# 基于系统中心本体的分层抽象模型

王 楠[1,3] 孙善武[2] 欧阳丹彤[3]
(吉林财经大学管理科学与信息工程学院 长春 130117)[1] (吉林财经大学网络实验中心 长春 130117)[2]
(吉林大学计算机科学与技术学院符号计算与知识工程教育部重点实验室 长春 130012)[3]

**摘 要** 本体类分层关系的确定,使得能够利用不同抽象度本体类之间的抽象映射自动构建物理世界的分层模型。根据本体类型的不同将已有的本体类定义为对象本体类(Object-based Ontology Class),并对流片段(Flow Fragment)的概念加以扩展,提出以系统为中心的流本体类(Flow-based Ontology Class)的概念。定义了流片段的行为不可区分性,从两个不同的方向讨论了流本体类的分层表示:一是对构成流本体类的流片段的不同抽象度分层扩展;二是对构成流本体类的不可区分的流片段的合并分层扩展。给出了流本体类的分层过程,定义了系统级的映射关系,即流片段之间的抽象映射,指出该映射为基于系统中心本体的模型抽象分层过程提供了直接的转换运算。并且,流本体类的分层关系的确定也为基于系统中心本体的模型设计任务提供了可共享和重用的机制。

**关键词** 分层抽象模型,抽象度,对象本体类,流本体类,流本体类分层
**中图法分类号** TP391 **文献标识码** A

## Hierarchical Abstraction Model Based on System-centered Ontology

WANG Nan[1,3] SUN Shan-wu[2] OUYANG Dan-tong[3]
(Department of Management Science and Information Engineering, Jilin University of Finance and Economics, Changchun 130117, China)[1]
(Network and Laboratorial Center, Jilin University of Finance and Economics, Changchun 130117, China)[2]
(Key Laboratory of Symbolic Computation and Knowledge Engineering of MOE, College of Computer Science and Technology, Jilin University, Changchun 130012, China)[3]

**Abstract** The hierarchical ontology class makes it possible to automatically construct the hierarchical model of the physical world by using the abstract mapping relationship between ontology classes with different abstraction degrees. In this paper, we redefined Ontology Class as Object-based Ontology Class, and extended the definition of Flow Fragment to propose the system-centered concept of Flow-based Ontology Class. The behavior-indistinguishable flow fragments were defined. We discussed the hierarchical representation of the flow-based ontology class in two directions: the abstraction hierarchy of flow fragment and the aggregation of indistinguishable flow fragments. In the hierarchical structure, the abstraction mapping relationship between flow-based ontology classes was constructed to realize the hierarchical modeling process based on system-centered ontology. At the same time, the hierarchical relationship of flow-based ontology classes also provides a mechanism of system-centered knowledge sharing and reuse.

**Keywords** Hierarchical abstraction model, Abstraction degree, Object-based ontology class, Flow-based ontology class, Hierarchical structure of flow-based ontology class

## 1 引言

基于模型的推理(Model-Based Reasoning)中很重要的一步是对物理世界 W 进行抽象建模,很多学者研究了抽象理论[1-6]。KRA(Knowledge Reformulation and Abstraction)模型[7,8]将抽象的表示一般化,给出了对 W 进行形式化建模的一般性框架。广义 KRA 模型(G-KRA)[9,10]扩展了 KRA 模型框架,将感知分成基本感知和抽象感知,利用人工建立抽象对象库和抽象映射,实现 W 的基本感知中基本部件到抽象部件的映射,从而生成抽象感知。我们在 G-KRA 模型框架中引入本体概念,将抽象对象库扩展为本体类,在本体类中实现 KRA 模型框架中的感知层、语言层和理论层的知识共享和重用,简化了 KRA 模型的表示[11,12]。同时,提出基于流片段的抽象模型,丰富了本体类型[13],为多重模型构建提供了前提。

