Інформаційна Технологія Управління Контентом у Системах Електронного Навчання

Ірина Кириченко
кафедра програмної інженерії
Харківський національний університет радіоелектроніки
Харків, Україна
iryna.kyrychenko@nure.ua

Information Technology of Content Management in E-Learning Systems

Iryna Kyrychenko
Department of Software Engineering
Kharkiv National University of Radio Electronics
Kharkiv, Ukraine
iryna.kyrychenko@nure.ua

***Анотація*—Управління контентом потребує пошуку та збору певної інформації. До контенту електронного навчання висуваються певні вимоги, які виражено моделлю. Для моделювання використано алгебру скінченних предикатів. Реалізація пошуку здійснюється за допомогою мультиагентної системи. Технологія збору навчального контенту представлена послідовністю кроків, які необхідно реалізувати.**

***Abstract*—Content management requires the search and collection of certain information. To the content of e-learning are put forward certain requirements, expressed by the model. The algebra of finite predicates is used for modeling. The search is carried out by using the multi-agent system. The technology of collecting educational content is represented by a sequence of steps that need to be implemented.**

Ключові слова—управління контентом; інформаційна технологія; агент; моделювання

Keywords—content management; information technology; agent; modeling

I. Вступ

Зацікавленість у системах електронного навчання [1] не згасає вже не одне десятиліття. При цьому для користувача важливо відібрати саме той курс, що його цікавить та задовольняє всім висунутим вимогам [2, 3, 4]. Мультиагентні системи інформаційного пошуку навчальних ресурсів дозволяють виконати представлені вимоги.

Агенти можуть бути досвідченими асистентами для пошуку необхідної інформації, даючи людям можливість використовувати інформацію опосередковано та ефективно.

II. Результати

Під агентною платформою (агентним середовищем, Agent Environment) зазвичай розуміють набір програмних інтерфейсів (API – Application Programming Interface), який надає можливості створення, життєвого циклу, обміну повідомленнями, зв'язку та доступу до інформації та баз знань агентів [5, 6]. Агентну платформу наділяють функціями базового життєвого середовища агентів. Основними функціями агентної платформи є: керування життєвим циклом агентів (народження, життя, смерть); обмін повідомленнями між агентами та агентами і агентною платформою; підтримка мобільності агентів; забезпечення безпеки впродовж життя агентів (для локальних систем та самих агентів); забезпечення низькорівневої інфраструктури (використання можливостей операційної системи для виконання своїх функцій – наприклад використання мережних функцій операційної системи для забезпечення взаємодії з іншою агентною платформою).

Агентна платформа є найважливішою частиною в процесі створення агентної системи. Під агентною платформою розуміють спеціалізований набір програмних інтерфейсів. Агентне середовище – конкретна реалізація агентної системи, конкретне «живе» середовище, яке функціонує, в якому утримуються якісь знання, та можуть народжуватися агенти.

Агентна система - це платформа, що може створювати, інтерпретувати, запускати, переміщати і знищувати агенти. Також як і агент, агентна система асоціюється з повноваженнями, що визначають організацію або персону, від імені яких працює система. Агентна система з повноваженнями конкретного користувача, реалізує

політику безпеки цього користувача в плані захисту його ресурсів.

Агентна система однозначно ідентифікується ім'ям і адресою. На одній машині можуть розташовуватися кілька агентних систем. Тип агентної системи описує сукупність параметрів агента. Наприклад, якщо типом агентної системи є «агенти», то це означає, що агентна система створена компанією IBM, підтримує мову Java як мову реалізації агентів, і використовує Java Object Serialization для перетворення агентів у послідовну форму.

Усе спілкування між агентними системами відбувається через комунікаційну інфраструктуру. Адміністратор мережевого регіону визначає служби комунікації для регіональних і міжрегіональних взаємодій. Комунікаційна інфраструктура забезпечує транспортні служби зв'язку (наприклад RPC), службу імен і службу безпеки для агентських систем [7].

