УДК 004.4’2

Глибовець А.М., Гороховський С.С., Піка А.А.

АГЕНТИ ДЛЯ РЕКОМЕНДАЦІЙ У КОЛАБОРАТИВНИХ СЕРЕДОВИЩАХ

У роботі проаналізовано технології створення рекомендаційних систем. Для ефективного розв’язання задач дизайну рекомендаційних систем у колаборативних середовищах досліджено особливості застосування агентів. Запропонована агентна архітектура системи для рекомендації вибіркових курсів. Описано створений прототип рекомендаційної системи.

Ключові слова: *рекомендаційні системи, колаборативні середовища, агентна архітектура.*

В работе проанализированы технологии создания рекомендационных систем. Для эффективного решения задач дизайна рекомендационных систем в колаборативных средах изучены особенности применения агентов. Предложена агентная архитектура системы для рекомендации выборных курсов. Описан созданый прототип рекомендационной системы.

Ключевые слова: *рекомендационные системы, колаборативные среды, агентная архитектура.*

In this work technologies which are used for recommender systems creation are analyzed. Peculiarities of using software agents for efficient recommender systems in collaborative environments design problem solving are researched. New agent architecture of recommender system for optional courses recommendation is proposed. Recommender system prototype created is described.

Key words: *recommender system, collaborative environments, agent architecture.*

ВСТУП

Значне зростання кількості інформації зумовило загострення проблеми пошуку корисної інформації в останній час. Потрібна технологія, яка б допомогла ефективно звузити коло можливих варіантів. Ідея отримання рекомендацій від досвідченіших користувачів є основою систем, що продукують рекомендації в колаборативних середовищах (КС), метою яких є підтримка співробітництва.

Рекомендаційні системи (РС) – технологія, яка активно розвивається. Уперше випробувані в середовищах електронної комерції, сьогодні вони використовуються в різних сферах, зокрема зарекомендували себе добре в системах електронної освіти.

При проектуванні програмних систем для рекомендацій у КС виникає ряд важливих задач, які потрібно розв’язати. Зокрема, простір інформації може бути дуже великим та змінюватись динамічно. Потрібно відфільтрувати значну кількість альтернатив та запропонувати користувачу тільки найкращі. Запит може бути сформульованим користувачем нечітко, оскільки він не є експертом у тій галузі знань, стосовно якої він хоче отримати рекомендації. Це виключає використання традиційних пошукових систем.

КС є відкритим та динамічним. Учасники можуть покидати його, нові члени можуть приєднуватись до групи в будь-який момент. Це середовище може бути гетерогенним – його учасники є експертами з різних галузей знань. Для вирішення багатьох із цих задач бачиться ефективне використання агентного підходу.

Метою роботи є дослідження задач, які покликані розв’язати РС, проблем, які постають перед розробниками у процесі проектування РС у КС, та їхнього вирішення із застосуванням агентного підходу.

1. РЕКОМЕНДАЦІЙНІ СИСТЕМИ

Рекомендаційна система – система, яка надає рекомендації користувачу [1]. Таке узагальнене визначення РС, яке зустрічається найчастіше в літературі, свідчить про досить широке коло різних систем, які потрапляють у цю категорію. Більш детальні специфікації РС зазвичай указують при описі конкретного способу її реалізації [2-4]. Спробуємо визначити певні спільні для всіх визначень риси та дамо власне визначення РС.

У будь-якому випадку маємо справу з користувачем, у якого багато альтернатив, серед яких потрібно зробити вибір. Користувачу може бракувати знань, щоб самостійно відкинути менш корисні альтернативи. Користувач також у певній формі (експліцитно чи імпліцитно) постачає системі інформацію про свою потребу. Отже, РС – система, яка використовує наявну інформацію про потребу користувача та певний алгоритм фільтрування, рекомендує користувачу набір альтернатив, які вона вважає найбільш корисними для задоволення цієї потреби.

Традиційно виділяють два типи фільтрування – за вмістом та колаборативне фільтрування. Деякі автори окремим типом класифікують гібридне фільтрування, яке поєднує риси двох попередніх [1; 3; 5].

У рекомендаційній системі, яка використовує фільтрування за вмістом, користувачі є незалежними від інших користувачів системи. Для генерування рекомендацій система потребує профіль користувача, у якому відображені його зацікавлення. У профілі в певній формі (яка залежить від обраного способу представлення та обраних алгоритмів) зберігається інформація про поняття (властивості у вигляді атрибутів), якими цікавиться користувач. Крім того, система має інформацію про всі поняття, які вона може рекомендувати. Беручи за основу опис понять із профілю користувача, система знаходить у власній базі даних схожі поняття, які й рекомендуються користувачу. Такий підхід корисний користувачам із чітко визначеними, специфічними зацікавленнями, які шукають схожі рекомендації.

