面向agent的软件开发方法*)

毛新军[1] 闫 琪[1] 王怀民[1] 楚蓓蓓[2]
(国防科学技术大学计算机系 长沙410073)[1] (信息工程学院计算机系 郑州)[2]

Agent Oriented Software Development Method

MAO Xin-Jun[1] YAN Qi[1] WANG Huai-Min[1] CHU Bei-Bei[2]
(Department of Computer, National University of Defence Technology, Changsha 410073)[1]
(Department of Computer, Information Technology Institute, ZhengZhou)[2]
E-mail: xjmao21@21cn.com

Abstract Agent oriented software development method is a new research area in the field of software engineering. As the method provides abstract concept, i.e., agent, to naturally model the computational entities and problem solving manners in complex system, and ways to effectively control the system complexity, many attentions have been imposed on it. The paper summarizes the research status, introduces and investigates a number of representative works, and at last concludes the existing problems and future further research work.

Keywords Agent, Agent oriented, Software development method

1. 引言

随着Internet和Intranet技术的迅猛发展、计算机应用的不断扩大和深入，当前软件系统广泛呈现出分布、自适应、动态可扩展、开放、异构、可成长等复杂性特征，如Internet环境下的信息服务系统、空中交通管制系统等。支持该系统建设的基础软件面临系统的动态可扩展性、自适应性、复杂交互合作、自我成长等一系列新的关键问题的挑战，其软件体系结构和开发方法，较传统软件都将发生深刻变化，从而对软件开发方法、技术、过程和工具等提出了新的要求。

agent是指具有自主性、交互性、反应性和自发性等特征、能持续发挥作用的计算实体。基于agent系统由具有一定资源和能力、相对独立且相互合作的一组agent组成。面向的agent软件开发技术代表了一种新的软件开发范型，它为开发具有上述特征的软件系统提供了新的思路。一方面，agent概念为全面准确地研究和分析上述软件系统的特点提供了合理的概念模型，agent概念的自主性、交互性、自发性等属性能够有效适应上述软件系统的动态可扩展、自适应、复杂交互合作、自我成长等特征；另一方面，基于agent系统提供了更高层次的抽象模型，能够自然、贴切、直观地表示复杂系统中的行为实体以及问题求解方式，提供了有效的手段(抽象、自然建模等)控制和管理软件复杂度，因而被认为是软件工程领域一次新的革命。

尽管agent技术得到了学术界和工业界的高度关注和重视，近年来研究取得了长足进展，但面向agent的软件开发范型并没有人们所预期和期待的那样广泛应用于工业应用，其中原因之一就是基于agent系统的开发缺乏系统的、面向agent软件开发方法的指导。要真正让agent技术在解决当前软件开发问题以及工业应用中发挥作用，充分发挥其潜力，真正有效促进复杂系统的开发，还需要与此相对应的软件开发方法学，来指导软件开发人员系统地对基于agent系统进行工程化开发，支持需求分析、软件设计和编码等软件开发活动。由于基础元概念模型的差异，现有许多软件开发方法(如结构化、面向对象等)不能为基于agent系统开发提供方法学上的指导，因此必须研究面向agent的软件开发方法。

自20世纪90年代中后期以来，学术界和工业界开始关注面向agent的软件开发方法的研究和应用，目前人们已经提出了一些面向agent的软件开发方法，其中较有影响的工作有：Wooldridge, Jennings和Kinney的Gaia方法[1,4]，Wood和Deloach的MaSE方法[2]，Yim, Odell, Parunak, Bauer等的agent UML方法[2]等等。

2. 面向agent的软件开发方法

基于agent系统的开发是一项极为复杂的工作，它不仅需要agent体系结构、通信语言、合作模型、本体论、程序设计语言等具体的技术、模型和实现工具，更需要系统方法学的支撑以对基于agent系统进行工程化的软件开发。

面向agent的软件开发方法是指支持基于agent系统开发的方法学，它提供了良定义和结构化的过程以及相应的描述语言对基于agent系统进行描述，支持软件生命周期中的需求分析、设计和实现等阶段的工作。对基于agent系统进行描述的本质实际上就是在一定抽象层次上对基于agent系统进行建模。遵循软件工程的抽象、逐步求精、多视点的原则，面向agent的软件开发方法不仅支持从不同抽象层次、不同的视点对基于agent系统进行建模，而且支持从高层次模型(问题描述)到低层次模型(软件解决方案)的逐步过渡和精炼，最终得到可实现的系统模型。

不同于结构化软件开发方法和面向对象的软件开发方法，面向agent的软件开发方法基于agent元概念模型，其核心思想是：系统是由一组具有自主性、反应性、自发性和交互

