=[*Param*]|{,*Param*} 16
Param=*Const*|*Term*|*Service* 17

Const=Constant of Type ⟨*String*|*Integer*|*Float*|*Double*|*Boolean*|*Object*⟩ 18
ServiceFlow=DAG_Formatted(*Service*,{*Service*}) 19

• 在表达式 1 中,规则的前件用 Condition 表达,后件用 Action 表达.Condition 通过文法的递归定义表示了数据中心发布的数据及其属性取值、服务中心发布的服务的返回值以及知识层本体中的词汇之间的取值约束和布尔逻辑关系.

• 在表达式 11 中,规则中引用的关系运算符除了基本的比较运算符以外,还可以包括领域本体中的属性.通过这样的属性,可以描述运算操作数(这里应该为本体中的词汇)之间更为丰富的语义关系.需要说明的是,这里引用的本体属性的值域(range)应该为布尔型.

• 表达式 17 描述了规则中服务参数可以有 3 种来源:常数、数据中心发布的数据、其他服务的返回值.

• 在表达式 19 中,服务流是多个服务的组合,服务流可以通过有向无环图(directed acyclic graph,简称 DAG)来描述成员服务之间的执行顺序和约束关系.

2.2.2 业务规则中对资源和服务的引用

从第 2.2.1 节的规范化业务规则的文法描述中可以看出,业务规则对分布式资源和服务的引用方式为:

• 在表达式 9 和表达式 10 中,规则的前件中引用的基本词汇可以是数据中心发布的数据名称及其属性名称(在文法表达式中,用前缀 *Metadata*::标明),也可以是知识层领域本体中的资源(resource)和描述资源间语义关系的属性(property,它们在文法表达式中用 *Ontology*::前缀标明).

• 规则的调用执行会产生一些导出数据.这些数据不是来自数据中心,而主要是在规则的调度执行过程中由调用规则的业务逻辑来设置;最终对这些数据的改变也不需要更新到数据中心.但是,这些数据对记录规则的执行对系统状态的改变有着重要意义,需要在事件处置过程中在内存进行维护.我们称这些数据为临界区数据(在文法表达式中用 *CriticalRange*::前缀标明).

• 在表达式 14 中,规则对服务的引用主要体现在后件的 Action 中.一个 Action 可以是一个 Service(它表示了服务中心的一个服务实例,如表达式 15 所示,用前缀 *Metaservice*::标明),也可以是一个 ServiceFlow.

• 在表达式 19 中,ServiceFlow 可以通过有向无环图来描述成员服务之间的执行顺序和约束关系.

• 在表达式 7 中,服务也可以出现在规则的前件表达式中,主要表达在相关数据或者其他服务的返回值上的复杂操作.这些操作的结果被用来进行取值和逻辑判断,以表达更为丰富的前件语义.

在规范化规则的文法描述中,描述业务所需的分布式资源都只是引用它们在逻辑上的组成,完全屏蔽了资源的位置、格式、发布平台以及接口方式.在资源层对这些信息屏蔽不仅简化了规则的编写,而且将事件处置的逻辑和用于决策的信息的技术特征分离,使得资源的配置更为灵活,业务逻辑的更新也更为便捷.

2.2.3 规则引擎和规则的调度执行

经过上述知识层的建设,在面向一个具体的事件处置应用时将形成大量的规则.我们将语义上相关的一系列规则的逻辑集合称为一个规则池.当规则池中规则的数目急剧上升时,如何高效地面向前端的应用组织规则之间的关系,并进而完成相关规则的分析、调度成为一个关键问题.这需要在知识层提供一个规则引擎来集中完成对规则的调度执行.我们对规则引擎的设计主要参考了著名的主动数据库 Ariel 的规则系统的体系结构[11].图 2 给出了基于规则的信息服务集成调度模型中规则引擎的内部设计,以及在具体的规则执行周期中规则引擎和业务层、资源层以及知识层进行信息交互的流程.

