УДК 004.03:330.46

В. М. Гужва, канд. екон. наук, доцент,
ДВНЗ «КНЕУ імені Вадима Гетьмана»

МЕТОДИ ТА ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ДЛЯ ПОБУДОВИ МУЛЬТИАГЕНТНИХ СИСТЕМ В ЕКОНОМІЧНІЙ СФЕРІ

АНОТАЦІЯ. В умовах ринкової економіки однією із найнагальніших задач є задача розробки нових підходів до побудови систем управління економічними об'єктами в цілому (і інформаційних систем управління зокрема). Одним із найефективніших підходів у цьому контексті слід вважати агентно-орієнтований підхід — мова йде про агентно-орієнтоване моделювання (АОМ). По суті пропонується моделювати та досліджувати діяльність будь-якого економічного об'єкта (наприклад, промислового підприємства) як агентно-орієнтованої системи (АОС). Практична реалізація АОМ потребує наявності відповідних методів та інструментальних засобів. Огляду й аналізу таких методів і засобів присвячена ця стаття.

КЛЮЧОВІ СЛОВА. Система управління, метод, інструментальний засіб, агентно-орієнтоване моделювання, агентно-орієнтована система, мультиагентна система.

Вступ

Важливим напрямком у вирішенні проблем підвищення ринкової стійкості промислових підприємств є розробка підходів, моделей і методів її кількісної оцінки та керування, адекватних ринковим умовам господарювання. Таку розробку сьогодні важко уявити без використання сучасних засобів комп'ютерного моделювання в цілому і, зокрема, комп'ютерного імітаційного моделювання.

Комп'ютерне імітаційне моделювання як новий науковий напрямок виникло в 60-і роки минулого століття. На сьогодні воно включає чотири наступні типи: моделювання динамічних систем, системна динаміка, дискретно-подійне моделювання та агентне моделювання. Всі ці напрямки застосовуються, у тому числі, і для розв'язку економічних задач на різних рівнях абстракції. При цьому, чим більше модель відповідає об'єкту, що моделюється, при розв'язку конкретної проблеми, тим вона вважається більш адекватною. Агентне моделювання, розвиток якого прямо визначається обчислювальними можливостями сучасних комп'ютерів, дозволяє представити (змоделювати) систему практично будь-

якої складності з великою кількістю взаємодіючих об'єктів, не прибігаючи до їхнього агрегування.

В дев'яності роки минулого століття спостерігалося поступове збільшення кількості і можливостей методологій об'єктно-орієнтованого програмування (ООП). У першу декаду століття, що наступило, подібна ситуація повторюється з агентно-орієнтованими методологіями (АОМ). Однак, на відміну від ООП, АОМ поки ще не досягли стадії широкомасштабного промислового впровадження і використовуються невеликими групами дослідників в академічному середовищі.

Методи агеноно-орієнтованого моделювання соціально-економічних систем. Більшість АОМ-підходів на даний момент використовувалися лише для створення невеликих промислових систем.

Наявні на сьогодні АОМ-підходи можна розділити на чотири основних класи:

— методи, що базуються на об'єктно-орієнтованих методах і технологіях з використанням відповідних розширень;

— методи, що використовують традиційні методи інженерії знань;

— методи, що базуються на організаційно-орієнтованих поданнях;

— методи, що поєднують у різній мірі три перших класи.

В методологіях *першого класу* розробляються розширення об'єктно-орієнтованих методологій і технологій для проектування агентно-орієнтованих систем (АОС). Відомі спроби безпосереднього застосування UML-нотацій для подання агентно-орієнтованих систем і зразків взаємодії агентів. Однак ці пропозиції не можуть охопити автономність та проактивну поведінку агентів, так само, як і різноманіття їх взаємодій. З іншого боку мають місце пропозиції розширити та адаптувати об'єктно-орієнтовані моделі і технології для визначення методологій мультиагентних систем (МАС). Це веде, наприклад, до розширених моделей для подання агентного поводження і взаємодії агентів і агентно-орієнтованих розширень UML. І хоча ці засоби можуть забезпечити достатні описи автономного поводження агентів і їх взаємодій, вони не мають адекватних концептуальних механізмів для роботи з організаціями і зі спільнотами агентів.

Другий клас АОМ будується на розширенні традиційних методів інженерії знань. Ці методології забезпечують формальні і композиційні мови моделювання для верифікації структури системи і функцій і є прийнятними для моделювання орієнтованих на знання та інформаційно-орієнтованих агентів. Оскільки ці під-

ходи зазвичай припускають централізований погляд на системи, засновані на знаннях, вони не можуть забезпечити адекватні моделі і підходи для соціального розгляду мультиагентних систем. Прикладом може слугувати методологія MAS-CommonKADS — методологія розробки агентно-орієнтованого програмного забезпечення, призначена для застосування на етапах аналізу і проектування [6].