*)本文得到国家自然科学基金60003002，973项目G1999032700的支持。毛新军 博士，副教授，研究方向：agent理论和技术，新型软件开发方法学等。

性等特征的 agent 组成，agent 之间通过社会性的行为来实现系统的整体功能和目标。agent 间的社会性行为不同于对象技术中的消息传递，而是基于某种 agent 通讯语言的 agent 交互以及在此基础上的复杂协同与合作。

目前人们已经提出了许多专用和通用的面向 agent 软件开发方法。根据这些方法所依赖的技术背景上的差异，我们将现有的面向 agent 软件开发方法分为以下几个类别。

表1 面向 agent 软件开发方法分类表

	特点	代表性工作
基于角色模型	借助于社会学和组织学等学科方面的知识，基于角色模型	Gaia，AOM
基于对象技术	借助于面向对象软件开发方法学的知识	MaSE，UML，AUML
基于知识工程	借助于知识工程的知识	DESIRE，ADEPT

(1)基于角色模型 该类别的方法借助于社会学和组织学等学科方面的知识，通过角色概念来理解系统中的行为参与者，将 agent 视为承担某种角色的自主行为实体。一个 agent 可以承担一个或者多个角色，一个角色也可以为多个 agent 所承担。代表性工作包括：Wooldridge 等人提出的 Gaia 方法[1,4]。

(2)基于对象技术 该类别的方法借助于面向对象软件开发方法学的知识，将 agent 视为具有并发和自主特征的特殊对象，通过对已有面向对象软件开发方法的扩充(尤其是 UML：Unified Modeling Language)来支持对基于 agent 系统的建模。代表性工作包括：Odell，Yim 等人提出的 AUML (Agent UML)，Deloach 等人提出的 MaSE(Multiagent System Engineering Methodology)方法。

(3)基于知识工程 该类别的方法借助于知识工程对基于 agent 系统进行建模，代表性工作包括 DESIRE(Design and Specification of Interacting Reasoning)，ADEPT (Advanced Decision Environment for Process Task)。

下面我们系统地介绍和评估各类别中代表性的面向 agent 软件开发方法。

3. 代表性工作

3.1 Gaia

Gaia 是由 Wooldridge，Jennings 和 Kinny 提出的一个通用的面向 agent 软件开发方法[4]，该方法支持对基于 agent 系统的微观(agent 体系结构)和宏观(agent 社会和组织结构，agent 间的合作)建模，Gaia 将基于 agent 系统视为一个组织或社会，支持自顶向下、逐步求精的软件开发，最终得到一个易于实现、满足用户需求并且独立于具体实现技术和工具的系统模型(包括用户需求模型和软件模型)。

Gaia 支持基于 agent 系统的分析和设计阶段的工作。在分析阶段，分析人员获取系统中的角色，描述各个角色的特征，并建立角色间的交互模型。系统中每个角色的特征由以下四个属性来表示：职责、权限、活动和协议。职责描述了角色在生命周期中所具有的功能，包括确保某些事情得以发生(活性职责)以及确保某些事情不能发生(安全性职责)，例如一个拍卖 agent 的职责是通过和有关 agent 的协商尽可能达成交易，同时防止交易价低于拍卖底线。权限是指 agent 在履行其职责过程中应具有的资源，对软件 agent 而言主要是指它能够存取的信息。活动是指 agent 无需和其它 agent 交互而完成的任务。协议是指角色间特殊的交互模式。交互模型定义了角色间的相互关系，例如交互、依赖、组织结构等。Gaia 提供了用于描述角色和交互的模版和模式，以更为简便、有效地支持对基于 agent 系统的分析。

在设计阶段，设计人员首先将分析阶段角色模型中的各个角色映射为 agent 类型，创建适当数目的 agent 实例，然后建立 agent 的服务模型以描述 agent 如何实现其承担的角色，最后建立系统的熟人模型表示 agent 之间的通讯。

Wooldridge 等人提出 Gaia 的初衷是希望它能够有效表示 agent 的灵活性、自主进行问题求解的能力、多 agent 系统的复杂组织结构等。该方法要求系统的组织结构和 agent 的能力在运行时是静态的，并且在设计阶段就可确定，系统中的各个 agent 通过合作来实现系统全局的目标。这就决定了该方法适合于封闭系统的开发，而不能支持开放、不可预测系统的开发。由于这一局限性，Zamboneli，Jennings 等人对 Gaia 方法进行了扩充，以支持具有开放特征的 Internet 应用系统的开发[1]。此外，该方法对系统行为建模方面的能力是比较弱的。

