УДК 004.91

Ю.М. Гонтар, О.Ю. Чередніченко, О.В. Янголенко, М.А. Вовк

Національний технічний університет «Харківський політехнічний інститут», Харків

РОЗРОБКА РОЗПОДІЛЕНОЇ СИСТЕМИ ОБРОБКИ БІЗНЕС-ІНФОРМАЦІЇ З ВИКОРИСТАННЯМ АГЕНТНОГО ПІДХОДУ

В статті розглянуто питання проектування інформаційної системи для обробки інформації, на якій базуються бізнес-процеси підприємства. Виходячи із великих обсягів даних, які необхідно аналізувати, та зважаючи на колаборативну природу рекомендаційних систем, обґрунтовано вибір розподіленої архітектури. Для реалізації інформаційної системи запропоновано використовувати агентний підхід. Розроблено формальну модель архітектури агента. Проаналізовано можливості агентних платформ та технологій зберігання даних для створення багатоагентної системи обробки бізнес-інформації.

Ключові слова: *розподілена інформаційна система, бізнес-інформація, агентний підхід, агентна платформа.*

Вступ

Розвиток наукових підходів в управлінні інформаційним забезпеченням підприємства обумовлено процесами інтеграції, глобалізації та трансформації економіки України. В таких умовах виникає потреба в розробці механізмів використання інформації, які поряд з фінансовими і нематеріальними можливостями підприємства забезпечували б його конкурентоспроможність. Залежно від організації процесів введення, обробки і зберігання даних, інформаційні системи управління підприємством можна розділити на наступні типи: облікові системи, комплексні системи обробки даних, інтегровані системи і динамічні системи управління [1].

Основний недолік існуючих систем – недотримання критерію «актуальності» інформації, а також низький рівень використання інформаційних мереж, неможливість оцінити стан всіх ресурсів підприємства і розширити коло осіб, які беруть участь у бізнес-процесі.

Діяльність більшості підприємств базується на значному архіві електронної документації. Ця документація створюється на основі функціонування бізнес-процесів та являє собою бізнес-інформацію. Нові бізнес-процеси на підприємстві можуть або використовувати раніше розробленні рішення, так як повторність використання зменшує час їх виконання, або створюватись заново на основі нової або вже існуючої в архіві бізнес-інформації [2].

Рекомендаційні системи набувають все більшого поширення як перспективна технологія, покликана допомогти користувачу вирішити проблему вибору серед великої кількості альтернатив [2]. Використання рекомендаційних систем в колаборативних середовищах є актуальним та дозволяє застосовувати нові алгоритми фільтрування, що підвищує ефективність рекомендацій.

Аналіз підходів щодо побудови інформаційної системи

Існує цілий ряд задач, які складно або неможливо розв’язати без використання агентного підходу, до яких належать наступні [3].

1. Мобільні обчислення; процес відбувається за рахунок пересування агентів між мобільними платформами, при цьому не потрібне постійне підключення до мережі, клієнт може переслати агента на сервер для виконання задачі та від’єднатися від мережі. Приєднавшись у іншій точці мережі, клієнт може забрати результати роботи агента.

2. Обробка інформації; у цьому випадку агент відбирає із усієї знайденої інформації тільки ту, яка потрібна клієнту.

3. Моніторинг даних; агент може сповіщати користувача щодо змін у джерелах інформації у реальному часі.

4. Універсальний доступ до даних; агенти можуть бути посередниками для роботи з різними джерелами даних, які мають механізми для взаємодії один з одним.

5. Пошук інформації; агент може пересуватися мережею та шукати інформацію за запитом, при цьому керуючись поставленими умовами щодо різних її характеристик (типу джерела інформації, його надійності, типу файлу).

Більшість розробників систем, що базуються на агентах, використовують інтуїтивний підхід до проектування. Авторами української теоретичної школи програмування (Ф.І. Андон, А.В. Анісімов, Д.Б. Буй, В.В. Бублик, В.М. Глушков, С.С. Гороховський, А.Ю. Дорошенко, К.М. Лавріщева, О.А. Летічевський, М.С. Нікітченко, Г.О. Цейтлін та ін.) [4 – 6] створені нові підходи до розв’язання проблем програмування. При виборі розподіленої архітектури програмних систем виділяють такі пе-

реваги: можливість одночасної роботи групи користувачів з системою як над окремими, так і над спільними задачами; можливість використання значних за обсягом баз знань, розміщених на різних вузлах мережі; масштабованість, відкритість, стійкість до відмов та ін. [7].

