УДК 681.3

О.В. Карсаев, В.Г. Конюший, В.В. Самойлов

ИНСТРУМЕНТАЛЬНАЯ СРЕДА И ЯЗЫК ДЛЯ РАЗРАБОТКИ ПРИКЛАДНЫХ МНОГОАГЕНТНЫХ СИСТЕМ

В работе приводятся описания инструментальной среды MASDK и языка проектирования ASML, предназначенных для разработки многоагентных систем. Среда поддерживает полный жизненный цикл разработки приложений, обеспечивает визуальное проектирование моделей многоагентных систем, и автоматическую генерацию программного кода агентов.

Многоагентные системы; платформа агентов; язык проектирования.

O.V. Karsaev, V.G. Konushiy, V.V. Samoylov

INSTRUMENTAL ENVIRONMENT AND LANGUAGE FOR DEVELOPMENT OF AGENT-BASED SYSTEMS

The paper presents development environment MASDK and modeling language ASML, intended for agent-based systems' development. The environment supports a complete life cycle of applications' development, provides for visual designing of multi-agent system models and automatic producing of agents' source code.

Multi agent systems; agent platform; modeling language.

Введение. Принципиально новые практические возможности прикладных многоагентных систем (МАС) по сравнению с традиционными программными системами основываются на том, что МАС позволяют реализовывать свойства поведения, которыми обладают реальные системы. Такими свойствами поведения являются *автономность, проактивность, децентрализация, открытость, социальность и взаимодействие*. Реализация таких свойств в программных системах также предполагает наличие определенной специфики, свойственной методологии разработки прикладных МАС. При этом полагается, что методология разработки [1] понимается в «широком смысле» и включает, в том числе такие аспекты, как язык описания таких систем и инструментальные средства поддержки процессов разработки.

В обобщенном виде архитектуру прикладных МАС можно представлять состоящей из трех взаимодействующих между собой подсистем (рис. 1). Подсистемы первого типа, «Внешнее окружение МАС», состоят из компонент, которые обобщенно называют «сенсорами» и «эффекторами» МАС. Их предназначением является описание данных предметного мира, являющегося внешним окружением МАС, и выполнение во внешнем окружении действий, являющихся результатами функционирования сообщества агентов. Компонентами данных подсистем могут являться интерфейсы пользователя, различного рода датчики, программируемые контроллеры и т.д. Подсистемами второго типа являются собственно сообщества агентов, которые формируются и обновляются в соответствии с текущей ситуацией, описываемой с помощью подсистем первого типа. При этом агенты в сообществе агентов «представляют интересы», сопоставляемых им сущностей внешнего окружения, и действуя от имени этих сущностей и взаимодействуя между собой, стремятся к достижению целей, стоящих перед системой в целом, или перед определенными сущностями предметного мира. Подсистемы третьего типа составляют компоненты, реализующие необходимую для функционирования сообщества агентов среду, называемую «платформой агентов». С учетом такого представления МАС можно говорить, что специфика их разработки главным образом связана с разработкой подсистем второго и третьего типа.

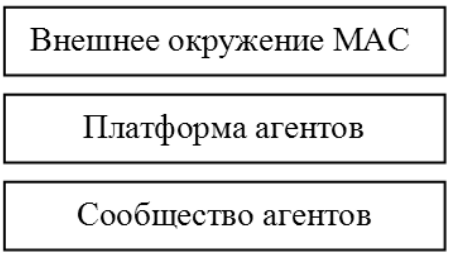

Рис. 1. Обобщенная архитектура МАС

Обоснование выбранного подхода для разработки прикладных МАС. Одним из наиболее эффективных подходов для поддержки процессов разработки программных систем является подход, в котором разделяются этапы разработки и описания модели прикладной системы и написания программного кода на основе разработанной модели системы. Такой подход позволяет существенно сокращать трудоемкость, стоимость и риски разработки программных систем. Он основывается на использовании специальных языков и средств для формального описания моделей программных систем, которые позволяют на втором этапе в значительной степени заменять написание программного кода его автоматической генерацией на основе формального описания модели системы. В связи с тем, что описание моделей систем при этом выполняется в виде различного рода диаграмм, такой подход также называют визуальным проектированием, и «программированием без программирования», имея в виду возможность автоматической генерации программного кода. В отношении объектно-ориентированного программирования такой подход начал активно развиваться, начиная с 80-х годов. Его основой является универсальный язык моделирования UML (Unified Modeling Language), который используется в различных инструментальных средах для описания моделей систем. Работы многочисленных исследователей в области развития агентно-ориентированного программирования и реализации аналогичного подхода показали необходимость разработки специфической модификации языка UML. В связи с этим разработка таких модификаций языка UML в последнее десятилетие стала объектом активных исследований.

Применительно к обобщенному представлению архитектуры прикладных МАС в виде трех подсистем такой язык непосредственно предназначается для разработки подсистем второго типа, т.е. для описания моделей поведения сообщества агентов. При этом полагается, что разработка подсистем первого типа может выполняться традиционным образом. Это, в частности, означает, что для описания их моделей можно использовать язык UML. Для разработки подсистем третьего типа, платформ агентов, как правило, предлагаются готовые программные компоненты.