3.2 MaSE

MaSE 由 Wood 和 Deloach 提出[3,7]。在知识和技术背景、适用范围等方面，MaSE 和 Gaia 较类似。但 MaSE 充分借鉴了面向对象软件开发方法(尤其是 UML)的经验和知识，支持实现阶段的工作，要求 agent 之间的交互是一对一的，并且提供了更多的视图和更为详细的过程对基于 agent 系统进行建模。

MaSE 方法共分7个相互关联、逐步细化的步骤。首先，根据系统的初始描述获取系统的目标，并且根据目标的重要性及其关系将它们组织成结构化的层次结构；第二步，根据初始描述，创建系统用况图和顺序图，描述系统中的角色以及角色之间的交互，以支持后续阶段的系统角色模型和任务的定义；第三步，建立系统的角色模型，并将第一阶段所获取的系统目标合理地分配到恰当的角色上，定义角色的任务模型以实现其拥有的目标；第四步，根据角色模型创建系统的 agent 类模型，描述系统中的角色及其组织结构和对话关系；第五步，定义 agent 之间的交互模型，详细定义 agent 间的对话内容；第六步，组装 agent，定义 agent 实现的体系结构，系统中的 agent 可以采用五种不同的体系结构进行组装包括 BDI，反应式，规划，知识和自定义；第七步，根据 agent 类创建实际的 agent 实例。其中，前三步用于定义系统的概念模型，因而属于需求分析阶段的工作，后四步用于定义系统实现模型，因而属于设计阶段的工作。此外通过 MaSE 工具，MaSE 支持将设计结果直接转换为相应的代码，因而 MaSE 也支持实现阶段的工作。

Wood 和 Deloach 提出 MaSE 方法的目的是希望能够得到一个实用的方法以及相应的工具系统地支持基于 agent 系统的开发。为此，MaSE 充分利用了在工业界得到实践检验、成熟的技术和成功的经验，尤其是基于 UML 的面向对象软件开发方法。但该方法仍然局限于仅支持对封闭式系统的开发，而不能适应开发、动态系统的建模。

3.3 UML 和 AUML

学术界和工业界的许多学者认为 agent 技术要真正走向应用，广泛支持工业界软件的工程化开发还必须借助于工业界现有的成熟技术和标准。对象技术是当前主流的软件工程

技术，经过20多年的实践其软件工程化的能力已得到了充分的检验。UML是一个对系统进行对象建模的可视化建模语言，它具有多视点、可视化、可扩展等特点，并于1997年被OMG批准为标准，成为工业界标准的对象建模语言。

AUML是指利用UML或者通过对UML的扩展，支持利用agent核心概念和有关机制对基于agent系统进行建模，因此严格地讲AUML是一个agent统一建模语言，而不是面向agent的软件开发方法，因为它仅仅提供了对基于agent系统进行建模的语言机制，而没有提供相应的过程和步骤来指导如何一步步地对基于agent系统进行建模。

Yim提出了一个以体系结构为核心的多agent系统设计方法[8]，该方法基于标准的UML，利用对象约束语言(Object Constraint Language)将面向agent的建模问题转化为面向对象的建模问题，将agent之间的关系变换为一组设计模式，从而支持开发人员直接利用UML以及相应的工具对基于agent系统进行建模。Bergrnti和Poggi提出用四个不同抽象层次的图形对基于agent系统进行建模，第一层是本体论图，利用UML中的类图描述系统中不同概念、术语和实体之间的关系，第二层体系结构图，用UML的实施图描述多agent系统的配置，第三层是协议图，利用UML的合作图描述agent之间的交互协议，第四层是角色图，利用UML的类图描述各个agent所承担的角色和具有的功能。这二种方法直接可利用UML对基于agent系统进行建模，无需对UML进行调整和扩充。

Parunak和Odell将现有的、用于描述agent结构的组织模型引入UML，通过对UML的扩充，支持对基于agent系统的社会组织结构进行建模。Odell, Parunak和Bauer通过对UML进行扩充，提出了描述agent交互协议的三层表示方法，提供了一组模版用于描述agent间的交互协议(包括消息通讯和通讯内容)。Odell进一步建议扩展UML，提供agent的统一建模语言AUML，以对基于agent系统的所有相关内容进行建模。根据建议，AUML应该拥有更加丰富的角色表示机制，因而需要修改和扩展交互图，为了描述agent的自主性以及agent之间的交互合作，需要修改和扩展包图，此外为了描述agent的移动(Mobile)特征，需要对UML的实施图进行调整。目前AUML已经提交给UML标准化组织，计划在UML 2.0中融入有关对基于agent系统进行建模的成分。