Коли агент переміщується, він рухається між стаціонарними процесами, що називаються місцями [7]. Місце - контекст усередині агентської системи, у якому може бути запущений агент. Цей контекст може надавати набір функцій, таких як контроль доступу, наприклад. Вихідне місце й місце адресації можуть знаходитися як у межах однієї агентської системи, так і в різних агентських системах, що підтримують однакову сукупність параметрів агента. Місце зіставляється з місцем розташування. Агентна система може містити одне або кілька місць і місце може містити одного або більш одного агентів. Хоча місце визначається як оточення, у якому запускається агент. Якщо агентна система не реалізує поняття місця, то місце визначається за замовчуванням. Коли клієнт запитує місце розташування агента, то як відповідь він одержує адресу місця, де знаходиться агент.

Варто зазначити, що наразі не існує мови програмування або інструментальної системи розробки, яка б повністю відповідала потребам побудови агентів. Така система повинна була б відповідати таким вимогам: забезпечення перенесення коду на різноманітні платформи, доступність на багатьох платформах, підтримка мережної взаємодії, багатопотокова обробка та інші [7, 8, 9].

Частіше всього в агентних технологіях використовуються: універсальні мови програмування (Java); мови, «орієнтовані на знання», такі, як мови представлення знань (KIF); мови переговорів й обміну знаннями (KQML, AgentSpeak, April), мови специфікацій агентів; спеціалізовані мови програмування агентів (TeleScript); мови сценаріїв й scripting languages (Tcl/Tk); символьні мови й мови логічного програмування (Oz) [7].

Одна з найголовніших властивостей агента – це інтелектуальність. Інтелектуальний агент володіє визначеними знаннями про себе і про навколишнє середовище, і на основі цих знань він здатний визначати свою поведінку. Інтелектуальні агенти є основною сферою інтересів агентної технології. Важливе також середовище існування агента: це може бути як реальний світ, так і віртуальний, що стає важливим у зв'язку з широким розповсюдженням мережі Internet. Від агентів вимагають здатності до навчання й навіть самонавчання.

Існує багато інструментальних рішень для проектування МАС (Media Access Control), та найпопулярнішими серед них є агенті платформи. Агентна платформа – це проміжний виконавчий рівень, який знаходиться між агентами та операційною системою [10, 11]. У деяких випадках вона може спиратися не на саму операційну систему, а на якусь платформу (Java Virtual Machine або .NET Framework).

Основні функції агентної платформи:

- представляє собою середовище для існування та взаємодії агентів;
- реалізує певні стандарти для забезпечення взаємодії та сумісності з іншими платформами.

У таблиці 1 наведена порівняльна характеристика найбільш відомих агентних платформ [12].

Для реалізації інформаційної системи веб-моніторингу була використана платформа JADE. До основних переваг цієї платформи можна віднести можливість інтеграції з іншими системами, підтримку стандартів специфікації FIPA-2000 та вільне розповсюдження [12].

Агентна платформа JADE може бути розподілена між декількома комп'ютерами [12]. Віртуальна машина Java – це основний контейнер агентів, що забезпечує середовище виконання агента і дозволяє кільком агентам одночасно виконуватися на тому ж самому хості.

ТАБЛИЦЯ І. ПОРІВНЯЛЬНА ХАРАКТЕРИСТИКА АГЕНТНИХ ПЛАТФОРМ

Критерій порівняння	**Платформи**			
	JADE	***Cougaar***	***ZEUS***	***Jason***
Мова програмування	Java	Java	Візуальні генератори коду	AgentSpeak
FIPA-сумісність	+	-	+	-
Ліцензія на використання	Не потрібна	Не потрібна	Потрібна	Не потрібна
Область застосування	Розподілені системи, що складаються з багатьох компонент	Розподілені системи, що складаються з багатьох компонент	Основа для написання правил та сценаріїв	Розподілені системи, що складаються з багатьох компонент