Перевагою фільтрування за вмістом є той факт, що рекомендації не залежать від інших користувачів системи, тому система не потребує критичної маси користувачів для початку генерування рекомендацій.

Головне обмеження цієї технології – нездатність рекомендувати нові поняття, які не випливають безпосередньо із зацікавлень, що вказані у профілі користувача. Тобто, користувач завжди буде отримувати рекомендації тих понять, які дуже схожі на поняття з його профілю. Це приводить до завеликої спеціалізації, що знижує корисність рекомендацій для користувача.

Для збереження високої якості РС повинна отримувати зворотній зв’язок від користувача про релевантність наданих рекомендацій. Ця інформація їй потрібна для поновлення профілю користувача. Тут виникає інша проблема – користувачі зазвичай неохочі постачати інформацію для зворотного зв’язку системі та прагнуть уникнути цього (типова проблема для всіх рекомендаційних систем). А оскільки рекомендації базуються цілком на інформації із профілю користувача, то чим менше відомостей отримано від користувача, тим менш корисний набір рекомендацій, які йому запропонує система. Для вирішення цієї проблеми використовують різні техніки автоматичного навчання для підтримки профілю. Проте реалізація їх є складною та не у всіх ситуаціях ефективною.

Використання РС із фільтруванням за вмістом є найбільш уживаним у системах електронної комерції. Цей підхід використовують також у поєднанні з колаборативним фільтруванням для отримання більш цінних рекомендацій в інших сферах – електронна освіта, програмування тощо.

РС із колаборативним фільтруванням знаходять користувачів зі схожими інтересами та рекомендують ті предмети, які високо оцінені такими користувачами. Цей процес дуже схожий на те, як ми просимо поради у друзів чи колег, тому його ще іноді називають соціальним фільтруванням. Зацікавлення окремих користувачів зберігаються у вигляді рейтингів, які показують наскільки користувачу подобається чи не подобається певний предмет. Таким

системам не потрібно мати додаткової інформації про цей предмет – лише оцінка користувача. Очевидна перевага в порівнянні з фільтруванням за вмістом – рекомендації не генеруються системою, виходячи лише із зацікавлень одного користувача.

РС із колаборативним фільтруванням залежать значною мірою від визначення «схожості» користувачів. Цю міру «схожості» потрібно визначити певним чином. Для її визначення РС використовують різні метрики – коефіцієнти Пірсона, векторна схожість, байєсівська імовірнісна класифікація [1].

Серед найпоширеніших алгоритмів для визначення схожості використовують алгоритми «найближчого сусіда». Типовий процес продукування рекомендацій із використанням таких алгоритмів містить такі кроки:

- Обчислення схожості між активним користувачем та іншими користувачами системи, використовуючи рейтинги понять із профілів користувачів.
- Вибір групи користувачів, профілі яких стануть базисом для генерування рекомендацій. Ця група визначається за рівнем схожості з активним користувачем. Принципи формування групи користувачів відрізняються в різних системах – одні вибирають фіксовану кількість користувачів, інші – тільки тих, у яких рівень схожості вищий за певну встановлену межу тощо.
- Використовуючи профілі користувачів із групи, яку було сформовано на попередньому кроці, система формує рекомендації. Деякі системи при формуванні рекомендацій ураховують не лише рейтинг предмета, але й рівень схожості користувача, який цей рейтинг установив з активним користувачем.

Одна з головних задач у системах із колаборативним фільтруванням – це задача «холодного початку» [1; 5]. Система не здатна ефективно генерувати рекомендації, якщо достатня кількість користувачів не вкажуть своїх інтересів у профілі. Інший важливий момент, який треба врахувати при проектуванні таких систем – це можливість масштабованості. Для генерування корисних рекомендацій у системі повинно бути багато користувачів, а їхні профілі повинні містити якнайбільше рейтингів понять, але з ростом системи зростає й час реакції системи на запит. Крім того, слід забезпечити механізми захисту приватності користувачів. Це складна задача, оскільки чим менше даних про користувача доступно, тим менш корисні рекомендації генерує система.

Гібридні системи зазвичай комбінують фільтрування за вмістом та колаборативне фільтрування, що дає змогу вирішити деякі проблеми, які присутні, коли використовувати ці підходи окремо.