• 业务程序对临界区数据池的更新操作作为规则的触发事件(标号为 1 的箭头).业务程序对临界区数据的更新操作具有不同粒度.从最小的粒度看,业务程序修改临界区每个变量的值都可以算是一个原子性的物理操作.但很多情况下,业务程序希望在一个对应用而言有意义的操作序列结束后,再一次性地更新临界区数据集的值,触发规则执行周期,完成系统状态的转换.这个操作序列可以看作是一个逻辑事件,它在面向事件的处置中更为合适,更有利于提高处置效率.

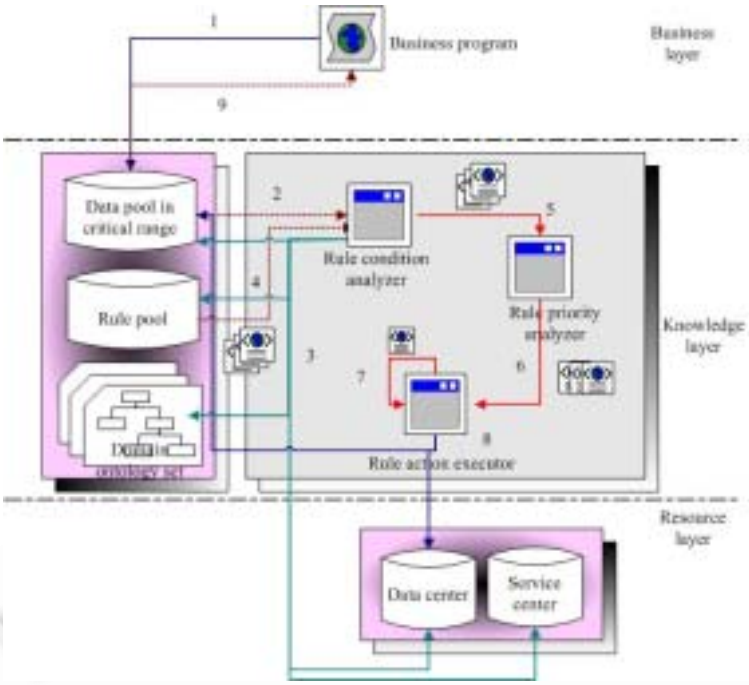

Fig.2 Design of rule engine for the model

图 2 模型中规则引擎设计

• 对临界区数据的更新将触发一个规则执行周期的开始(标号为 2 的箭头).周期的第 1 步由规则条件分析器进行条件分析.该分析器装载规则池中每条规则的前件,并在必要时访问临界数据池、领域本体集、数据中心和服务中心,获取前件中所引用的临界数据、本体词汇、数据和服务的相关取值,计算前件的真假值(标号为 3 的箭头).分析完成后,规则池将所有满足当前条件的规则集返回给条件分析器(标号为 4 的箭头).

• 规则执行周期的第 2 步是进行待决规则的优先级分析,由规则优先级分析器完成.优先分析器以条件分析器输出的待决规则集作为输入(标号为 5 的箭头),通过计算每条规则的优先级取值,按照预期的规则执行顺序输出规则的排序列表(标号为 6 的箭头).

• 规则执行周期的第 3 步是根据第 2 步得到的规则排序执行每条规则(标号为 7 的箭头).每条规则的执行方式是执行规则后件中描述的服务和服务流.这些服务和服务流的调用不仅会引用相关的数据、词汇、服务作为服务的输入,还可能在执行后对临界区数据池、数据中心进行数据更新(标号为 8 的箭头).规则执行器依次完成规则的执行,直到所有的规则后件被触发调用完毕.

• 一个规则执行周期之后,当业务程序再访问临界区数据池时(标号为 9 的箭头),将获得新的临界数据值,规则执行环境一次转换就完成了.当业务流程再次更新临界数据时,一个新的业务执行周期又开始了.

3 基于规则的信息服务集成调度模型实现

3.1 基于规则的信息服务集成调度模型的实现规划

基于规则的信息服务集成调度模型的实现规划如图 3 所示.

第 1 步,事件处置过程中所需的分布式资源在资源层中数据中心、服务中心的注册和发布,是通过定义资源的元数据规范提供知识层领域本体对资源引用的 API 接口;第 2 步,数据中心和服务中心还应提供相应的 API 接口,使得知识层业务规则在进行撰写、解析以及分析执行时具有统一的入口.第 1 步、第 2 步共同形成的 API 接口集将作为知识层和资源层的交互界面.