Виходячи з вимог та обмежень, які ставляться перед розробниками інформаційних систем, було запропоновано декілька підходів до розподілення. Серед них базовими є два: повний розподіл на окремі структурні компоненти, що взаємодіють між собою, та розподіл на рівні баз знань та надання доступу до них через Інтернет [7]. Виділяють окремо підходи, у яких розглянуто застосування Інтернет та Web-технологій для спільного використання баз знань при груповій роботі користувачів [7, 8].

Як показав проведений аналіз, існуючі підходи мають наступні недоліки. Повний розподіл призводить до надмірних витрат на зв'язок між компонентами, особливо при роботі механізму виводу з бази знань.

Спосіб, який пропонується для розподілу на рівні баз знань, ґрунтується на схемі представлення знань у вигляді фреймів та обміну ними за допомогою протоколу TCP/IP, що, по-перше, звужує можливе представлення знань до фреймового, по-друге, знижує відкритість системи, за рахунок специфічного способу взаємодії по мережевому протоколу.

Таким чином, перспективними є ті підходи до побудови розподілених інформаційних систем, які спрямовані на повторне використання певних складових частин. До таких підходів належать онтології та агентна парадигма створення програмного забезпечення.

Результати досліджень

Поняття агенту прийшло з теорії штучного інтелекту. Штучний інтелект базується на визначенні інтелектуального агенту як системи, яка може прийняти самостійне рішення та виконати дії, пов'язані з ним. Такі агенти отримують результати зі сприйняття свого середовища та реалізують функцію, що відображає послідовність актів сприйняття та дій [8]. Найважливішою властивістю агента у теорії штучного інтелекту вважається його здатність до навчання.

З точки зору програмної інженерії, агент – це автономна програма, яка здатна керувати власними діями у інформаційному середовищі функціонування для отримання результатів виконання поставленої задачі та зміни стану середовища [10]. Будь-який агент є відкритою системою, яка знаходиться у деякому середовищі. При цьому ця система має власну модель поведінки, яка відповідає деяким принципам.

Таким чином, агент вважається здатним сприймати інформацію із зовнішнього середовища із обмеженою роздільною здатністю, обробляти її на основі власних ресурсів, взаємодіяти з іншими агентами та діяти на середовище, переслідуючи свої цілі.

Агенти повинні мати такий мінімальний набір властивостей [10]:

– активність, здатність до організації дій;

– автономність, відносна незалежність від середовища;

– комунікативність, яка дозволяє вирішувати свої задачі разом з іншими агентами та забезпечується розвиненими протоколами комунікації;

– цілеспрямованість, яка передбачає наявність власних джерел мотивації.

Для розв'язання таких задач, які неможливо розв'язати за допомогою одного агенту або монолітної системи, можуть використовуватися багатоагентні системи (БАС). Багатоагентна система – це система, що утворена декількома взаємодіючими агентами. У БАС усі елементи обов'язково взаємодіють один з одним. Система може вважатися БАС, а не просто спільнотою агентів, тільки якщо усі агенти діють для досягнення однієї цілі. Однак, відсутність єдиної цілі у агентів не заперечує можливість їхньої групової поведінки.

Дуже важливо відмітити, що агенти, які входять до БАС, скоріше за все, не були спеціально для цього спроектовані. Це можуть бути повторно використані або розроблені для розв'язання більш універсальних задач агенти. У таких випадках агенти можуть мати власні цілі, які не повністю співпадають з цілями системи, однак сумісні з ними. Такі агенти можуть бути корисними один одному для розв'язання поставлених перед ними задач. Тому для агентів, що входять до БАС, найважливішою властивістю є комунікативність.