利用UML和AUML对基于agent系统进行建模是可行的，一方面，我们可以在一定程度上将agent视为是一种特殊的对象，从而利用对象技术对基于agent系统进行建模，另一方面，对象技术尤其是UML是经过实践检验能够有效用于工业化软件生产的成熟和标准化的技术，熟悉该技术的开发人员无需专门深入学习agent的有关知识就可对基于agent系统进行建模，因而可以很快在基于agent系统的开发中发挥作用。利用UML或者AUML可以支持基于agent系统开发过程中分析、设计和实现阶段的工作。然而，无论是UML还是AUML它们都是基于"对象"这个一阶概念对基于agent系统进行建模，因而不能充分利用agent元概念模型的灵活性和高度抽象能力。

3.4 DESIRE

DESIRE是支持基于agent系统设计和实现的环境以及基于该环境的软件开发方法。该方法借助于传统计算机科学技术中的许多知识，例如人工智能、知识工程、时序逻辑等，对基于agent系统进行建模和实现，它主要支持设计和实现阶段的软件开发工作。DESIRE通过以下四个模型对基于agent系统进行建模。(1)任务模型，描述系统的任务层次，每个任务的输入和输出，以及任务之间的相互关系；(2)信息交换模型，描述任务之间的信息交换；(3)任务执行顺序模型，描述任务执行的控制结构；(4)知识结构，利用时序逻辑描述描述任务执行体(agent)所拥有的知识。

DESIRE方法依赖于DESIRE软件开发环境，该环境对基于agent系统的开发提供工具支撑。DESIRE方法适合于具有知识密集型和问题求解型特征的基于agent系统的开发，由于运用了人工智能、知识工程、时序逻辑等方面的知识，因而对开发人员的知识背景要求较高。

结论和展望 面向agent的软件开发方法是近年来软件工程领域出现的一个新兴的研究领域，它基于agent这一抽象概念模型，试图通过一组技术、过程和工具支持基于agent系统的软件开发，尤其是支持高层的分析和设计阶段的工作。经过近几年的发展，目前人们已经提出了近十种面向agent的软件开发方法。这些方法基于不同的知识和技术背景，采用了不同的建模语言和过程支持基于agent系统的开发，因而具有不同的适用范围。

然而现有研究存在诸多问题，主要体现在：(1)许多方法或者是针对特定系统和特定的agent体系结构，因而缺乏通用性；或者被定义为特殊的面向对象方法，因而不能充分利用agent元概念模型的灵活性和高度抽象能力；(2)现有方法假定构成系统的成分在设计阶段是预先可定义的，因而不支持具有开放、自我成长系统的开发，而今后越来越多的软件系统将具有开放、自我成长特征；(3)建模的表达能力不够，例如不能对系统的全局性行为约束进行描述，对系统动态行为建模的支持比较有限。

随着面向agent软件开发方法的不断发展、完善和逐步成熟，上述问题将会逐步得到解决，有关标准化问题将会引起人们(例如，OMG和FIPA)的关注并提上议事日程，制定统一的面向agent软件开发方法将是agent技术真正走向工业化应用的重要标志。

参考文献

1 Omicini A, Zamboneli F. Coordination for Internet Application Development. Autonomous Agents and Multi-agent Systems, 1999,2(3):251～269

2 Bauer B, Muller J, Odell J. Agent UML: A Formalism for Specifying Multi-agent Software Systems. International Journal of Software Engineering and Knowledge Engineering, 2001,11(3)

3 Deloach S A, Wood M F, Sparkman C H. Multiagents Systems Engineering. International Journal of Software Engineering and Knowledge Engineering, 2001,11(3)

4 Wooldridge M, Jennings N R, Kinny D. The Gaia Methodology for Agent-Oriented Analysis and Design. International Journal of Autonomous Agents and Multi-agent System, 2000,3

5 Jennings N R. On Agent-based Software Engineering. Artificial Intelligence, 2000,117(2)

6 Jennings N R, Wooldridge M. Agent Oriented Software Engineering. Handbook of Agent Technology, AAAI/MIT Press, 2000

7 Shehory O, Sturm A. Evaluating Agent-based System Modeling Techniques: [Technique Report TR-ISE/IE-003-2000]. Israel Institute of Technology, 2000

8 Yim H, Jongwoo K, Park S. Architecture-Centric Object Oriented Design Method for Multi-agent Systems. In: Proc. of 4th Intl. Conf. on Multiagent Systems, 2000

9 Odell J, Parunak H, Bauer B. Extending UML for Agent. In: Proc. of 17th National Conf. on Artificial Intelligence, 2000