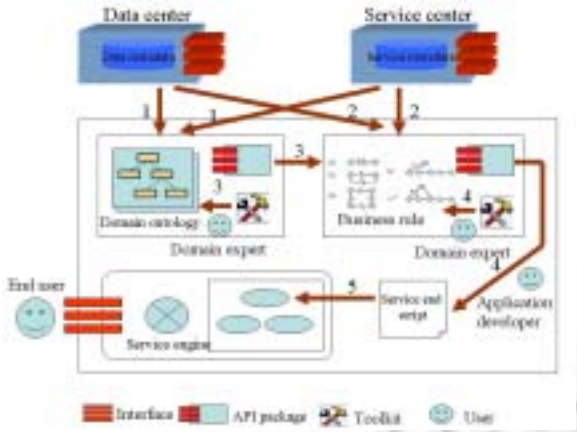

Fig.3 Implementation layout of the model

图 3 模型的实现规划

第 3 步、第 4 步为知识层的建设.第 3 步将针对面向事件处置所涉及的领域本体的特征,提供支持常用本体规范的本体编辑器,以及能够对本体文件进行领域分类、存储和维护的本体仓库管理工具.另外,由于规范化规则的表达也引用了本体词汇,因而还将发布能够检索本体仓库、访问本体模型的 API 接口,作为知识层内部本体知识访问的统一入口.

第 4 步将根据规范化规则文法制定描述业务规则的 XML Schema,并基于该 Schema 提供规则编辑工具,支持规则对数据中心、服务中心、领域本体模型的引用;其次,开发业务规则的分析器、选取器以及执行器,实现规则引擎;最后,提供面向模型业务层业务编程所需的 API 集,支持对领域本体、业务规则的引用入口以及对规则子系统运行周期的控制.

第 5 步是使用基于规则的信息服务集成调度模型进行前端事件处置业务实现时的一种典型方式.在这种方式下,事件处置逻辑被实现为多个服务端脚本,并部署到服务引擎中.每个脚本使用知识层提供的 API 触发规则的执行周期,完成相关的辅助决策分析.最终,用户将间接地通过服务引擎发布的界面参与到业务流程的处置过程中,完成相关的人机交互.

3.2 业务规则的XML表达

在如上所述的模型实现规划中,业务规则和规则引擎的实现是整个模型建设的核心.每条业务规则在实现时被描述为一个 XML 文件.该 XML 文件的 XML Schema 根据第 2.2.1 节的规范文法描述制定.描述业务规则的 XML 根元素为 Rule,该元素主要包括以下几个子元素:

• Comment,该元素通过一个文本子节点描述了该规则的业务含义.

• Condition,该元素通过一个迭代定义的 XML 子树描述了该条业务规则的前件表达式.表达式操作符的优先级定义基于业务规则的文法.Condition 元素的所有子节点通过 XML 元素的包含关系体现表达式操作符的优先级,即每个父元素描述表达式的操作符,该父元素的子元素描述该操作符的操作数(一个具体的操作数或者另一个操作符),父元素中的操作符将比子元素中的拥有更高的优先级.

• Action,该元素为规则的后件,它可以包含一个 Service 或 ServiceFlow 子元素.其中,Service 将引用服务中心发布的一个服务实例;而 ServiceFlow 则通过其子元素的进一步描述,表达服务中心发布的一组服务实例之间的流程控制和数据交换关系,从而形成一个工作流.