Для успішного функціонування агент повинен мати рецептори, ефектори, процесор та пам'ять. Рецептори – це спеціальні пристрої, які приймають інформацію із зовнішнього середовища. Ефектори – це виконавчі органи, які діють на середовище. Під пам'яттю розуміється здатність агента зберігати інформацію про свій стан та стан середовища.

Рецептори утворюють систему сприйняття агента, забезпечуючи приймання та первинну обробку інформації, що надходить із середовища. Система сприйняття може контролювати дії за рахунок порівняння розбіжностей між наявним та очікуваним станом. У пам'яті агента повинні бути дані про типові реакції на інформаційні сигнали від рецепторів, а також інформація про стан ефекторів та ресурсів. Окрім того, у пам'яті повинні зберігатися програми переробки вхідної інформації у керуючі сигнали, які подаються на ефектори, та результати реакцій на ту чи іншу ситуацію.

Для представлення можливих дій агента та його взаємодії із зовнішнім середовищем необхідно мати інструмент, який дозволяє у формальному вигляді описувати поведінку агента. Поведінка агента покладена в основу його формальної архітектури (рис. 1).

Формальна архітектура агента – це інструмент, який дозволяє проектувати поведінку агента із використанням чітких формальних методів. Формальна архітектура агента задається через опис середовища, в якому функціонує агент, сприйняття агентом цього середовища та його діями [11].

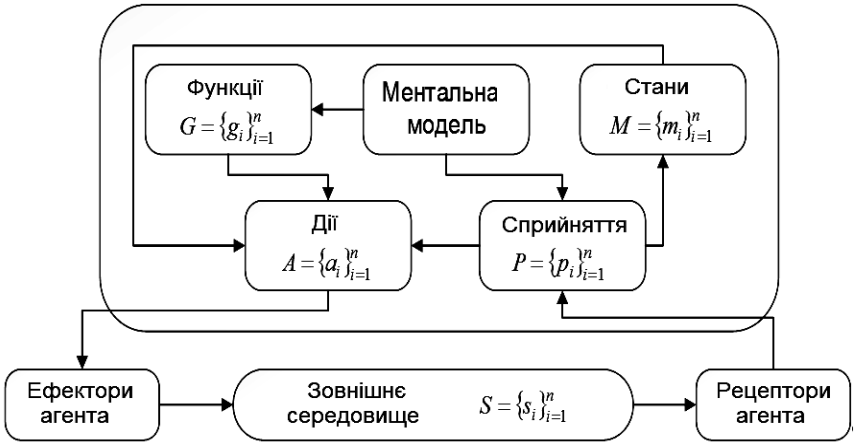

Рис. 1. Формальна архітектура агента

Позначимо зовнішнє середовище агента за допомогою множини станів S. Можливі дії агента описуються за допомогою множини дій A. Абстрактно агент може представлятися як функція

$$g_S : S \to A,$$

тобто вибір конкретної дії із множини можливих дій агент здійснює на основі поточного стану зовнішнього середовища $s_i \in S$. При цьому дії агента можуть впливати на середовище, але не контролювати його повністю.

Для представлення агента зручно використовувати модель сприйняття зовнішнього середовища. Для цього вводиться множина можливих сприйнять P та функція $f : S \to P$, яка описує, у який спосіб певні стани середовища сприймаються агентом. Тоді агент представляється за допомогою функції

$$g_P : P \to A,$$

тобто дія агента визначається у загальному випадку поточним сприйняттям стану зовнішнього середовища $p_j \in P$.

Модель агента із сприйняттям еквівалентна базовій. Проте вона дозволяє ввести наступну додаткову властивість агента: різні стани середовища можуть однаково сприйматися і навпаки – один стан може по-різному сприйматися агентом.

Іншим варіантом рішення задачі включення попередніх дій при виборі поточної дії є введення поняття стану агенту. При цьому вважається, що агент має певні внутрішні структури даних, які він модифікує в залежності від сприйняття поточного стану зовнішнього середовища, та на основі отриманих результатів обирає дію. Для формалізації цього процесу вводиться множина I внутрішніх станів агента та функція оновлення внутрішнього стану, яка відповідає за оновлення внутрішнього стану у відповідності до поточного сприйняття середовища:

$$h : I \times P \to I.$$

Тоді агент описується за допомогою функції

$$g_I : I \to A,$$

тобто дія обирається на основі поточного стану агента. Для коректного опису поведінки агента із станом необхідно визначати початковий стан i_0.