До *третього класу* відносяться моделі і підходи, спрямовані на створення АОС з «організаційно-орієнтованої» точки зору. По суті ці підходи визначають організацію як набір (колекцію) взаємодіючих ролей, не вирішуючи при цьому проблеми колективного поводження агентів. Прикладом може слугувати методологія Gaia [7]. Gaia — це методологія агентно-орієнтованого аналізу і проектування, що явно використовує організаційну точку зору. Найбільш абстрактною сутністю в ієрархії концептів Gaia є «система». Хоча термін «система» використовується в стандартному сенсі, він також означає «співтовариство» чи «організацію».

В останні роки розширюється клас методологій проектування АОС, що *поєднують у собі об'єктно-орієнтовані та організаційні підходи*. Прикладом може слугувати методологія PASSI, назва якої розшифровується як «процес для специфікації і реалізації агентних спільнот» (Process for Agent Societies Specification and Implementation). Вона є результатом тривалого періоду теоретичних досліджень і експериментів в області робототехніки.

Автори методології намагалися використовувати існуючі стандарти скрізь, де це можливо. Тому за мову моделювання обрана UML, для реалізації агентів використовується архітектура FIPA, а для подання знань при обміні повідомлень між агентами використовується XML. Застосування методології PASSI вимагає послідовної побудови сукупності моделей, що відбивають різні аспекти проектування АОС.

Інструментальні засоби для побудови мультиагентних систем. Проблема створення інструментальних засобів проектування мультиагентних систем (МАС) дуже інтенсивно досліджується в останні роки. Численні приклади обговорень можна знайти як в Інтернеті, так і в провідних закордонних фахових наукових виданнях. До числа найвідоміших проектів з багатьох інструментальних систем, орієнтованих на вирішення подібних задач, слід віднести такі, як AgentBuilder, Ascape, FIPA-OS, Grasshopper, Gypsy, JASON JATLite, JAFMAS, MAML, ProcessLink, Swarm, Zeus та інші.

Разом з тим слід зазначити, що практично не існує інструментальних засобів, які б задовольняли комплексу вимог, що пред'являються до них на сучасному рівні. Такий комплекс вимог можна розділити на два класи — *загальносистемні* та *інструментально-технологічні*.

До *першого класу* варто віднести:

— наскрізну підтримку всіх фаз життєвого циклу АОС і МАС;

— наявність єдиного інтегрованого середовища розробки всіх компонентів АОС;

— підтримка різних категорій користувачів;

— забезпечення засобів візуального проектування;

— автоматичне створення програмних кодів для інтелектуальних агентів (ІА);

— підтримка колективної роботи над проектом системи та автоматизованого документування усіх фаз процесу розробки.

До *другого класу* відносяться такі складові, як:

— забезпечення побудови розподілених баз знань і механізмів логічного висновку;

— підтримка процесу формування логічних моделей розподіленої системи (формування онтологій описуваних областей і розподіл знань по інтелектуальних компонентах АОС);

— забезпечення способів побудови моделей поводження ІА;

— реалізація механізмів паралельного функціонування, комунікації і координації агентів;

— підтримка модифікованості і розширюваності моделей ІА в процесі реального функціонування тощо.

В своїй більшості відомі на сьогодні засоби не вийшли поки що за рамки академічних проектів і не перетворилися в інструмент промислової технології виробництва МАС. Очевидно це справа майбутнього. У найближчі роки повинні виділитися лідери агентного промислового проектування, але видима розмаїтість інструментальних засобів у цій області буде більшою, чим у сучасних системах об'єктно-орієнтованого програмування (ООП), тому що в основі МАС лежить широкий спектр теоретичних моделей та архітектур агентів. Варто врахувати і зростаючу кількість нових логічних моделей і перехід до створення гетерогенних АОС, у яких розмаїтість типів ІА та інших компонентів буде значною.

Розглянемо деякі приклади існуючих інструментальних засобів для побудови АОС.

Серед розробок на теренах СНД варто виділити інструментальні засоби групи інтелектуальних систем Санкт-Петербургського ін-

ституту інформатики та автоматизації, очолюваної В. І. Городецьким [2], інструментальне середовище MagentAEngine, що базується на роботах В. А. Віттиха і П. О. Скобелєва [1].