Такой же подход, в котором для формирования платформы агентов предлагаются готовые решения, а разработка моделей поведения агентов выполняется с помощью визуального проектирования, в течение последних девяти лет развивается в СПИИРАН. В последующих разделах статьи приводится краткое описание возможностей разработанных к настоящему времени версий платформы агентов, языка описания моделей МАС ASML (Agent-based System Modeling Language) и основанной на этом языке инструментальной среды MASDK (Multi Agent System Development Kit), предназначенной для разработки прикладных МАС.

P2P платформа агентов. Для разработки подсистем третьего типа, т.е. для реализации платформы агентов предлагается использовать готовую программно-реализованную компоненту, «P2P Платформу агентов». Эта платформа по отношению к агентам обеспечивает выполнение следующих основных функций и сер-

висов. Она обеспечивает взаимодействие агентов с внешней средой, управляет жизненным циклом агентов, а именно – управляет созданием и запуском, остановом и удалением агентов, предоставляет агентам сервисы, обеспечивающие их взаимодействие, в частности, поиск требуемых для взаимодействия агентов и отправку и получение сообщений. Таким образом, платформа агентов обеспечивает на проблемно-независимом уровне возможность реализации специфических свойств поведения МАС, перечисленных во введении. В частности, открытость МАС проявляется в том, что в зависимости от складывающейся ситуации в окружающей среде платформа агентов обеспечивает создание новых агентов, предоставляющих интересы новых сущностей внешнего окружения, удаление старых агентов, а также возможность установления новых и удаление старых связей между агентами при изменении состава сообщества агентов. Свойство взаимодействия проявляется в том, что платформа предоставляет агентам возможность обмениваться сообщениями.

Следует отметить, что предлагаемая платформа агентов позволяет создавать «полностью» распределенные системы. В связи с этим название предлагаемой платформы включает определение «P2P» (peer-to-peer). Суть этой возможности в отличии от большинства других реализаций платформ агентов состоит в том, что при разработке распределенных систем среда функционирования агентов формируется на основе множества экземпляров платформы агентов, установленных на разных вычислительных устройствах. При этом регистрация агентов, необходимая для обеспечения сервисов поиска и взаимодействия агентов, выполняется локально, т.е., агенты регистрируются на той платформе, на которой они функционируют. Таким образом, в разрабатываемых приложениях могут полностью отсутствовать какие-либо компоненты, свойственные централизованным архитектурам.

Платформа агентов является проблемно независимой компонентной. В связи с этим она только предоставляет функциональные возможности для реализации тех свойств, которые отличают прикладные МАС от традиционных программных систем. На проблемно ориентированном уровне такие свойства должны описываться в подсистемах второго типа, т.е. в моделях поведения агентов, описание которых выполняется на языке ASML с помощью среды MASDK.

Язык и среда описания моделей агентов. Инструментальная среда MASDK обеспечивает визуальное проектирование моделей МАС. Используемый при этом язык описания моделей МАС является развитием одной из наиболее известных и популярных методологий разработки МАС, методологии GAIA [2]. Полное сопоставление языка ASML с методологией GAIA, с описанием реализованных в языке возможностей по развитию этой методологии, приводится в работе [3]. В частности, одна из таких возможностей состоит в следующем. Проектирование моделей прикладных систем выполняется как на архитектурном уровне, так и на детальном уровне. На архитектурном уровне в модели описываются классы агентов, их роли и сценарии поведения классов агентов для выполнения этих ролей, схемы и протоколы взаимодействия классов агентов, а также онтология предметной области. На детальном уровне выполняется спецификация классов агентов и сценариев их поведения в терминах модельных параметров и переменных.

Архитектуру агентов, описываемых с помощью языка ASML, можно представить с помощью рис. 2. Поведение агентов описывается в терминах ролей. Агент каждого класса может исполнять одну или несколько ролей. Описание модели классов агентов включает описание правил и ситуаций, при срабатывании/возникновении которых инициируется выполнение соответствующих ролей классов агентов. Для каждой роли в модели класса агента описывается соответствующий сценарий поведения. Описание сценариев поведения состоит в описании совокупности действий агентов и порядка их выполнения. Агенты одного класса различаются тем, что они имеют свои персональные данные.