Така архітектура агента має один суттєвий недолік, а саме – агент, що задається у такий спосіб, не отримує інформацію про здійснені ним дії, що обмежує його можливості у накопиченні досвіду та аналізі потенційних наслідків його дій.

Одним із можливих способів подолання цього недоліку є представлення інформації про дії агента як частину інформації про зовнішнє середовище, проте такий підхід не є наочним та інтуїтивно зрозумілим.

Більш правильним вирішенням цієї проблеми є включення інформації про здійснювані дії явно у вхідні дані функції вибору дії:

$$g_A : (P \times A)^* \to A.$$

У такому вигляді агент явно отримує інформацію про вже здійснені дії та при виборі дії спирається на сприйняття станів навколишнього середовища.

Для агента із станом інформація про попередні дії враховується у функції оновлення стану:

$$h : I \times P \times A \to I.$$

Параметром функції оновлення стану є не послідовність усіх дій агента, а тільки остання виконана дія.

При використанні підходів, заснованих на формальній логіці, функції сприйняття та вибору дії агента описуються як набір тверджень, або правил, цієї логіки.

При цьому агент підтримує базу знань, що містить множину тверджень формальної логіки, які описують причинно-наслідкові зв'язки між станами зовнішнього середовища та сприйняттями агента, а також між сприйняттями середовища та діями агента. Такий агент називається логічним агентом.

Агент, який обирає дію на основі поточного сприйняття, ігноруючи всю історію попередніх сприйняттів, є простим рефлексним агентом. Такий тип агентів є достатньо простим. У багатьох випадках для успішного функціонування агента можуть знадобитися знання двох видів. З одного боку, це інформація про те, як середовище змінюється незалежно від агента. З іншого боку, це знання про те, як власні дії агента впливають на середовище. Агент, який використовує такі знання про існування зовнішнього середовища, є рефлексним агентом, заснованим на моделі.

Розглянемо, яким чином властивості агентів можуть бути корисними при створенні розподіленої системи в колаборативних середовищах. Здатність діяти автономно від імені користувача – властивість, яка значно економить час та зусилля користувача. Розподілені інформаційні системи повинні бути спроектовані таким чином, щоб користувач не відволікався від звичної діяльності.

Ознакою хорошого стилю є мінімальна кількість звернень системи до користувача, в ідеалі – лише при ініціалізації профілю. В великих колаборативних середовищах (до яких відносяться системи обробки бізнес-інформації та підтримки прийняття рішень) профілі користувачів можуть зберігатися на кількох серверах. Кількість користувачів та кількість даних в їхніх профілях є великою, тому надсилання агента для пошуку на інший сервер є ефективнішим, ніж копіювання всіх даних для локального використання.

Проактивність агентів – властивість, яку можна використати для заохочення спілкування та співпраці окремих користувачів. Рекомендаційна система є відкритою – тобто кількість користувачів змінюється динамічно. БАС є зручним середовищем для проектування відкритих систем, оскільки тут вже є інструменти для реєстрації нових агентів, для спілкування між агентами, сервіси, які допомагають виявити нових учасників системи.

Існує багато інструментальних рішень для проектування БАС, та найпопулярнішими серед них є агенті платформи. Агентна платформа – це проміжний виконавчий рівень, який знаходиться між агентами та операційною системою [12]. У деяких випадках вона може спиратися не на саму операційну систему, а на якусь платформу (Java Virtual Machine або .NET Framework).

У табл. 1 наведена порівняльна характеристика найбільш відомих агентних платформ.

Таблиця 1

Порівняльна характеристика агентних платформ

Критерій порівняння	JADE	Cougaar	ZEUS	Jason
Мова програмування	Java	Java	Візуальні генератори коду	AgentSpeak
FIPA-сумісність	+	-	+	-
Ліцензія на використання	Не потрібна	Не потрібна	Потрібна	Не потрібна

Для розробки прототипу рекомендаційної системи обрано платформу JADE. До основних переваг цієї платформи можна віднести: можливість інтеграції з іншими системами, підтримку стандартів специфікації FIPA-2000 та вільне розповсюдження.