Для створення МАС використовуються наступні основні класи: Agent і Behavior. Клас Agent (агент) представляє собою загальний базовий клас для агентів, обумовлених користувачем. З точки зору розроблювача, агент JADE – звичайний екземпляр Java-класу, що розширює базовий клас Agent. Мається на увазі успадкування властивостей для здійснення основних взаємодій з агентною платформою (реєстрація, конфігурація, віддалене керування тощо) і основний набір методів, що можуть викликатися для реалізації заданого поводження агента [7]. Обчислювальна модель агента є багатозадачною і паралельною, у якій задачі (чи поводження) виконуються одночасно. Кожна функціональна можливість і/чи сервіс, наданий агентом, повинні бути реалізовані як одне чи кілька поводжень. Внутрішній планувальник, схований від розроблювача, автоматично керує плануванням поводжень.

Агент JADE може знаходитися в одному з кількох станів, представлених у класі Agent наступними об’єктами:

- AP_INITIATED: об'єкт класу Agent створений, але поки незареєстрований у AMS, тому він не має ні імені, ні адреси і не може вести переговори з іншими агентами;
- AP_ACTIVE: об'єкт Agent зареєстрований у AMS, має постійне ім'я й адресу і може використовувати всі сервіси JADE;
- AP_SUSPENDED: об'єкт Agent у даний момент часу зупинений і не виконує ніякої дії (поводження);
- AP_WAITING: Agent блокований, його внутрішній стан – без дії і він пробудиться при виконанні деякої умови (наприклад, надходження повідомлення);
- AP_DELETED: внутрішній стан – завершений, і агент не зареєстрований у AM;
- AP_TRANSIT: мобільний агент вводить цей стан для того, щоб, поки агент переміщається до нового місця розташування, система продовжувала накопичувати і зберігати повідомлення, що будуть надіслані йому, коли він прибуде в нове місце;
- AP_COPY: внутрішній стан використовується платформою JADE для створення копії агента;
- AP_GONE: внутрішній стан використовується платформою JADE для повідомлення про те, що мобільний агент перемістився до нового місця розташування й установив стійкий стан.

Клас Behaviour (поведінка). Розроблювач визначає дії агента шляхом специфікації його поводження. Агент здатний виконувати декілька паралельних задач у відповідь на різні зовнішні події. Для того щоб керування агентом було ефективним, кожен агент JADE складається з окремого потоку виконання і всі його задачі (наміри) повинні бути реалізовані як об'єкти класу Behaviour [12]. Розроблювач при постановці задачі агентові повинен визначити один чи кілька похідних класів від базового класу Behaviour і додати певні поводження до списку його задач. Клас Agent надає два методи: addBehaviour і removeBehaviour, що дозволяють керувати чергою задач агента, а саме додавати або видаляти поводження. Поводження агента можуть бути додані кожного разу, коли це необхідно, а не тільки в межах методу Agent.setup.

Spring Framework — це програмний каркас (фреймворк) з відкритим кодом та контейнера з підтримкою інверсії управління для платформи Java. Основні особливості Spring Framework можуть бути використані будь-яким додатком Java, але є розширення для створення веб-додатків на платформі Java EE. Незважаючи на це, Spring Framework не нав'язує якоїсь конкретної моделі програмування, Spring Framework став популярним в спільноті Java як альтернатива, або навіть доповнення моделі Enterprise JavaBean (EJB) [13].

Spring Framework складається з кількох модулів, які надають широкий спектр послуг:

- Контейнер Інверсії управління: Конфігурація компонентів додатків і управління життєвим циклом об'єктів Java, здійснюється головним чином через Інверсію управління.
- Аспектно-орієнтоване програмування: дозволяє реалізувати наскрізні процедури.
- Доступ до даних: робота з реляційною системою управління базами даних на платформі Java з використанням JDBC і об'єктно-реляційні відображення та інструментів з NoSQL баз даних.
- Управління транзакціями: об'єднує кілька API, управління транзакціями та координує операції для Java-об'єктів.
- Модель-Вигляд-Управління (Model-View-Controller): програмний каркас на основі HTTP сервлета, що забезпечує створення веб-додатків і веб-служб RESTful.
- Аутентифікація і авторизація: налаштування процесів безпеки, які підтримують цілий ряд стандартів, протоколів, інструментів і практик за допомогою підпроекту Spring Security (колишня система безпеки AcerI для Spring).
- Віддалене управління: конфігураційний вплив і управління Java-об'єктами для місцевої (локальної) або віддаленої конфігурації через JMX.
- Тестування: підтримка класів для написання юніт-тестів та інтеграційних тестів.