• References,该元素可以包含 3 个子元素:OntologyConnections,DataCenterConnections 和 ServiceCenterConnections.这 3 个子元素各自列举了该规则描述中在 Condition 和 Action 定义中表达资源、服务和本体词汇时对本体仓库、数据中心和服务中心的引用.例如,下面是一条规则实例中描述 Conditions 时所引用的服务实例:

〈*Service ID*=“1” *Name*=“min*Distance*” *ConnectionID*=**“2”**〉

〈*ParamList*〉…〈/*ParamList*〉

〈/*Service*〉

该 Service 通过 ConnectionID 引用服务实例 minDistance 所在的服务中心,相应的 References 的描述为:

〈*References*〉

…

〈*ServiceCenterConnections*〉

〈*ConnectionDesc* ***id*=“2”**〉*TCP*:**168.168.168.220:1601**〈/*ConnectionDesc*〉

〈/*ServiceCenterConnections*〉

〈/*References*〉

它说明访问该服务中心的描述符为 TCP:168.168.168.220:1601,由此可以获得对该服务实例的访问入口.

多个逻辑上相关规则文件形成一个规则池.规则引擎将完成对规则池中规则的调度执行.

3.3 规则引擎的实现

规则引擎主要由以下几部分组成:

- 临界区数据池,它用来存储和维护临界区的所有数据(见第 2.2.1 节).同时,它也提供了一系列的方法来进行数据管理,包括添加一个新的临界区数据到数据池中、查询符合条件的临界区数据、删除符合条件的临界区数据.而这些操作都将触发规则装载器、分析器和执行器开始一个新的规则执行周期.
- 规则装载器,其主要功能是读入规则文件,完成文件解析,并装载到规则的内存模型中来.
- 规则分析器,其主要功能是分析由装载器得到的规则内存模型的前件表达式,通过维护表达式的算符优先关系,自底向上计算得到规则前件在当前情况下的真假值,选出满足条件的规则集合.
- 规则执行器,其主要功能是执行满足条件的规则内存模型后件中的行为.
- 规则引用容器,它能够创建对应于规则文件 References 元素中各种类型的连接,并能够根据连接的描述符和类型对连接进行枚举.

规则的内存模型、装载器、分析器、执行器以及临界区数据池的实现和相互关联,形成了模型知识层规则引擎的基本框架.基于规则引擎和其他知识层的相关建设,就可以向业务层提供更多的编程接口,方便业务逻辑中对规则执行周期的控制,使规则的执行起到辅助决策的作用.

4 基于规则的信息服务集成调度模型实例——统一接警事件处置

4.1 城市应急联动系统和统一接警事件处置

城市中跨领域事件处置的一个典型应用是应急事件的联动处置.社会应急联动系统(joint emergency response system,简称 JERS)[12]就是综合各种城市应急服务资源,采用统一的号码,用于公众报告紧急事件、紧急求助、统一接警、统一指挥和联合行动,为市民提供相应的紧急救援服务,为城市的公共安全提供强有力保障的系统.

统一接警是应急联动应用中对紧急事件展开处置的第一步,这里主要有两个用户角色:报警者和统一接警员.统一接警的基本业务就是结合报警者和统一接警员的角色,为统一接警员提供统一的警情分析、事件记录界面,协助统一接警员完成对当前信息的准确定位和规范化描述,并转给下一阶段联动处置调度所用.统一接警中计算机所起到的主要作用是自动搜集与分析事态相关的信息、服务、领域知识和业务经验,通过对统一接警员在接警过程中的输入进行分析,提供最具参考价值的后继问询或行动建议,以达到辅助决策的目的.

4.2 基于规则的信息服务集成调度模型下的解决方案

我们将采用以下假想的应急联动事件作为模型的演示案例:

天津市某区某化学工厂于夜间发生了一起火灾.该工厂以生产白磷等化学药品为主,而白磷是一种剧毒的易挥发性化学物质,在 40℃时就会变为无色气体.火灾发生在离保存白磷等化学药品的仓库不远处,并且火势很快蔓延到距离该化工厂约 200 米处的居民住宅区内,造成了部分住宅为火所困.

在这起火灾中,一名技术工人向联动中心的统一接警员汇报了火情和相关情况,接警员借助面向事件的分

层信息服务模型对事态进行分析,并全面记录事件要素,提出了处置建议.借助于模型中的领域本体和业务规则,我们希望在接警中避免忽略由火灾带来的潜在化学危害,并为下一阶段的事件处置在火警救援和医疗急救两方面提出建议.在该演示系统中,资源层部署的数据和服务见表 1,所引用的规范化业务规则见表 2.