Розглянемо, яким же чином поєднання цих двох підходів може принести позитивний результат. Недоліком систем із фільтруванням за вмістом є їхній ретроспективний характер – нездатність порекомендувати нові предмети. Колаборативне фільтрування цю проблему вирішує – рекомендації, згенеровані на основі профілів інших користувачів, не обов'язково схожі на поняття, присутні в профілі активного користувача.

Система з колаборативним фільтруванням не дає корисних рекомендацій користувачам, у яких нетипові, оригінальні смаки, оскільки бракує схожих користувачів для генерування рекомендацій. У таких випадках ефективніша фільтрація за вмістом. Крім того, фільтрація за вмістом допомагає вирішити проблему «холодного початку» – поки не буде достатньої кількості користувачів, використовується фільтрування за вмістом, коли ж критична маса користувачів досягнута, у системі починають також працювати методи колаборативного фільтрування.

У гібридній системі інформація про зацікавлення користувачів представлена в профілі у двох видах – як набір атрибутів певного предмета та як його оцінка користувачем, що є одночасно й перевагою й недоліком системи. Перевага полягає в тому, що більша інформованість дає можливість використовувати ефективніші алгоритми фільтрування та генерувати корисніші рекомендації. Недолік – користувачам потрібно ввести більше інформації, що як ми вже знаємо, вони роблять неохоче.

Поєднання фільтрування за вмістом та колаборативного фільтрування не вирішує проблем масштабованості системи та захисту приватності користувача.

2. ВИКОРИСТАННЯ АГЕНТІВ У РС ІЗ КОЛАБОРАТИВНИМ ФІЛЬТРУВАННЯМ

У цій роботі під агентом ми будемо розуміти наступне. Агент – програмна сутність, яка автономно виконується від імені інших сутностей (інших програм, користувачів); здійснює певні дії проактивно і/або реактивно; має здатність навчатись, спілкуватись з іншими такими сутностями та мігрувати [6].

Розглянемо конкретніше, які властивості агентів можуть бути корисними при створенні РС у колаборативних середовищах.

Здатність діяти автономно від імені користувача – властивість, яка значно економить час та зусилля користувача. РС повинні бути спроектовані, таким чином, щоб користувач не відволікався від звичної основної діяльності. Ознакою хорошого стилю є мінімальна кількість звернень системи до користувача, в ідеалі – лише при ініціалізації профілю [5].

У великих колаборативних середовищах (а саме в таких працюють РС із колаборативним фільтруванням) профілі користувачів можуть зберігатися на кількох серверах. Кількість користувачів та кількість даних у їхніх профілях є великою, тому надсилання агента для пошуку на інший сервер є ефективнішим, ніж копіювання всіх даних для локального використання. Під ефективністю ми розуміємо: а) раціональніше використання обчислювальних ресурсів – обчислювальна робота розподілена між декількома серверами; б) зменшення навантаження на мережу – немає необхідності копіювати великі обсяги даних; в) підвищення швидкодії – міграція агента на іншу платформу потребує менше часу, ніж копіювання великих обсягів даних.

При централізованому зберіганні профілів користувачів на одному сервері проблема збереження приватності даних стоїть особливо гостро. Користувачі не бажають надавати відомості про себе, особливо це стосується сфери електронної комерції, відповідно система не може давати корисних рекомендацій. Можна запропонувати користувачам зберігати свій профіль на локальній машині, агент, який діє від імені такого користувача, контролює доступ до даних, доступ до даних інших користувачів здійснюється через спілкування з іншими агентами. При такій реалізації може знизитись продуктивність системи. Але такий підхід дає більший контроль користувачу над його профілем.

Проактивність агентів – властивість, яку можна використати для заохочення спілкування та співпраці окремих користувачів. Прикладом ситуації, де це може бути корисно, є системи електронної освіти. Агент може здійснювати пошук студентів, які працюють над тією ж темою, що й активний користувач, і рекомендувати їх у якості помічників для сприяння колаборативному навчанню.

РС є відкритою – тобто кількість користувачів змінюється динамічно. Мультиагентна система є зручним середовищем для проектування відкритих систем, оскільки тут уже є інструменти для реєстрації нових агентів, для спілкування між агентами, сервіси, які допомагають виявити нових учасників системи.

Серед рекомендацій РС агентного типу можна виділити: RASCAL, CALM, KARE, TV-Recommender, IC-Service.