Незалежність баз даних може бути реалізована за допомогою шаблону Репозиторій (Repository). Такий підхід використовується для відокремлення логіки, яка отримує дані, і перетворює її в моделі сутностей для подальшої роботи на доменному рівні. На використання чистої архітектури та принципу інверсії залежностей для забезпечення повного контролю над компонентами системи використовується шаблон впровадження залежностей (Dependency Injection). Таким чином, даний підхід надає можливість підміни залежностей об’єктів системи за допомогою такого контейнеру без необхідності зміни інших компонентів системи [13].

В процесі розробки зроблено висновок, щодо застосування мікросервісної архітектури та написання окремих модулей системи як незалежних програм, виконання яких буде оброблятися агентною платформою, тобто поведінка (Behavior) агенту буде містити бізнес логіку окремого компоненту (сервісу) системи. Так за допомогою аспектно-орієнотованого програмування виконання методів та обробка їх результатів надавалась агентам, які для взаємодії сервісів використовують ACL середовище обміну повідомлень.

Алгоритмічне забезпечення інформаційної технології збору навчального контенту складається з наступної послідовності кроків:

Крок 1. Ініціалізація агента А0 із множиною станів $S_{A0} = \{s_i, i = \overline{1,7}\}$, множиною сприйняттів $P_{A0} = \{p_j, j = \overline{1,5}\}$ та множиною дій $A_{A0} = \{a_k, k = \overline{1,6}\}$.

Крок 2. Агент А0 отримує модель пошуку навчального контенту

$$M_{SEARCH} = \langle R_{SEARCH}, R_{SOURCE}, P_{SEARCH}, P_{PATTERN}, Seed \rangle .$$

Крок 3. Агент А0 перевіряє чергу посилань у кеш-пам'яті згідно із функцією сприйняття $f_{A0} : S_{A0} \to P_{A0}$.

Крок 4. Агент А0 обирає дію на основі функції вибору дії $g_{A0} : P_{A0} \to A_{A0}$.

Крок 5. Агент А0 створює першого агента А1 із моделлю теми навчального об'єкта та стартовим посиланням.

Крок 6. Агент А0 створює агента А2 із моделлю навчального об'єкта.

Крок 7. Агент А0 створює агента А3 із моделлю шаблону метаопису.

Крок 8. Ініціалізація агента А1 із множиною станів $S_{A1} = \{s_i, i = \overline{1,4}\}$, множиною сприйняттів $P_{A1} = \{p_j, j = \overline{1,4}\}$ та множиною дій $A_{A1} = \{a_k, k = \overline{1,4}\}$.

Крок 9. Агент А1 перевіряє посилання згідно із функцією сприйняття, основаною на моделі $f_{A1} : S_{A1} \overset{K_{A1}}{\to} P_{A1}$.

Крок 10. Агент А1 обирає дію на основі функції вибору дії $g_{A1} : P_{A1} \overset{M_{A1}}{\to} A_{A1}$.

Крок 11. Ініціалізація агента А2 із множиною станів $S_{A2} = \{s_i, i = \overline{1,2}\}$, множиною сприйняттів $P_{A2} = \{p_j, j = \overline{1,3}\}$ та множиною дій $A_{A2} = \{a_k, k = \overline{1,6}\}$.

Крок 12. Агент А2 перевіряє інформаційний ресурс згідно із функцією сприйняття, заснованою на моделі $f_{A2} : S_{A2} \overset{K_{A2}}{\to} P_{A2}$.

Крок 13. Агент А2 обирає дію на основі функції вибору дії $g_{A2} : P_{A2} \overset{M_{A2}}{\to} A_{A2}$.