到稿日期:2010-09-02 返修日期:2011-01-27 本文受国家自然科学基金重大项目基金(60496320,60496321),国家自然科学基金(60973089,60773097,60873148),吉林省科技发展计划项目基金(20060532,20080107),欧盟合作项目(155776-EM-1-2009-1-IT-ERAMUNDUS-ECW-L12),符号计算与知识工程教育部重点实验室开放项目(93K-17-2009-K05),吉林省科技发展计划项目(20100173),吉林省教育厅"十一五"科学技术研究项目(2009512,2010392)资助。

王 楠(1980—),女,博士生,主要研究方向为基于模型的诊断等,E-mail:ctuwangnan@126.com;孙善武(1969—),男,副教授,主要研究方向为计算机网络;欧阳丹彤(1968—),女,博士生导师,主要研究方向为基于模型的诊断等。

本文将本体类的概念扩展为对象本体类和流本体类，并给出了流本体类的分层过程表示，实现了流本体类的自动创建、更新、共享和重用，为基于系统中心本体的模型构建和设计任务提供了可共享和重用的机制。

本文第2节扩展流片段的概念，并根据本体类型的不同将本体类定义为对象本体类和流本体类；第3节从两个不同的方向给出了流本体的抽象分层过程，并定义了不同层次流本体之间的映射关系，该映射关系为基于系统中心本体的模型抽象分层过程提供了直接的转换运算；最后给出总结与展望。

## 2 流本体类

在引入流本体类前，我们首先对文献[11,13]中的相关概念进行扩展。

**定义 1**(对象本体类，Object-based Ontology Class) 对象本体类 OOC 是一个四元组，即 $OOC=(E,D,L,T)$，其中 $E=(OBJ,ATT,FUNC,REL,OBS)$，表示客观世界的构成实体集合，其中各部分的定义引用 KRA 模型中感知 P 的构成要素 OBJ、ATT、FUNC、REL 的定义。特别地，OBS 中的观测不表示某个特定感知世界中的实体信息，而是在对各个不同世界的感知过程中不断扩展的实体信息集合。$D$ 对应于 G-KRA 模型框架中的结构 $S$($D$ 表示 datadabase，体现其共享性和重用性)，当中存储不断更新的构成客观世界的各种独立的实体信息。$L$ 和 $T$ 对应于 KRA 抽象模型中的语言层 $L$ 和理论层 $T$ 的定义，详见文献[7,8]。

为了从不同的本体角度感知客观世界，文献[13]定义了流片段的概念，并将对应的对象本体类中的对象分成 Active 类型和 Passive 类型。其中 Active 类型的对象具有引起某种现象发生的能力，而 Passive 类型的对象只能保证或阻止某种现象的出现。本文将流片段的概念进行扩展如下。

**定义 2**(流片段，Flow Fragment) 流片段是广义流从起始对象到终止对象的一条有向路径。一个流片段定义为一个三元组 $FF=\langle OBJMappings, RELMappings, \{PHObjMapping_i \rightarrow Behaves_i\}\rangle$，并且满足：

(1)OBJMappings 是一个有限的映射集合，其中的每一个元素都表示到某一对象本体类 OOC 中的一个对象的映射，并且 OBJMappings 中至少包含一个 Active 类型的对象，设 $|OBJMappings|=n$，流起始对象为 Os，流终止对象为 Oe；

(2)RELMappings 是一个有限的映射集合，其中每一个元素是到 OOC 中的某两个对象间的关系的映射；

(3)$\{PHObjMapping_i \rightarrow Behaves_i\}$是体现流片段功能现象的对象及其行为的集合。其中 $PHObjMapping_i$ 是流片段的构成对象集合 OBJMappings 中的一个对象，$Behaves_i$ 表示该对象在流片段 FF 中的行为集合，显示了流片段 FF 的功能现象。

作为一个流片段，OBJMappings 中必须包含一个到 Active 类型对象的映射，并且提供一条广义流可以流动的路径。这一点保证了流片段作为一个独立的子系统在满足一定条件的前提下，完成某些功能，达到某种目标。另外，广义流应该终止于流片段中的某个对象，这点保证了流片段的闭合性。因此，OBJMappings 中的流起始对象和终止对象应该做相应的标识。