Платформа розробки мультиагентних систем JADE включає в себе динамічне середовище, де можуть функціонувати JADE агенти; бібліотеку класів, яку програмісти можуть використовувати для розробки власних агентів; набір графічних інструментів, що дозволяють управляти активністю запущених агентів.

Платформа JADE є розподіленою та представляє собою набір контейнерів (рис. 2).

Контейнером називається динамічне середовище виконання мультиагентних додатків, в якому знаходяться агенти. Кожен контейнер може містити кілька агентів. Набір активних контейнерів називається платформою. Один з контейнерів завжди є головним (Main container), всі інші контейнери зв'язуються з ним і реєструються в момент запуску. Тому першим контейнером при старті платформи має бути головний, а всі інші контейнери мають бути звичайними і повинні заздалегідь знати, на якому контейнері вони будуть реєструватися, тобто повинні мати дані про хости і порти.

Структурно JADE складається з основних пакетів [12]:

– ядро системи, до якого вбудований клас Agent (надає базову функціональність агентам);

– пакет, який визначає базову поведінку агентів;

– пакет, який визначає засоби взаємодії агентів;

– пакет, який надає базові засоби для створення графічного інтерфейсу агента;

– пакет, який забезпечує різні варіанти життєвого циклу агента.

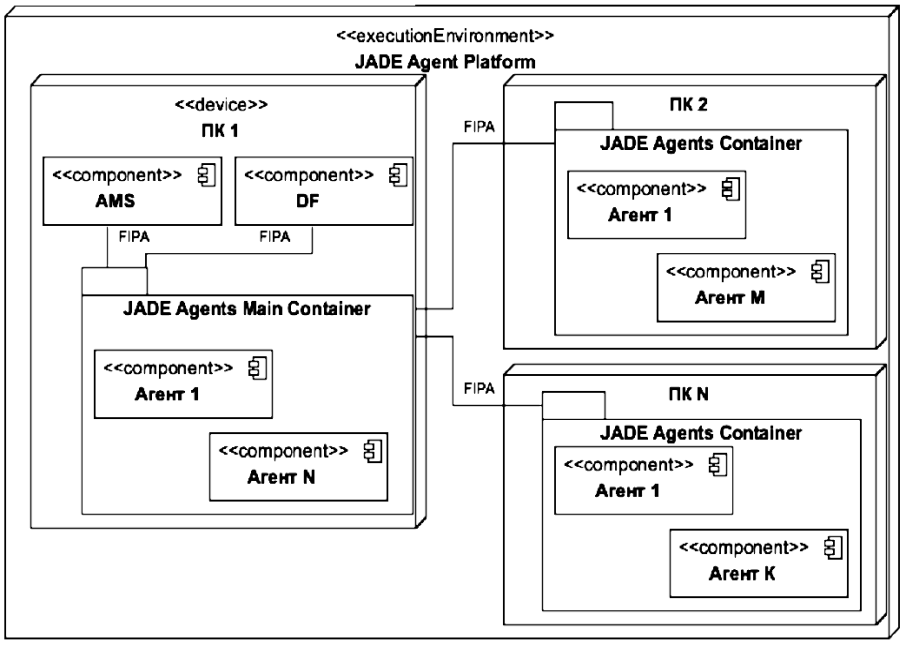

Рис. 2. Платформа JADE

Для реалізації інформаційної системи пропонується використовувати дві технології зберігання даних: базу даних та локальну кеш-пам'ять. База даних необхідна для тривалого зберігання даних. Локальна кеш-пам'ять застосовується з метою зменшення часу, необхідного для надання відповіді на запити агентів. В якості засобу управління кеш-пам'яттю пропонується використовувати технологію Ehcache, яка представляє собою універсальну систему розподіленого кешування для Java додатків і сервлетів [13]. Це невеликий додаток, який реалізує як динамічний кеш (в оперативній пам'яті), так і дисковий кеш, причому спроектований для роботи з кешами великого обсягу (порядку гігабайтів) в розподілених системах.