Table 1 Data and services in the prototype system

表 1 统一接警演示中资源层部署的数据、服务

Data/Service name	Data/Service granularity	Data/Service name	Data/Service granularity
StaffBasicInfo	Atomic data	TJFireStationLayer	Atomic data
StaffStateInfo	Atomic data	TJKeyBuildingMap	Aggregated data
StaffEnterpriseInfo	Atomic data	TJFireDivisionMap	Aggregated data
StaffEducationInfo	Atomic data	getMapLayerField	Atomic service
StaffIntegratedInfo	Aggregated data	minDistance	Atomic service
TJResidentLayer	Atomic data	getCoord	Atomic service
TJFactoryLayer	Atomic data	showFireStationSchedule	Composite service
TJStateLayer	Atomic data	isVIP	Atomic service
TJRoadLayer	Atomic data		

Table 2 Rules in the prototype system

表 2 统一接警演示中的业务规则

Domain	Rule description in specification	Business signification of rule
Common	if true then Ask(Ontology::QuestionTable#Location,Ontology::QuestionTable #State,Ontology::QuestionTable #VictomState)	New event emerged, should query the location, state and accident victim state, etc.
	if(CriticalRange::Location<>null) then Set(CriticalRange::LocationCoords,Metaservice::getCoord(CriticalRange::Location,Metadata::TJKeyBuildingMap))	Obtain the exact coordinates of the accident location.
Fire fighting	if (CriticalRange:: VocationDivision = 'Chemistry' AND CriticalRange::FirePosition = 'Garage') then Set (CriticalRange::ChemicalProduct, Metaservice::getMapLayerField(Metadata::TJFactoryLayer,"Product",CriticalRange::Location))	A fire near some chemical plant garage, should check the chemical products of that plant.
	if (CriticalRange::State= "FireAccident") then Ask(Ontology::FireQuestionTable#VocationDivision,Ontology::FireQuestionTable#BurningSource,Ontology::FireQuestionTable#BuildingType,Ontology::FireQuestionTable#FirePosition)	A fire accident, should query the burning kind, burning position and damaged building situation, etc.
	if (CriticalRange:: LocationCoords 〈〉null AND CriticalRange::State= "FireAccident") then showFireStationSchedule(CriticalRange::LocationCoords,Metadata::TJKeyBuildingMap)	Calculate and exhibit the reinforcement routes of each fire station for further reference.
	if (CriticalRange::Time ="Night" AND CriticalRange::State = "FireAccident") then Record("Nighttime fire accident, lighting car dispatch are recommended.")	Register decision-making references – Nighttime fire accident, lighting car dispatch are recommended
	if(CriticalRange::ChemicalProduct Ontology::subClassOf Ontology::FirstAidQuestionTable#PoisonCause.Ontology::FirstAidQuestionTable#Poison AND Ontology::FirstAidQuestionTable#Volatile Ontology::value CriticalRange::hemicalProduct AND MinDistance(CriticalRange::LocationCoords,Metadata::TJResidentLayer)<200) then Record("Dangerous volatile chemical around the fire, which is within 200 meters distance to residence, emergent medical care demanded.")	Register decision-making references – Report of VIP injury during fire accident.
First aid	if ((CriticalRange::VictomState = 'Dead' OR CriticalRange::VictomState = 'Surrounded') AND Metaservice::isVIP(CriticalRange::VictomName)=true) then Record("VIP injury during fire accident, immediate report recommended.")	Register decision-making references – leakage of dangerous chemical, personal first-aid is recommended.

统一接警演示在业务层的业务流程可以用图 4 来表示.