在一定条件下，流片段可以通过 PHObjMappings 中的对象显示出某种或某些功能现象，$\{PHObjMapping_i \rightarrow Behaves_i\}$集合可能为空，此时表示该流片段没有任何可以观测的对象提供外界观测数据。

流片段之间的关系可以通过流片段中相互关联的对象来构建，定义相互关联的对象之间的定量或定性关系以及特定的激活条件，超出了本文研究的范围，这里引用的流片段之间的关系完全是一种广义的说明。当感知具体的客观世界或者人工系统时，需要对这些关系进行相应的实例化，并将其转化成对应的具体的关系表示。因此，本文不对流片段之间的关系进行分类，而是通过识别抽象模型中所有相互关联的对象来形式化地表示流片段间的关系集合(详见文献[13])。

对象本体类的概念构造了构成抽象模型的对象以及对象之间关系的集合，而由流片段感知过程则可生成以流片段为中心的流本体类。

**定义 3**(流本体类，Flow-Based Ontology Class) 流本体类定义为一个二元组 $FBOC=\langle FFs, FFRELs\rangle$。其中 FFs 表示流片段集合，FFRELs 表示流片段之间的关系集合。

流本体类的构建过程类似于对象本体类，在感知不同的客观世界的过程中不断地扩展生成。

## 3 流本体类的抽象分层过程

流片段的构成对象类型由感知的客观世界确定。由对象本体类可知，各种抽象算子的作用生成不同抽象度的对象本体类，可以定义类似的抽象算子应用于流片段，得到各种不同抽象度的流片段，构成不同的流本体类，这些不同的流本体类通过其中流片段之间的映射关系相关联。同时，不同的流片段的现象对象可能具备相同的行为模式，这种行为上不可区分的流片段可进行合并运算。通过这些独立于建模过程的运算，来实现流本体类的分层表示。

流本体类的分层表示可以从两个不同的方向讨论：一是对构成流本体类的流片段的不同抽象度分层扩展；二是对构成流本体类的不可区分的流片段的合并分层扩展。

### 3.1 流片段的抽象分层

构成流片段的对象是到某个对象本体类中相应对象的映射，而对象本体类的分层结构中定义了不同抽象度的对象本体类之间的映射关系。如果将一个流片段看作一个感知世界，则对流片段的分层运算类似于文献[11]中基于对象本体类的分层建模过程，不同的是各个流片段之间存在着关联关系，这种关联关系在流片段分层过程以及建立流本体类之间的关系时需要特殊处理。

**定义 4**(基本流片段，Fundamental Flow Fragment) 由基本对象本体类中的对象构成的流片段称为基本流片段。基本流片段通过直接感知客观世界的基本模型得到，获得基本流片段的过程称为基本流感知过程，记为 $\mathcal{FP}$。

**定义 5**(抽象流片段，Abstraction Flow Fragment) 由抽象对象本体类中的对象构成的流片段称为抽象流片段。抽象流片段可以通过对客观世界的抽象模型进行抽象流感知得到(感知过程记为 $\mathcal{AFP}$)，也可以通过运用对象本体类中定义的本体抽象算子运算得到，或者通过基本对象本体类与抽象对

象本体类之间的映射关系转换得到。

流片段的抽象度定义为该流片段对应的对象本体类的抽象度。

流感知过程 $\mathcal{FP}$ 的形式化描述详见文献[13]。

除了利用对不同抽象度的客观世界进行流感知以获得不同抽象度的流片段之外，还可以通过不同抽象度的对象本体类之间定义的抽象算子和映射关系对流本体类中流片段进行分层运算，如图 1 中第①部分所示，其中 $OP_i=\{\Psi_i,\Omega_i,\Phi_i,EM_i,CM_i\}$。各个运算算子的含义见文献[11]。

流本体类是流片段及流片段之间关系的集合。构成流本体类的流片段可能由多个同类世界的流感知获得(这里的同类世界指的是抽象化以后可以运用相同的推理规则进行推理的客观世界)，因此各个流片段的组成对象映射于同一对象本体类。