3. РОЗРОБКА АРХІТЕКТУРИ ТА СТВОРЕННЯ ПРОТОТИПУ РС

Розроблену архітектуру рекомендаційної системи продемонструємо на прикладі системи для рекомендування вибіркових курсів. Головні компоненти архітектури зображені на рисунку 1.

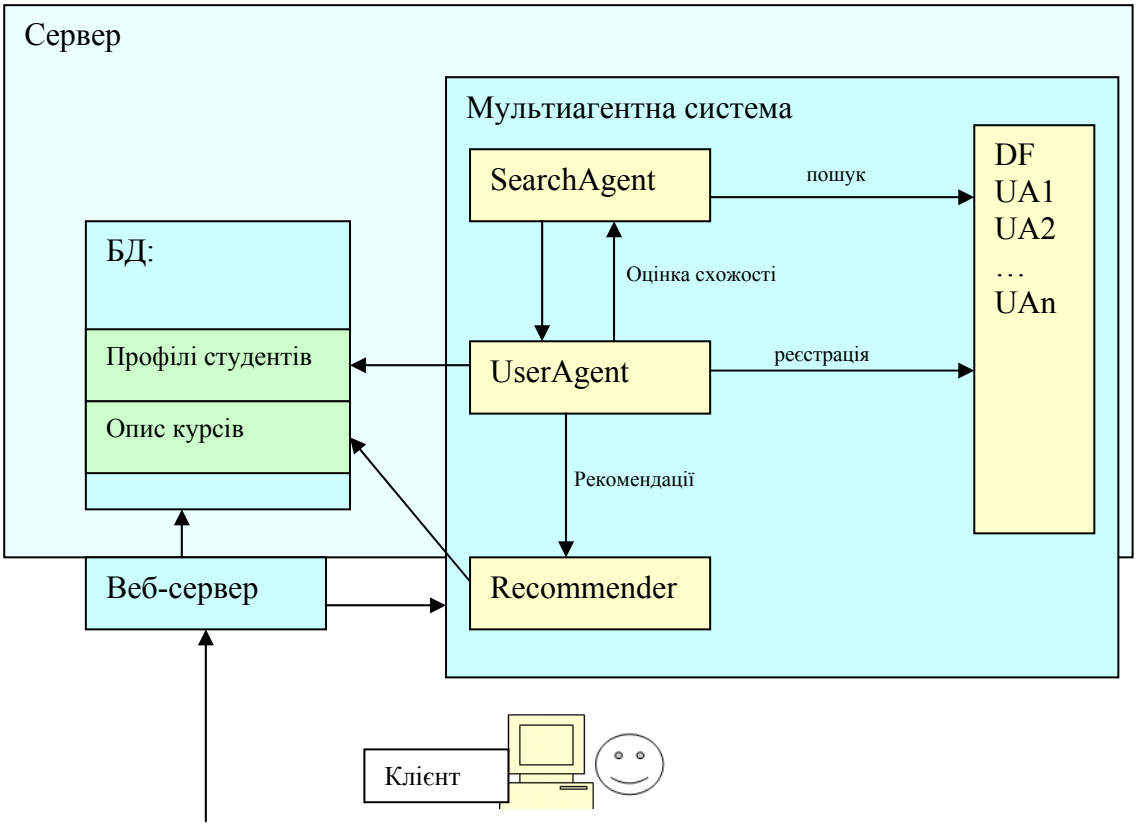

Рис. 1. Архітектура рекомендаційної системи

Як видно з рисунка, серверна частина складається з веб-сервера, мультиагентної системи та бази даних, яка містить описи курсів та профілі студентів. Клієнту для доступу до системи не потрібне додаткове програмне забезпечення, він, користуючись будь-яким веб-браузером, здійснює запити до веб-сервера. Веб-сервер виконує роль посередника, який перенаправляє запит клієнта мультиагентній системі, отримує від неї рекомендації та формує веб-сторінку з рекомендаціями для клієнта.

Основна функціональність системи зосереджена у мультиагентній системі та виконується такими трьома типами агентів.

UserAgent – персональний агент, який створюється під час реєстрації користувача в системі та існує до закриття користувачем свого профілю. При створенні агент реєструється, використовуючи сервіс жовтих сторінок, де вказує факультет та рік навчання. Ці відомості використовуються пізніше агентом SearchAgent для пошуку персональних агентів потрібних студентів. UserAgent має доступ до персонального профілю користувача. Він отримує від агента SearchAgent відомості про профіль іншого користувача, який здійснює пошук, та обчислює схожість цього профілю та профілю власного користувача, надсилає результат агенту SearchAgent. Другий тип запиту, який він обробляє – це запит на формування рекомендації вибіркових курсів за вказаними критеріями. Сформовані рекомендації надсилає агенту Recommender відповідного користувача.