Інструментальне середовище MAS-DK, розроблена колективом В. І. Городецького, орієнтоване на підтримку швидкого прототипування прикладних МАС і підтримує життєвий цикл програмних систем — від концептуального проектування до генерації і налагодження виконуваного коду ІА. Для генерації програмного коду ІА використовуються специфікації класів і окремих агентів.

Специфікація додатків починається з визначення онтології предметної області і протоколів взаємодії. Далі описуються класи агентів і призначаються ролі при виконанні протоколів. На наступному етапі розробляються моделі змінних класів, моделі керування, машини станів, бібліотеки сценаріїв поводження і бібліотеки зовнішніх функцій. Процес специфікації МАС на кожному із етапів підтримується інтегрованими графічними редакторами.

Дане інструментальне середовище успішно застосовувалося при створенні АОС для захисту комп'ютерних мереж, розподіленого навчання на основі класифікації і інтелектуальної обробки даних.

Досить широке застосування, особливо в академічних та навчальних цілях, отримала платформа мультиагентного програмування JADE (Java Agent Development Framework), цілком реалізована мовою Java. Проект розробляється компанією Telecom Italia з 2000 року, є вільно розповсюджуваним програмним забезпеченням (ПЗ) по ліцензії LGPL (Lesser General Public License Version 2) [3].

JADE пропонує програмісту — розробнику агентних систем наступні інструментальні засоби:

— FIPA-compliant Agent Platform — агентну платформу, що базується на рекомендаціях FIPA і включає обов'язкові типи системних агентів: керування агентною платформою (AMS — Agent Management Service), каналу комунікації (ACC — Agent Communication Channel) і служби каталогів (DF — Directory Facilitator). Ці три типи агентів автоматично активуються при запуску платформи;

— Distributed Agent Platform — розподілену агентну платформу, що може використовувати кілька комп'ютерів (вузлів), причому на кожному вузлі запускається тільки одна віртуальна машина (Java Virtual Machine). Агенти виконуються як Java-потоки. Для доставки повідомлень між агентами, в залежності від їх міс-

цезнаходження, використовується відповідний транспортний механізм — Multiple Domains support — ряд заснованих на FIPA-специфікаціях DF-агентів, що можуть об'єднуватися у федерацію, реалізуючи мультидоменне агентне середовище;

— Multithreaded execution environment with two-level scheduling — багатопотокове середовище виконання з дворівневим плануванням. Кожен JADE-агент має власний потік керування, але також здатний працювати в багатопотоковому режимі;

— Library of interaction protocols — бібліотеку протоколів взаємодії з використанням стандартних інтерактивних протоколів fipa-request і fipa-contract-net. Для створення агента, що функціонує згідно з даними протоколами, розробнику необхідно реалізувати тільки специфічні доменні дії, уся незалежна від прикладної програми протокольна логіка буде здійснюватися системою JADE;

— Administration GUI — графічний інтерфейс адміністратора, що забезпечує прості операції керування платформою та відображає активних агентів та контейнери агентів. Використовуючи GUI, адміністратори платформи можуть створювати, знищувати, переривати і відновляти дії агентів, створювати ієрархії доменів і мультиагентні федерації DF.

JADE написана мовою програмування Java з використанням Java RMI, Java CORBA IDL, Java Serialization і Java Reflection API. Вона спрощує розробку мультиагентних систем завдяки використанню FIPA-специфікацій та інструментів, що підтримують фази налагодження і розгортання системи. Агентна платформа може встановлюватися на комп'ютерах з різними операційними системами та конфігуруватися через віддалений GUI-інтерфейс навіть під час виконання агентів. Комунікаційна архітектура пропонує гнучкий і ефективний процес обміну повідомленнями, де JADE створює чергу і керує потоком ACL-повідомлень, які є приватними для кожного агента.

Агенти здатні звертатися до черги за допомогою комбінації декількох режимів роботи: блокування, голосування, перерви в роботі та зіставлення з еталоном. На даний момент у системі використовується Java RMI, event-notification і IIOP, але можна додати й інші протоколи. Також передбачена можливість інтеграції SMTP, HTTP і WAP. Більшість комунікаційних протоколів, що вже визначені міжнародним співтовариством розробників AOC, доступні і можуть використовуватися після визначення поводження системи і її основних станів. Онтологія керування агентами реалізована з підтримкою визначених користувачем контентних мов.

Останнім часом визначений інтерес викликає інструментарій INGENIAS Development Kit (IDK). Це середовище є розробкою дослідницької групи GRASIA з університету Мадрида [4].