### 3.2 流片段的合并

对构成流本体类的流片段进行不同抽象度的分层扩展得到流本体类基于流片段的分层表示。可以看出各层流本体类中的流片段数量在分层过程中没有变化，只是通过改变各个流片段自身的抽象度来达到流本体类的逐层抽象目的。而从功能角度来看，处于活动状态的流片段可以实现某些特定的功能，这种功能通过现象对象的行为进行最原始的描述，因此这里从流片段这一层面进行讨论，使得能够对某些流片段进行自动的合并运算，改变流本体类本身的抽象度。

**定义 6**（流片段的行为不可区分性） 两个流片段 $FF_1$ 和 $FF_2$ 是行为不可区分的，当且仅当 $FF_1$ 和 $FF_2$ 中包含的现象对象类型和行为是一致的。

对流本体类中流片段进行合并运算改变了流本体类中流片段的构成，这里用流本体类的抽象度 $\gamma^*$ 形式化地表示两个流本体类的流片段合并运算方向。

设两个流本体类 $FBOC_1$ 和 $FBOC_2$，其抽象度分别为 $\gamma_1^*$ 和 $\gamma_2^*$，流片段的合并运算用 $Cff$ 表示，且 $FBOC_2=Cff(FBOC_1)$，则 $\gamma_1^*\leqslant\gamma_2^*$，即 $FBOC_2$ 可能比 $FBOC_1$ 包含更少的流片段。

行为不可区分的流片段的合并可以减少流片段的抽象运算次数。两种运算交替使用可以得到流本体类的分层过程，如图 1 所示。

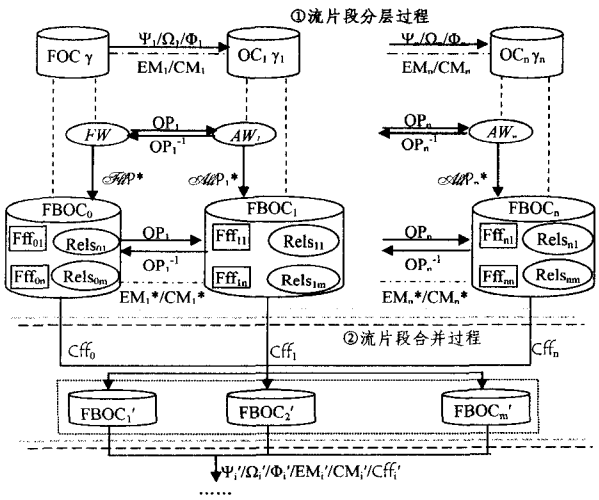



图 1 流本体类的分层过程

在图 1 中，流本体类 $FBOC_0,FBOC_1,\cdots,FBOC_n$ 是流本体类的基于流片段的分层关系，其构成对象是到相应的对象本体类的映射。而流本体类 $FBOC_1',FBOC_2',\cdots,FBOC_m'$ 则是对 $FBOC_0,FBOC_1,\cdots,FBOC_n$ 进行流片段合并得到的新的流本体类，其相对抽象度可能发生变化。由图中可以看出，基于流片段分层过程中的多个流本体类经过流片段的合并运算可能生成同一个流本体类。图 1 所示的流本体类的分层关系从不同的方向不断进行下去，直到流片段的抽象运算终止或者流本体类中的流片段数目不再发生变化。

在 G-KRA 模型框架下，通过对基于对象本体类的基本模型(初步感知阶段获得)或抽象模型(抽象感知阶段获得)进行流感知，创建或扩展相应的流本体类，并生成流本体类的分层表示，再根据各层流本体类之间的运算关系反作用于被感知模型，生成对应的基于流本体类的分层模型表示。当然，流本体类的抽象分层过程可以独立于基于系统中心本体的分层模型构建过程。随着待感知模型的不断输入，流本体类中的流片段类型也将不断丰富，最终为客观世界的模型设计任务提供知识共享和重用。

流本体类中的流片段类型随着待表示世界的不断输入而不断丰富，而流本体类的分层结构表示了系统级的映射关系，即流片段之间的抽象映射，从而为基于系统中心本体的模型抽象分层过程提供了直接的转换运算。同时，流本体类的分层关系的确定，也为基于系统中心本体的模型设计任务提供了可共享和重用的机制。