Ehcache може реалізовувати кілька різних стратегій кешування, наприклад, LFR (кешування виходячи з частоти використання) або FIFO, а також можна контролювати кеш на рівні окремих об'єктів (задаючи схему інвалідації об'єктів в кеші – по часу життя або часу простою). Ehcache підтримує стандартний протокол роботи з кешем JSR107 JCACHE.

Однією з унікальних функцій Ehcache є «персистентний кеш», який дозволяє зберігати стан об'єктів навіть після перезавантаження віртуальної java-машини. Крім цього, система кешування підтримує консоль управління JMX, що дозволяє прозоро інтегрувати її в систему управління будь-яким додатком, використовуючи тільки стандартні протоколи і можливості.

Розподілене кешування може застосовуватися для забезпечення роботи на кластері або в інших розподілених схемах, використовуючи вбудований (з версії 1.2) механізм RMI на основі протоколу TCP. Синхронізація та/або реплікація кешей між вузлами може бути як загальною, так і локальною, для окремих кешей за своєю схемою, асинхронною або синхронною. Для більшої функціональності програми, що використовує кеш, абсолютно нічого не потрібно, все залежить тільки від одного конфігураційного XML файлу.

Для представлення знань, необхідних агентам для функціонування, у роботі пропонується використовувати онтології, створені у на платформі Protégé. Protégé – це вільно поширювана Java-програма, призначена для побудови (створення, редагування та перегляду) онтологій прикладної області [14]. Вона включає редактор онтології, що дозволяє проектувати онтології, розгортаючи ієрархічну структуру абстрактних і конкретних класів і слотів. На основі сформованої онтології Protégé можна генерувати форми отримання знань для введення екземплярів класів і підкласів. Даний інструмент підтримує використання мови OWL і дозволяє генерувати html-документи, що відображають структуру онтологій. Оскільки він використовує фреймову модель подання знань, це дозволяє адаптувати його і для редагування моделей предметної області, представлених не на мові OWL (Web Ontology Language), а в інших форматах (UML, XML, SHOE, DAML + OIL, RDF і RDFS і т. п.).

Агентна платформа Jade підтримує інтеграцію із платформою Protégé та дозволяє імпортувати агентам необхідні онтології у форматі Protégé-OWL. Отже, реалізація інформаційної системи обробки бізнес-інформації на базі агентної платформи JADE та із використанням технології управління кеш-пам'яттю Ehcache, що дозволяють виконати усі системні вимоги щодо здатності до перенесення, супроводження та ефективності.

Висновки

Таким чином, в результаті проведеного дослідження обрано методологію, інструментальні засоби та технології створення розподіленої інформаційної системи для обробки бізнес-інформації.

Використання агентної парадигми з урахуванням формальної архітектури надає можливості побудови колаборативного середовища, що забезпечить ефективну обробку, зберігання та використання бізнес-інформації.

Обчислювальна модель агента є багатозадачною. Окремі задачі або моделі поведінки агента можуть виконуватися одночасно. Кожна функціональна можливість або сервіс, що представляється агентом, повинні бути реалізовані як один варіант поведінки або їх набір.

Такий підхід дозволяє конфігурувати виділення потоків виконання для агентів відповідно до різних варіантів: один потік на всіх агентів, виділення одного потоку на агента, фіксований пул робочих потоків, більший або менший, ніж кількість агентів у системі.

Подальше дослідження буде присвячено реалізації прототипу інформаційної системи та аналізу ефективності пропонованих рішень.

Список літератури

1. Turban E. Decision support and expert systems: management support systems / E. Turban. – Englewood Cliffs, N.J.: Prentice Hall, 1995. – 887 p.

2. Чередниченко О.Ю. Мозгоподобные структуры для сбора и автоматизированной переработки бизнес-информации / О.Ю. Чередниченко, О.В. Янголенко, Ю.Н. Гонтарь // International Collection of scientific proceedings «European Cooperation». – 2015. – Vol. 2(2). – P. 125-136.