SearchAgent – створюється для одноразової роботи при отриманні запиту від користувача про рекомендацію. Цей агент шукає персональних агентів студентів відповідного факультету та вказаних років навчання, і надсилає їм запит про обчислення схожості профілів студента, що просить рекомендацію, та відповідних студентів, чиї інтереси вони представляють. Отримавши від них обчислені коефіцієнти схожості, SearchAgent формує запит про рекомендацію та надсилає персональним агентам студентів з указаним коефіцієнтом схожості. На цьому його роль закінчується, і він припиняє своє існування.

Recommender – агент, який також створюється одноразово та існує до завершення сеансу отримання рекомендацій користувачем. Він отримує рекомендації від персональних агентів інших студентів. Він має також доступ до бази з описом курсів та формує набір курсів, що відповідають критерію пошуку користувача. Третій тип рекомендацій, який ним формується – це рекомендації курсів відповідно до прослуханих користувачем. Для формування такої рекомендації використовується база даних з описом зв'язків між курсами.

Два останні типи агентів також створюються для кожного користувача при ініціалізації ним запиту про рекомендацію. Як бачимо, мультиагентна система повністю розподілена, тут не використовується жоден компонент, який би виконував роль диспетчера та потенційно міг би стати вузьким місцем у системі. Для збільшення продуктивності системи використовуються кілька рівнів паралелізму. Перевагою розподілу функцій персонального агента користувача між трьома агентами є можливість ефективної паралельної підтримки сеансу із власним користувачем та обробки запитів інших користувачів. Для підвищення продуктивності обробки запиту користувача прийнято рішення здійснювати запит до персональних агентів інших користувачів про обчислення схожості профілів. Таким чином, ця операція виконується паралельно, кожен агент обчислює лише один коефіцієнт схожості. Ще один вид паралелізму в системі – формування агентом Recommender рекомендацій відповідно до прослуханих курсів та за ключовим словом паралельно з роботою інших агентів по формуванню рекомендацій студентів.

Профілі студентів зберігаються у вигляді окремих XML-документів, таким чином, відсутність централізованої бази даних студентських профілів є спробою уникнути ще одного можливого вузького місця у системі.

В усій РС єдиним централізованим компонентом є веб-сервер. Проте він не виконує жодної обчислювальної роботи, а лише перенаправляє запити та формує вивід, тому також не повинен знижувати загальну продуктивність системи.

3.1. Профіль

Профіль кожного студента зберігається як окремий XML-документ. Профіль містить загальні відомості – ім'я користувача в системі, пароль доступу, факультет, рік навчання. Користувачі є анонімними, але за бажанням вони можуть указати будь-які контактні дані, які потім будуть відображатись разом із рекомендованими ними курсами.

Друга частина профілю містить набір оцінених користувачем курсів, для кожного курсу вказаний його ідентифікаційний номер, оцінки користувача за критеріями та обчислена сумарна оцінка.

Для формування профілю користувач повинен експліцитно ввести всі дані. Він може в будь-який момент додавати оцінки нових курсів. Користувач може оцінювати як вибіркові, так і обов'язкові курси.

В окремому XML-документі зберігаються такі відомості про курси: назва курсу, рік викладання, імена викладачів, які читають курси, тип курсу (обов'язковий чи вибірковий), короткий опис курсу. Ще один XML-документ містить інформацію про зв'язки між курсами, яка використовується для рекомендування курсів відповідно до тих, які вже прослухані студентом. Право змінювати інформацію про курси має тільки адміністратор.

Таким чином, створення та підтримка профілю користувача повністю покладена на студента, а підтримка бази даних про дисципліни на адміністратора системи. Це не дуже зручно, але поки це єдиний спосіб дізнатись оцінку студентом прослуханих курсів. Одним із можливих варіантів розвитку системи є її інтеграція із системою електронної освіти, дослідження можливості альтернативних способів отримання інформації та впровадження.