**结束语** 本文根据本体类型的不同将已有的本体类概念扩展为对象本体类和流本体类，从系统中心本体的角度给出了流本体类的分层过程表示，实现了流本体类的自动创建、更新、共享和重用，为基于系统中心本体的模型构建和设计任务提供了可共享和重用的机制。在下一步的工作中，我们将从流片段的现象对象的行为自动抽取和定性描述入手，继续深入探讨行为不可区分的流片段的自动合并运算的形式化问题。

## 参 考 文 献


[1] Sacerdoti E. Planning in a hierarchy of abstraction spaces[J]. Artificial Intelligence, 1974, 5: 115-135

[2] Lowry M. The abstraction/implementation model of problem reformulation[C]// Int. Joint Conf. on Artificial Intelligence. Milano, Italy, 1987: 1004-1010

[3] lBredeche N, Shi Z, Zucker J-D. Perceptual learning and abstraction in machine earning: an application to autonomous robotics [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2006, 36(2): 172-181

[4] Mozetic I. Hierarchical model-based diagnosis[J]. Int. Journal of Man-Machine Studies, 1991, 35(3): 329-362

[5] Chittaro L, Ranon R. Hierarchical model-based diagnosis based on structural abstraction[J]. Art. Intell., 2004, 155(1/2): 147-182

[6] 欧阳丹彤，欧阳继红，程晓春，等. 分层的基于模型诊断方法[J]. 计算机科学，2004，31(10A)：254-277

[7] Saitta L, Zucker J. Semantic Abstraction for Concept Representation and Learning[C]// Proc. SARA. 1998: 103-120

[8] Saitta L, Zucker J-D. A Model of Abstraction in Visual Perception[J]. Applied Artificial Intelligence, 2001, 15(8): 761-776

[9] 孙善武，王楠，欧阳丹彤. 广义 KRA 抽象模型[J]. 吉林大学学报：理学版，2009，47，(3)：537-542

[10] Wang Nan, OuYang Dan-tong, Sun Shan-wu, et al. Formalizing the Modeling Process of Physical Systems in MBD[C]// The 2009 International Conference on Web Information Systems and Mining (WISM'09) and the 2009 International Conference on Artificial Intelligence and Computational Intelligence (AICI'09). 2009: 685-695

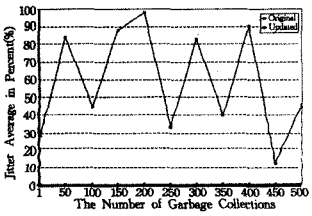
[11] 王楠，欧阳丹彤，孙善武. 基于本体的分层抽象模型[J]. 计算机科学(已录用)

[12] Wang Nan, OuYang Dan-tong, Sun Shan-wu. Formalizing Ontology-based Hierarchical Modeling Process of Physical World[C]// The 2010 International Conference on Web Information Systems and Mining (WISM'10) and the 2010 International Conference on Artificial Intelligence and Computational Intelligence (AICI'10). 2010: 18-24

[13] Sun Shan-wu, Wang Nan. Formalizing the Multiple Abstraction Process Within the G-KRA Model Framework[C]//2010 International Conference on Intelligent Computing and Integrated Systems(ICISS2010). 2010: 281-284

(上接第 149 页)

式中，$\bar{x}=\frac{1}{N}\sum_{i=1}^{N}x_i$。



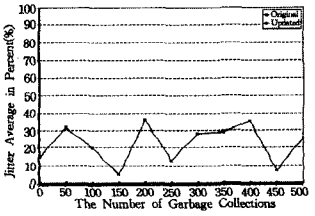




图 9 响应时间的的标准差比较 图 10 响应时间的最大离均差比较

由实验结果可以看出，原始的 OSGi 事件响应时间具有较大的不确定性，修改后的 OSGi 框架在 120 个实验数据中，标准差与平均值的比率最大值为 0.03%，考虑到线程切换的时间开销，我们认为修改后的 OSGi 框架的响应时间基本无抖动，具有确定性。由此得：基于实时事件机制的 OSGi 框架可以满足嵌入式实时系统的实时性要求。