У сукупності IDK являє собою набір інструментальних засобів для опису, перевірки і реалізації МАС. Дане середовище розробки є функціонально замкненим та платформо-незалежним. Кросплатформенність забезпечується за рахунок реалізації мовою Java, а функціональна замкнутість виявляється в тому, що середовище дозволяє здійснювати повний цикл проектування МАС — від візуального опису до фінального моделювання.

Даним інструментарієм можуть користуватися як програмісти-початківці, яким потрібно описати свою МАС і сформувати для неї код, так і досвідчені розробники, що бажають доповнити функціонал IDK за рахунок власних модулів.

У складі IDK передбачені наступні модулі:

— HTML-модуль дозволяє побудувати набір HTML-файлів на основі опису мультиагентної системи;

— JADE-модуль перетворить опис МАС у код, що виконується, на платформі JADE. На виході даного модуля розробник одержує набір агентів, поводження кожного з яких описано в метамоделі агента, а взаємодія між ними описана в метамоделі взаємодії;

— JADE Leap-модуль є модифікацією JADE-модуля і дозволяє адаптувати розроблену МАС на платформу JADE Leap для наступного використання в мобільних пристроях;

— модуль верифікації коду перевіряє опис МАС на наявність недовизначених сутностей і вказує місце розташування помилок.

Моделювання МАС здійснюється при використанні Ant Apache XML Parser, що завантажує JADE-контейнер і запускає на ньому зазначену МАС.

IDK підтримує два види візуального опису — мову AUML і методологію INGENIAS. AUML є по суті розширенням мови UML, призначене для опису АОС. Методологія INGENIAS ґрунтується на визначенні набору метамоделей, що описують поводження кожного агента, взаємодію між агентами, організацію МАС, оточення, цілі та задачі, визначені для кожного агента. Подання МАС, складене в контексті метамоделі, відображається на однойменній діаграмі.

Методологія передбачає побудову наступних метамоделей:

— організаційної метамоделі, що описує архітектуру МАС і її функції. Функції визначаються в момент визначення цілей організації і порядку їхнього виконання;

— метамоделі оточення, що описує сутності, які впливають на сприйняття агентів;

— метамоделі цілей/задач, що описує зміни станів агента в часі залежно від виконання конкретних задач, шляху досягнення цілей і дії, що починаються у випадку, якщо ціль не може бути досягнута;

— метамоделі агента, що описує поводження кожного агента;

— метамоделі взаємодії, що описує поводження двох і більш взаємодіючих агентів. Залежно від мови опису вона може бути представлена у вигляді UML-діаграми кооперації, AUML-діаграми чи GRASIA-діаграми взаємодії.

Отриманий у вигляді метамоделей опис МАС може оброблятися відповідними модулями.

Поводження агентів визначається їхніми цілями і задачами, а також взаємодіями між ними. Цілі у свою чергу можуть розбиватися на більш прості підцілі, які повинні досягатися визначеними планами, послідовностями задач. У методології INGENIAS будь-яка взаємодія може мати одного ініціатора та одного чи кількох учасників. Потік повідомлень у МАС може бути описаний діаграмою Grasia чи іншими діаграмами взаємодії, такими, як UML чи Agent UML. У INGENIAS будь-яка взаємодія починається з задачі. Сукупність задач і взаємодій визначає глобальне поводження системи. На основі всього опису системи відбувається генерація коду. Отриманий опис може бути використаний для створення програмного коду МАС під конкретну платформу.

Удосконалювання інструментальних засобів АОП виявляється в тому, що вони стають здатними практично реалізувати строгі теоретичні моделі МАС, особливо архітектуру BDI. У цьому сенсі яскравим прикладом є інструментальний пакет JASON, що є реалізацією відомої мови опису МАС AgentSpeak(L) [5].

JASON є першим повним інтерпретатором мови AgentSpeak(L), що включає можливості міжагентної комунікації на основі теорії мовних актів.

Даний інтерпретатор забезпечує такі можливості, як:

— сильне заперечення, що допускає розгляд як замкнутих, так і відкритих моделей світу;

— обробка помилок у планах, виконуваних агентами;

— посилання на мітки плану, що можуть використовувати складні вирішальні функції;

— підтримка середовищ розробки, що не реалізують спочатку AgentSpeak(L) через програмування мовою Java;

— запуск МАС розподілених у локальній мережі;

— використання функцій вибору, що набудовуються, функцій довіри, ревізії переконань, міжагентної комунікації і дій;

— використання бібліотек внутрішніх дій, визначених користувачем мовою Java.