3. Decision Support Systems: Issues and Challenges / Ed. by G. Fick and R. H. Sprague. – Oxford: Pergamon Press, 1980. – 189 p.

4. Alkhateeb F. Multi-Agent Systems – Modeling, Interactions, Simulations and Case Studies / F. Alkhateeb, E. Al Maghayreh, I. Abu Doush. – InTech, 2011. – 512 p.

5. Основы инженерии качества программных систем / Ф.И. Андон, Г.И. Коваль, Т.М. Коротун и др. – К.: Издательский дом «Академпериодика», 2007. – 672 с.

6. Гороховський С.С. Агентні технології: спроба критичного огляду / Г.Г. Гороховський // Наукові записки. Т. 18: Спеціальний випуск. – К.: Національний університет "Києво-Могилянська Академія", 2000. – С. 391-395.

7. Соммервил И. Инженерия программного обеспечения / И. Соммервил. – М.: Издательский дом «Вильямс», 2002. – 624 с.

8. Kshemkalyani A.D. Distributed Computing: Principles, Algorithms, and Systems / A.D. Kshemkalyani, M. Singhal. – Cambridge University Press, 2008. – 736 p.

9. Рассел С. Искусственный интеллект, современный подход / С. Рассел, П. Норвиг. – М.: Вильямс, 2007. – 1410 с.

10. Henderson-Seller B. Agent-Oriented Methodologies / B. Henderson-Seller, P. Giorgini. – London: Idea Group Publishing, 2005. – 413 p.

11. Wooldridge M.J. An introduction to multiagent systems / M.J. Wooldridge. – John Wiley & Sons, LTD, 2009. – 461 p.

12. Bellifemine F. Developing Multi-Agent Systems with JADE / F. Bellifemine, G. Caire, D. Greenwood. – John Wiley & Sons Ltd, The Atrium, 2007. – 286 p.

13. Ehcache operations guide [Електронний ресурс]. – Режим доступу до ресурсу: http://ehcache.org/generated/2.9.0/pdf/Ehcache_Operations_Guide.pdf.

14. Calero C. Ontologies for Software Engineering and Software Technology / C. Calero, F. Ruiz, M. Piattini. – Berlin: Springer, 2006. – 339 p.

Надійшла до редколегії 19.02.2016

Рецензент: д-р техн. наук, проф. М.Д. Годлевський, Національний технічний університет «Харківський політехнічний інститут», Харків.

РАЗРАБОТКА РАСПРЕДЕЛЕННОЙ СИСТЕМЫ ОБРАБОТКИ БИЗНЕС-ИНФОРМАЦИИ С ИСПОЛЬЗОВАНИЕМ АГЕНТНОГО ПОДХОДА

Ю.Н. Гонтарь, О.Ю. Чередниченко, О.В. Янголенко, М.А. Вовк

В статье рассмотрен вопрос информационной системы для обработки информации, на которой основываются бизнес-процессы предприятия. Исходя из больших объемов данных, которые необходимо анализировать, и принимая во внимание коллаборативную природу рекомендационных систем, обоснован выбор распределенной архитектуры. Для реализации информационной системы предложено использовать агентный подход. Разработана формальная модель архитектуры агента. Проанализированы возможности агентных платформ и технологий хранения данных для создания многоагентной системы обработки бизнес-информации.

Ключевые слова: *распределенная информационная система, бизнес-информация, агентный подход, агентная платформа.*

DEVELOPMENT OF DISTRIBUTED SYSTEM OF BUSINESS-INFORMATION PROCESSING BASED ON THE AGENT APPROACH

Yu.M. Gontar, O.Yu. Cherednichenko, O.V. Yanholenko, M.A. Vovk

The problem of information system development for processing of enterprise business-information is considered in the given paper. The choice of distributed architecture is grounded based on the big volumes of data to be processed and based on the collaborative nature of recommendation systems. It is suggested to use agent approach for information system realization. The formal model of agent architecture is developed. The capabilities of agent platforms and data storage technologies are analyzed with respect to the problem of development of multiagent system of business-information processing.

Keywords: *distributed information system, business-information, agent approach, agent platform.*