**结束语** 本文通过分析 RTSJ 对 OSGi 框架的影响，并针对 OSGi 事件派发线程的实时性无法保证的问题，提出了基于实时线程的事件机制，解决了 OSGi 为了自适应环境变化所导致的实时服务之间无法切换的问题，从而保证了整个实时嵌入式系统的实时性。

我们的研究目标是提出一个基于 OSGi 的自适应实时中间件，而将 RTSJ 整合到 OSGi 中的工作目前只完成了第一步。下一步还将对以下两个方面进行深入的研究：(1) 基于 RTSJ 的 OSGi 各实时组件间的运行期隔离；(2) 基于实时 OSGi 的端到端的自适应 QoS 协商机制。

## 参 考 文 献

[1] Erl T. Service-Oriented Architecture: Concepts, Technology, and Design [M]. Prentice Hall, 2005

[2] OSGi Service Platform Core Specification v4. 0 [EB/OL]. http://www. osgi. org, 2010-4-2

[3] Belliardi R, Brosgol B, Dibble P, et al. The real-time specification for Java-version 1. 0. 2 [M]. USA: Addison-Wesley, 2004

[4] Dvorak D, Bollella G, Canham T, et al. Project Golden Gate: Towards Real-Time Java in Space Missions [C]// Proceeding of the Seventh IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. Vienna: IEEE Press, 2004: 15-22

[5] Etienne J, Cordry J, Bouzefrane S. Applying the CBSE Paradigm in the Real-time Speci_cation for Java [C]// Proceedings of the 4th interna-tional workshop on Java technologies for real-time and embedded systems. USA: ACM Press, 2006: 218-226

[6] Hu J, Gorappa S, Colmenares J A, et al. Compadres: A Lightweight Component Middleware Framework for Composing Distributed, Real-Time, Embedded Systems with Real-Time Java [C]// Proceeding of ACM/IFIP/USENIX 8th Int'l Middleware Conference (Middleware 2007). vol. 4834. Newport Beach: Springer, 2007: 41-59

[7] Plšek A, Loiret F, Merle P, et al. A Component Framework for Java-based Real-Time Embedded Systems [C]// Proceeding of ACM/IFIP/USENIX 9th International Middleware Conference. Leuven: Springer, 2008: 124-143

[8] Hong W E, Ku B J, Lee M J, et al. Combined Approach of OSGi and RTLinux Framework for Supporting Software Architecture of Internet Embedded Real-Time System [C]// Proceeding of IEEE Real-Time and Embedded Technology and Applications Symposium. Toronto: IEEE Press, 2004: 296-305

[9] Gui N, Florio V D, Sun H, et al. A Hybrid real-time component model for reconfigurable embedded systems [C]// Proceedings of the 2008 ACM symposium on Applied computing. Fortaleza: ACM Press, 2008: 1590-1596

[10] Tia T, Liu J W, Shankar M. Aperiodic request scheduling in fixed-priority preemptive systems [R]. UIUCDCS-R-94-1859. Dept. Computer Science, University of Illinois at Urbana-Champaign, 1994

[11] Goodenough J B, Sha L. The priority ceiling protocol: A method for minimizing the blocking of high priority Ada tasks [C]// Proceedings of the second international workshop on Real-time Ada. vol. 7. Devon: ACM Press, 1988: 20-31

[12] Sha L, Rajkumar R, Lehoczky J P. Priority Inheritance Protocols: An Approach to Real-Time Synchronization [J]. IEEE Transactions on Computers, 1990, 39(9): 1175-1185

[13] Tsigas P, Zhang Y, Cederman D, et al. Wait-Free Queue Algorithms for the Real-time Java Specification [C]// Proceeding of the12th IEEE Real-Time and Embedded Technology and Applications Symposium. San Jose: IEEE Press, 2006: 373-383

[14] Apache Felix [EB/OL]. http://felix. apache. org/site/index. html, 2010-3-2