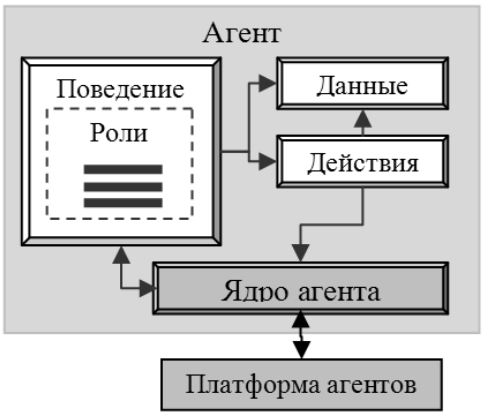

Рис. 2. Абстрактная архитектура агента

На основе модели МАС, описываемой в языке ASML, автоматически генерируется программный код, описывающий проблемно-ориентированное поведение классов агентов МАС. Генерируемый код интегрируется с готовой программной компонентой, «ядром агента». Эта компонента является инвариантной по отношению к проблемно-ориентированной модели поведения класса агентов. Она обеспечивает взаимодействие агента с платформой, и реализует базовый механизм (который иначе называется «движком»), который управляет исполнением описываемых в модели сценариев поведения агента.

Сценарии поведения классов агентов состоят из действий, которые в модели МАС описываются на вербальном (концептуальном) уровне и специфицируются на этапе детального проектирования в терминах модельных переменных и параметров. В связи с этим сгенерированные библиотеки программных классов, описывающих действия классов агентов, содержат только наименования методов, соответствующих действиям, и объявления параметров и переменных. Процедуры (тела методов) разрабатываются обычным образом. Такой подход, с одной стороны, позволяет гибко развивать автоматически сгенерированный программный код классов агентов за счет разработки или использования (подключения) произвольных, сколь угодно сложных процедур и/или программных компонент, реализующих действия классов агентов. С другой стороны, он позволяет сохранять полное соответствие модели МАС и программного кода классов агентов в процессе итеративной разработки прикладных систем, в течение которой как модель МАС, так и дополнительно разрабатываемый программный код могут быть подвержены многократной модификации.

Результатом разработки прикладной МАС являются библиотеки, реализующие описание поведения и взаимодействия классов агентов. Они используются платформой агентов в процессе функционирования МАС для создания необходимых экземпляров агентов.

Главное отличие языка ASML, обеспечивающее по сравнению с языком UML более подходящие выразительные возможности для описания моделей МАС, состоит в следующем. Язык ASML позволяет описывать динамические аспекты поведения класса агентов. В частности, разработанный язык позволяет описывать схемы и протоколы взаимодействия классов агентов. Протоколы взаимодействия «являются связанными» с определенными сценариями поведения классов агентов. Связь протоколов со сценариями поведения состоит в том, что сценарии поведения предусматривают определенный набор действий, выполнение которых влечет

посылку сообщений, переход в режим ожидания для получения ответных сообщений, а также действия, связанные с обработкой полученных сообщений. При этом язык ASML при описании моделей МАС обеспечивает полное соответствие описания сценариев поведения агентов с описанием связанных с ними протоколами взаимодействия.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Bergenti F., Gleizes M.P., Zambonelli F.* Methodologies and Software Engineering for Agent Systems, Kluwer Academic Publishers, 2004.
2. *Zambonelli F., Jennings N., and Wooldridge M.* Developing Multiagent systems: The GAIA methodology. ACM Transactions on Software Engineering and Methodology, 12(3). – 2003. – P. 417-470.
3. *Gorodetsky V., Karsaev O., Samoylov V., Konushy V.* Support for Analysis, Design and Implementation Stages with MASDK // LNCS 5386. 2009. – P. 272-287.

Карсаев Олег Владиславович
Санкт-Петербургский институт информатики и автоматизации РАН.
E-mail: ok@iias.spb.su.
199178, г. Санкт-Петербург, 14 линия, д. 39.
Тел.: 88123233570.

Конюший Виктор Григорьевич
E-mail: kvg@iias.spb.su.

Самойлов Владимир Владимирович
E-mail: samovl@iias.spb.su

Karsaev Oleg Vladislavovich
Saint-Petersburg Institute for Informatics and Automation RAS.
E-mail: ok@iias.spb.su.
39, 14 liniya, St. Petersburg, 199178, Russia.
Phone: 88123233570.

Konushy Victor Grigorievich
E-mail: kvg@iias.spb.su.

Samoylov Vladimir Vladimirovich
E-mail: samovl@iias.spb.su.

УДК 004.89

А.В. Смирнов, Т.В. Левашова, Н.Г. Шилов

КОНТЕКСТНО-ЗАВИСИМАЯ САМООРГАНИЗАЦИЯ РЕСУРСОВ ИНТЕЛЛЕКТУАЛЬНОЙ СРЕДЫ ПРИ СОВМЕСТНЫХ ДЕЙСТВИЯХ*

Целью исследований является разработка концепции самоорганизации ресурсов интеллектуальной среды в соответствии с контекстом текущей ситуации, имеющей место в среде. Задачами исследований являются разработка сервис-ориентированной архитектуры самоорганизующейся интеллектуальной среды и протокола самоорганизации ресурсов среды.

Интеллектуальная среда; сервис-ориентированная архитектура; протокол самоорганизации.

* Работа выполнена при финансовой поддержке РФФИ (проекты № 08-07-00264, 09-07-00436 и № 09-07-12111) и Президиума РАН (программа 14, проект № 213).