Крок 14. Ініціалізація агента А3 із множиною станів $S_{A3} = \{s_i, i = \overline{1,3}\}$, множиною сприйняттів $P_{A3} = \{p_j, j = \overline{1,2}\}$ та множиною дій $A_{A3} = \{a_k, k = \overline{1,3}\}$.

Крок 15. Агент А3 перевіряє знайдений ресурс згідно із функцією сприйняття, заснованою на моделі навчального об'єкта $f_{A3} : S_{A3} \to P_{A3}$.

Крок 16. Агент А3 обирає дію на основі функції вибору дії $g_{A3} : P_{A3} \to A_{A3}$.

Таким чином, розроблена прикладна інформаційна технологія передбачає агентну реалізацію етапів пошуку та збору навчального контенту.

III. Висновки

Таким чином, для управління контентом у системах електронного навчання проводиться пошук та збір певної інформації. Відповідно до вимог, що висуваються до контенту, математично описується модель за допомогою алгебри скінченних предикатів. Пошук реалізовано шляхом використання мультиагентної системи.

Основними перевагами такого підходу є :

- незалежність від фреймворків та конкретних бібліотек;
- легкість у тестуванні – бізнес правила можуть бути протестовані окремо, без користувацького інтерфейсу, баз даних тощо.
- незалежність користувацького інтерфейсу – відображення може бути зміненим без впливу на інші компоненти системи, що знижує шанси виникнення потенційних помилок;
- незалежність від платформи – бізнес правила не знають, де вони будуть використані, і не прив'язані до особливостей конкретної платформи;
- незалежність баз даних – бізнес логіка додатку не прив'язана до конкретної бази даних, що дає можливість її зміни у будь-який момент часу без впливу на інші складові системи.

Література References

[1] http://www.dut.edu.ua/uploads/l_1216_25218115.pdf

[2] Маркетингова інформація. Маркетингове дослідження // http://pidruchniki.com/12640422/marketing/marketingova_informatsiya_marketingovi_doslidzhennya

[3] Маркетингова інформаційна система (МІС) // http://pidruchniki.com/1628041460643/marketing/marketingova_informatsiyna_sistema_mis

[4] Система збору зовнішньої маркетингової інформації // http://stud.com.ua/49872/marketing/sistema_zboru_zovnishnoyi_marketingovoyi_informatsiyi#41

[5] Методичні вказівки: Інтелектуальні агенти // http://eir.zntu.edu.ua/bitstream/123456789/2178/1/subbotin_methodical_instructions.pdf

[6] Ткачук Н.В. // Глосарій деяких сучасних термінів програмної інженерії – 1-6 с – 2005

[7] Дослідження логічних моделей семантики переговорів інтелектуальних агентів в мультиагентних систамх // http://science.donntu.edu.ua/ius/kirgaev/diss/indexu.htm

[8] SWEBOK Guide V3.0 // ISBN-10: 0-7695-5166-1 // ISBN-13: 978-0-7695-5166-1 // 2013

[9] С.М. БУРБЕЛО, О.С. СТАРОДУБ, М.С. БОГДАНОВА // Вибір гнучких методів розробки програмного забезпечення - УДК 004.738.5 - 2013

[10] Ю.М. Гонтар, О.Ю. Чередніченко, О.В. Янголенко, М.А. Вовк - розробка розподіленої системи обробки бізнес-інформації з використанням агентного підходу / Національний технічний університет «Харківський політехнічний інститут», Харків – 137-141с - УДК 004.91

[11] Пошукові можливості web-систем // http://eprints.isofts.kiev.ua/331/1/05_andon.pdf

[12] О. А. Симоненко , О. Я. Сова , В. А. Романюк , Я. Л. Уманець. Аналіз існуючих агентних платформ для побудови систем управління вузлами мобільних радіомереж класу manet Системи обробки інформації. — 2014. — № 1(117). — С. 200-203.

[13] Spring Framework // https://wikivisually.com/lang-uk/wiki/spring_framework