当统一接警员接到一个新的报警后,他将能够通过 Web 网页浏览到接警业务初始界面.一个接警业务界面包含了该事件所涉及的各个领域的问题表表单,以及用于反映事件空间处置策略的电子地图.接警业务初始界面提示接警员向报警者询问独立于领域的事件要素信息(步骤①),并从报警者处获得回复(步骤②),填充到问题表的相应域段中(步骤③);

接警业务界面将已填充的问题表表单提交给服务端接警业务脚本(步骤④).服务端业务脚本通过调用知识层提供的 API 接口集装载面向统一接警事件处置的规则库,并启动知识层规则子系统的执行周期,触发当前满

足条件的规则的执行,并最终将相关的执行结果反馈给服务端业务脚本(步骤⑤);

服务端业务脚本将更新的信息通过新的 Web 页面展现给统一接警员(步骤⑥).新的接警业务界面或者能够通过新的问题表表单提示接警员向报警者进一步询问与本事件更为相关的细节问题,或者提示统一接警员考虑其他的事件状况分析,或将相关的事件空间处置策略在电子地图中展示出来;

接警业务重复以上步骤①~步骤⑥的流程,直到完成对本次事件要素的全面记录.最终,接警业务脚本将在线生成事件记录和处置建议的汇总(步骤⑦),并通过 Web 页面在接警业务界面中展现给统一接警员(步骤⑧);

统一接警员从接警页面获得最终的事件信息和处置建议(步骤⑨),完成了联动事件的接警过程,将联系有关单位和部门开展跨领域事件处置.

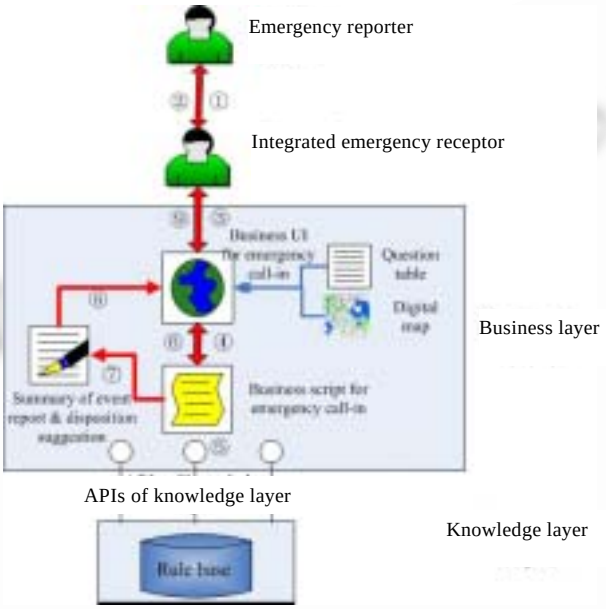

Fig.4 Business process of the prototype system

图 4 统一接警演示在业务层的业务流程

5 结束语

本文提出了一个基于规则的信息服务集成调度模型.通过模型的初期设计和实现,验证了模型在解决现实中相关应用的可行性和有效性.使用规则表达事件处置中的领域知识和业务经验,并基于规则的触发机制来调用与事件相关的资源和服务,辅助事件分析和决策是一个新的思路和探索.与基于工作流的资源、服务组合方式相比,规则在面向事件的处置中的优势在于它更加适合城市中各类事件的发展变化特征.城市中应急事件处理尤其具有这种特点.使用规则不仅复用了既往事件中有益的处置策略,还能通过规则的更新反映出针对未来事件的新的处置方式,从而弥补失误,减少不必要的代价.当然,通过实践我们也发现,要使得这样的一个信息服务模型真正成熟起来,仍然有很多问题需要在进一步的工作中加以解决:

1) 针对大型事件应用,面向大量规则时规则系统的处理能力.这不仅需要进一步建立和完善规则系统中的优先分析器,建立面向事件的规则优先体系,还需要对规则分析器进行优化,以便在每个规则触发周期来临时快速匹配到高优先级的规则群体.

2) 提供规则编辑器的初衷是为领域专家提供编辑业务规则的友好界面.但是我们发现,目前的规则描述基本上还处于技术执行层面,还不能脱离资源层的数据、服务提供纯语义层面的表达.在未来,可以在目前的规则之上提供高层的语义规则.这种语义规则的表达完全使用领域词汇,而由模型定义两层规则之间的映射机制.这种映射机制还可以被用来解决规则的语义引用和资源层的物理提供不相匹配时的问题.