За твердженням розроблювачів у системі JASON реалізована операційна семантика, формально визначена для мови AgentSpeak(L).

Дана мова визначає агента через специфікацію множини базових переконань і планів. Атомарне переконання є звичайним предикатом першого порядку. Атомарні переконання чи їхні заперечення позначаються відповідними літералами. Початкова множина переконань є набором базових атомарних переконань.

У мові AgentSpeak(L) визначаються два типи цілей: досяжні цілі і тестові цілі. Досяжна мета встановлюється тоді, коли агент прагне досягти визначеного стану світу, у якому зв'язаний предикат є дійсним. Практично для цього ініціюється виконання вкладених планів.

Тестова мета повертає уніфікацію для зв'язаного предиката з одним з переконань агента, у випадку невдачі така ціль відкидається.

Перемикаюча подія може ініціювати виконання деякого плану. Подія може бути внутрішньою, коли досягнута визначена підціль, чи зовнішньою, породжувану у результаті модифікації переконань після сприйняття стану зовнішнього середовища. Плани використовують базові дії, що агент здатний виконувати в середовищі. Дії також визначаються як предикати першого порядку, але позначаються спеціальними символами.

План утвориться перемикаючою подією, за якою випливає кон'юнкція літералів, що утворять контекст. Контекст повинний бути послідовністю таких поточних переконань агента, для яких даний план може бути застосованим. Залишкова частина плану є послідовністю базових дій і цілей, що агент досягає (чи перевіряє), коли план обраний для виконання.

Програми агентів інтерпретуються циклічно. У кожному циклі оновлюється список подій, породжуваних або перцепцією із зовнішнього середовища, або виконанням підцілей, визначених у тілі плану. Далі інтерпретатор уніфікує обрану подію з перемикаючими подіями в заголовках планів, визначаючи, таким чином, множину релевантних планів. Перевіряючи вміст контексту цих планів, інтерпретатор визначає множину застосовних планів і вибирає один прийнятний план з цієї множини. Коли усі формули в тілі плану виявляються виконаними, то ціль вважається досягну-

тою і план також видаляється з черги планів. Цикл роботи інтерпретатора повторюється.

На сьогодні з використанням AgentSpeak(L) пов'язане невелике число діючих додатків, серед яких у літературі згадуються програми моделювання росту міст і керування складським роботом у середовищі віртуальної реальності.

Реалізація інтерпретатора JASON виглядає продуманою і дуже елегантною, що дозволяє припустити більш широке його використання як у дослідницьких, так і прикладних проектах.

Незважаючи на значну кількість інструментальних засобів розробки МАС, проблема створення інструментарію, що поєднує теоретично обґрунтовану методологію проектування й ефективну реалізацію в об'єктно- орієнтованому середовищі, як і раніше повною мірою не вирішена.

Література

1. *Виттих В. А.* Мультиагентные модели взаимодействий для построения сетей потребностей и возможностей в открытых системах / В. А. Виттих, П. О. Скобелев // Автоматика и телемеханика. — 2003. — № 1. — С. 177—185.

2. *Городецкий В. И.* MAS DK: инструментарий для разработки многоагентных систем и примеры приложений / В. И. Городецкий, О. В. Карсаев, И. В. Хотенко, А. В. Хабалов // Труды Международного конгресса «Искусственный интеллект в XXI веке» (ICAI 2001), 3—8 сентября 2001 г. — М.: Физматлит, 2001. — С. 249—262.

3. http://jade.tilab.com/

4. *Pavyn J.* The INGENIAS Methodology and Tools / J. Pavyn, J. Gymez-Sanz & Rubйn Fuentes // Agent-Oriented Methodologies, Brian Henderson-Sellers & Paolo Giorgini (eds.), Idea Group Publishing, 2005, p. 236—276.

5. *Bordini R. H., Hubner J. F., Wooldridge M.* Programming Multi-Agent Systems in AgentSpeak with Jason. — Jonh Wiley&Sons: Chichester, 2007. — 294 p.

6. Schreiber, G., Akkermans, H., Anjewierdern, A., de Hoog, R., Shadbolt, N., Van De Velde, W., Wielinga, B.. Knowledge Engineering and Management: the Common-KADS Methodology. MIT Press. Cambridge, MA. 2001. — 455 p.

7. *Wooldridge, M.* The Gaia Methodology for Agent-Oriented Analysis and Desing / M.Wooldridge, N. R. Jennings, D. Kinny // Journal of Autonomous Agents and Multi-Agent Systems. — 2000. — 3(3). — P. 285—312.