3.2. Оцінка схожості студентів

Для оцінки схожості студентів використовується модель векторного простору. Зацікавлення студента представлені як вектор, що складається з оцінок прослуханих ним курсів. Оскільки

порівнюються студенти різних років навчання, то для формування векторів використовуються оцінки тільки тих предметів, які студенти могли прослухати від початку навчання до року навчання молодшого студента. Предмети, не оцінені користувачами, позначаються 0. Наприклад, зацікавлення двох студентів можуть бути представлені у вигляді таких векторів: $v_1 = (0.2, 0.5, 0, 1)$, $v_2 = (0, 0.8, 0.2, 0.7)$, де кожна позиція містить оцінку студента того самого курсу. Перший студент не прослухав або не оцінив курсу № 3, а другий – курсу № 1.

Схожість двох студентів оцінюється як косинус кута між векторами, що їх представляють, з використанням класичної формули.

Модель векторного простору – популярна техніка, яка давно використовується для представлення документів у різних технологіях інформаційного пошуку [7], зараз її успішно застосовують рекомендаційні системи для представлення характеристик користувачів або предметів, ця модель використовується в усіх проаналізованих системах. Вибір векторної схожості для обчислення схожості студентів зумовлений тим, що саме ця техніка продемонструвала найкращі результати при проведенні серії експериментів [8, 9], тобто за відгуками користувачів, рекомендаційна система продукувала більший відсоток корисних рекомендацій, порівняно з іншими техніками.

3.3. Алгоритм рекомендування

Система починає роботу за запитом користувача. Користувач указує такі критерії для формування рекомендацій: на який рік навчання отримати рекомендацію, ключовий термін для пошуку, отримати рекомендації від студентів, випускників чи обох, мінімальна середня оцінка курсу, який можна рекомендувати, мінімальний коефіцієнт схожості зі студентами, від яких можна отримувати рекомендації, а також може обрати режим фільтрування за викладачем. Він може не вказувати жодних параметрів, тоді використовуються значення за замовчуванням. Веб-сервер надсилає запит мультиагентній системі. У системі створюється агент *SearchAgent*, який здійснює пошук персональних агентів студентів того самого факультету відповідних років навчання, використовуючи сервіс жовтих сторінок агентної платформи, та надсилає їм профіль студента, і запит обчислити схожість. Крім того, створюється агент *Recommender*, який паралельно формує набір курсів, в описах яких зустрічається вказаний користувачем термін для пошуку, та набір рекомендацій за попередньо прослуханими курсами. Агенти *UserAgent* обчислюють схожість та надсилають її агенту *SearchAgent*, який формує набір тих користувачів, коефіцієнт схожості з якими вищий, ніж заданий, та надсилає їм запит про рекомендацію. На цьому робота агента *SearchAgent* завершується, і він ліквідовується. Агенти *UserAgent* відповідно до вказаних у запиті критеріїв формують рекомендації та надсилають агенту *Recommender*. На цьому роль агентів *UserAgent* завершується, але вони продовжують функціонувати в системі та обробляти інші запити.

Агент *Recommender* отримує рекомендації від персональних агентів інших студентів та формує з них єдиний набір рекомендацій. Потім він знаходить перетин множини рекомендацій від студентів та множини курсів, в описах яких знайдено ключове слово, та надсилає впорядковані рекомендації веб-серверу, який формує вивід для користувача. Користувач може запросити докладнішу інформацію про певний курс, тоді веб-сервер надсилає запит агенту *Recommender*, той формує відповідь, яку веб-сервер надсилає користувачу. Сеанс закінчується, коли користувач натискає кнопку «новий запит» або «вихід», тоді агент *Recommender* знищується. На рисунку 2 зображена діаграма послідовностей, що ілюструє послідовність описаних подій у системі.