规则中的行动不仅可以包括资源层的服务,还可以包括完成事件处置的各种行为.如何扩展规则的后件定义并在业务层提供对这些自定义行为的合理开发方式,是基于该模型的前端应用的一个关键问题.

References:

[1] Hu CM, Huai JP, Sun HL. Web service-based grid architecture and its supporting environment. Journal of Software, 2004, 15(7):1064−1073 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/15/1064.htm

[2] Zhao W, Hu WH, Zhang SK, Wang LF. Study and application of a workflow meta-model. Journal of Software, 2003,14(6): 1052−1059 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/1052.htm

[3] Ding K, Jin BH, Feng YL. Modeling and analysis of transactional workflows. Chinese Journal of Computers, 2003,26(10): 1304−1311 (in Chinese with English abstract).

[4] Hu HT, Li G, Han YB. An approach to business-user-oriented larger-granularity service composition. Chinese Journal of Computers, 2005,28(4):694−703 (in Chinese with English abstract).

[5] Stonebraker M, Hanson EN, Potamianos S. The POSTGRES rule manager. IEEE Trans. on Software Engineering, 1988,14(7): 897−907.

[6] Agrawal RH, Cochrane R, Lindsay B. On maintaining priorities in a production rule system. In: Proc. of the 17th Int'l Conf. on Very Large Data Bases. 1991.

[7] Hu JM, Zhang SS, Yu XY. A workflow model based on ECA rules and activity decomposition. Journal of Software, 2002,13(4): 761−767 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/761.pdf

[8] Chirathamjaree C, Mukviboonchai S. The mediated integration architecture for heterogeneous data integration. In: Proc. of the IEEE TENCON 2002. 2002.

[9] Zamboulis L. XML data integration by graph restructuring. In: Proc. of the BNCOD 2004. LNCS 3112, Springer-Verlag, 2004. 57−71.

[10] Zhang JW. xmlToyBrick: Design and implementation of a semantic-driven, dynamic XML data integration tool [MS. Thesis]. Beijing: Peking University, 2006 (in Chinese with English abstract).

[11] Hanson EN. The design and implementation of the ariel active database rule system. IEEE Trans. on Knowledge and Data Engineering, 1996,8(1):157−172.

[12] Wang WJ. An architecture of web service interoperability for application in event preparedness plan over virtual organization [Ph.D. Thesis]. Beijing: Peking University, 2004 (in Chinese with English abstract).

附中文参考文献:

[1] 胡春明,怀进鹏,孙海龙.基于 Web 服务的网格体系结构及其支撑环境研究.软件学报,2004,15(7):1064−1073. http://www.jos.org.cn/ 1000-9825/15/1064.htm

[2] 赵文,胡文蕙,张世琨,王立福.工作流元模型的研究与应用.软件学报,2003,14(6):1052−1059. http://www.jos.org.cn/1000-9825/14/1052.pdf

[3] 丁柯,金蓓弘,冯玉琳.事务工作流的建模和分析.计算机学报,2003,26(10):1304−1311.

[4] 胡海涛,李刚,韩燕波.一种面向业务用户的大粒度服务组合方法.计算机学报,2005,28(4):694−703.

[7] 胡锦敏,张申生,余新颖.基于 ECA 规则和活动分解的工作流模型.软件学报,2002,13(4):761−767. http://www.jos.org.cn/1000-9825/13/761.pdf

[10] 张建伟.xmlToyBrick:一个 Web 环境下基于语义的动态 XML 数据集成系统的设计与实现[硕士学位论文].北京:北京大学,2005.

[12] 王文俊.Web 服务互操作的体系结构与跨管理域的 VO 事件预案支撑技术[博士学位论文].北京:北京大学,2004.

罗英伟(1971－),男,江西吉安人,博士,副教授,主要研究领域为分布式计算,GIS.

汪小林(1972－),男,博士,副教授,主要研究领域为分布式计算,GIS.

刘昕鹏(1981－),男,硕士生,主要研究领域为 GIS,应用集成.

许卓群(1936－),男,教授,博士生导师,主要研究领域为科学计算可视化,并行计算,人工智能,GIS.

彭豪博(1981－),男,硕士生,主要研究领域为 GIS,应用集成.