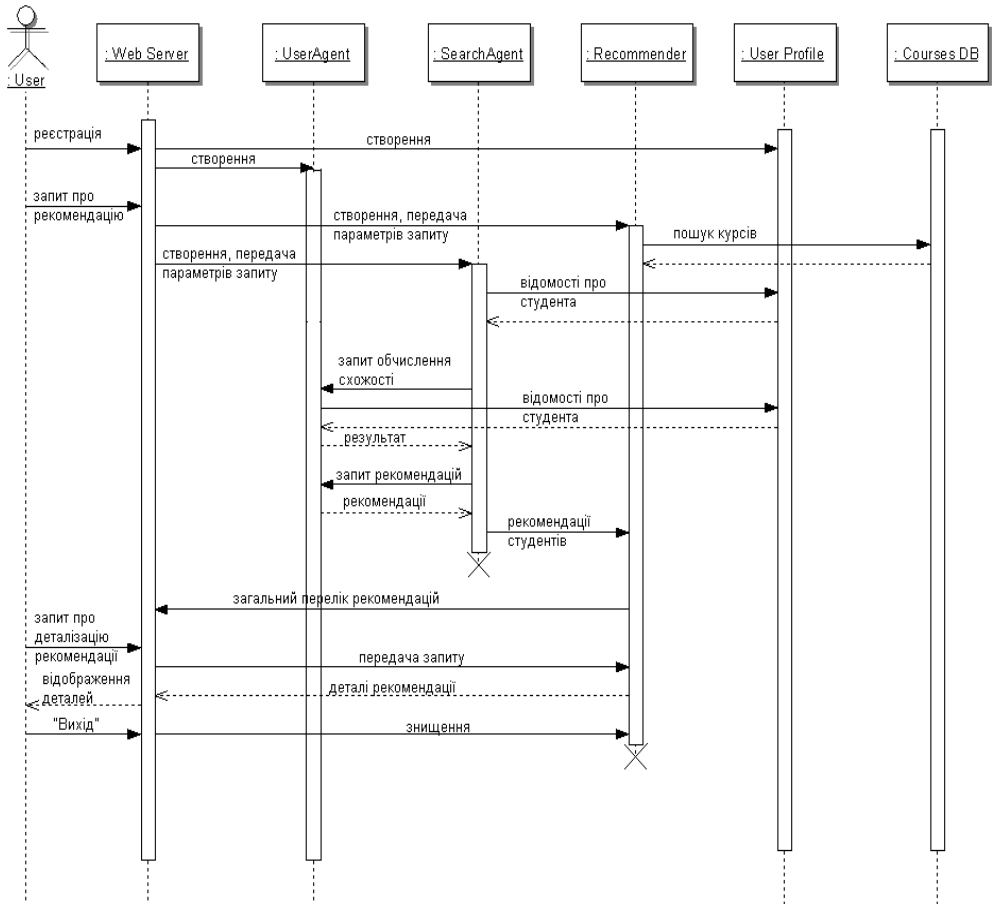

Рис. 2. Діаграма послідовностей

Отже, використаний алгоритм гібридного фільтрування – набір рекомендацій формується із множини рекомендацій від схожих студентів та множини рекомендацій відповідно до прослуханих дисциплін, а також додана можливість пошуку за ключовим словом.

3.4. Опис функціональних можливостей прототипу рекомендаційної системи

Взаємодія користувачів із рекомендаційною системою відбувається через веб-інтерфейс. У системі є два типи користувачів – студент та адміністратор.

Студент після реєстрації може оцінювати прослухані курси, або надсилати запит про рекомендацію. На рисунку 3 зображений приклад параметрів для пошуку, які може ввести студент.

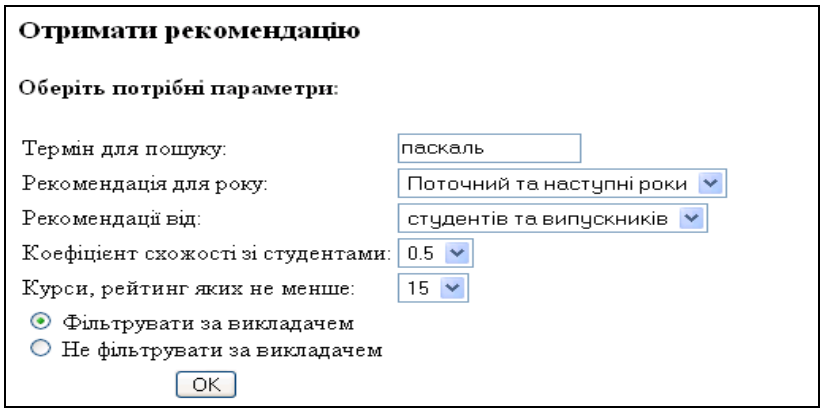

Рис. 3. Параметри рекомендації

Результатом такого запиту є набір рекомендацій. Спочатку відображаються рекомендації студентів, які відповідають заданому ключовому слову, а потім окремо рекомендації з кожної категорії, які не потрапили в перетин. Клікнувши по назві курсу, можна побачити докладніші відомості про курс.

Студент обирає курс для оцінювання серед тих курсів, які викладаються на його факультеті, при цьому, курси, оцінені студентом раніше, не відображаються. Оцінюється курс за вибраними критеріями. Студент може також указати викладача, який читав курс.

Адміністратор має можливість додавати та видаляти описи нових курсів, а також додавати та вилучати зв'язки між курсами.

При додаванні зв'язку відбувається перевірка на існування такого ж або оберненого зв'язку в системі. Якщо зв'язок знайдено, система забороняє додавати аналогічний та відображає відповідне повідомлення. При видаленні курсу видаляються також усі зв'язки із цим курсом.

3.5. Використані технології

Для створення мультиагентної системи використана агентна платформа JADE 3.6. База даних зі студентськими профілями та описами курсів зберігається у вигляді XML-документів. Представлення результатів користувачу здійснюється з використанням технології jsp. В якості веб-сервера використано Apache Tomcat 4.1.

JADE (Java Agent DEvelopment Framework) – агентна платформа з відкритим кодом, що безкоштовно розповсюджується Telecom Italia. JADE відповідає специфікаціям FIPA. Агентна платформа добре документована та пропонує ряд графічних інструментів для спрощення розробки та тестування мультиагентних систем, може бути розподілена між кількома серверами [10]. Платформа активно розвивається, є приклади її успішного комерційного застосування.

JADE створено на мові програмування Java, що є однією із причин вибору для розробки веб-інтерфейсу рекомендаційної системи технології Java Server Pages (jsp).

ВИСНОВКИ

У роботі досліджувались агентні РС у колаборативних середовищах.

Визначено основні риси РС, запропонована їхня класифікація відповідно до алгоритмів фільтрування. Проаналізовані алгоритми фільтрування та сфери застосування, показана ефективність застосування гібридного фільтрування. Показані переваги використання агентних технологій при проектуванні РС у колаборативних середовищах. Визначено коло проблем, які потрібно розв'язати при проектуванні систем, та окреслено шляхи їхнього вирішення. Запропонована нова агентна архітектура РС на прикладі системи для рекомендації вибіркових курсів.

Перевагою архітектури є використання кількох рівнів паралелізму для підвищення продуктивності: розподіл функцій персонального агента користувача між трьома агентами дає можливість *ефективно* підтримувати сеанс із власним користувачем та *паралельно* обробляти запити інших користувачів; операція обчислення схожості з користувачем виконується *паралельно* агентами інших користувачів; формування рекомендацій відповідно до прослуханих курсів виконується *паралельно* з формуванням рекомендацій від студентів. Мультиагентна система є одноранговою, відсутність централізованих компонентів є вдалою спробою уникнути вузьких місць у системі. Застосовано алгоритм гібридного фільтрування.

Розроблений профіль студента з використанням моделі векторного простору. Вибір векторної схожості для обчислення подібності студентів зумовлений тим, що саме ця техніка продемонструвала найкращі результати при проведенні експериментів.

Створено прототип рекомендаційної системи з використанням агентної платформи JADE, який може бути використаний у системах електронного навчання.

Можливі напрями продовження роботи: дослідження ефективності застосування інших алгоритмів фільтрування за вмістом; інтеграція рекомендаційної системи із системою

електронної освіти, та дослідження можливості використання альтернативних джерел для підтримки профілю студента та поповнення бази даних про дисципліни; розробка методів тестування системи, способів оцінки корисності рекомендацій та застосування цих результатів для підвищення ефективності алгоритмів рекомендування.

ЛІТЕРАТУРА

1. Maria Fasli, Agent Technology for E-Commerce, John Wiley & Sons Ltd, 2007.
2. Hong Lin. Architectural Design of Multi-Agent Systems – Technologies and Techniques, 2007.
3. Gulden Uchyigit, Keith Clark, A Multi-Agent Architecture for dynamic collaborative filtering, 2003.
4. Osmar R. Zaıane, Building a Recommender Agent for e-Learning Systems, 2002
5. Miquel Montaner, Collaborative Recommender Agents Based On Case-Based Reasoning and Trust, 2003.
6. Vijayan Sugumaran, Application of Agents and Intelligent Information Technologies, Idea Group Inc, 2007.
7. Chris Buckley, The Importance of Proper Weighting Methods, Human Language Technology Conference, 1993.
8. Frank McCarey, Mel O Cinneide, Nicholas Kushmerick, RASCAL: A Recommender Agent for Software Components in an Agile Environment, 2006.
9. Frank McCarey, Mel O Cinneide, Nicholas Kushmerick, Recommending Library Methods: An Evaluation of the Vector Space Model (VSM) and Latent Semantic Indexing (LSI), 2006.
10. Офіційний сайт платформи JADE [Електронний ресурс]. – Режим доступу: http://jade.tilab.com/.

Стаття надійшла до редколегії 13.04.10 р.