УДК 681.324:681.3.066:681.3.01

О.А. Войцех
О.М. Хошаба, к.т.н, доц.
Вінницький державний технічний університет

РОЗРОБКА МУЛЬТИАГЕНТНИХ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ У МІЖНАРОДНІЙ КОМП'ЮТЕРНІЙ МЕРЕЖІ INTERNET

Розглянуто разробку мультиагентних інтелектуальних систем у міжнародній комп'ютерній мережі internet.

Вступ

Проблема побудови інтелектуальних агентів і мультиагентних систем (МАС), що має 40-літню історію (Городецький і ін., 1998; Тарасов, 1998), сформувалася на основі результатів робіт в області таких наукових напрямків, як розподілений штучний інтелект, розподілене рішення задач і рівнобіжний штучний інтелект (Demazeau et al., 1990; Церанек, 1991; Rasmussen et al., 1991). Однак, в останнє десятиліття за оцінками багатьох фахівців, МАС виділяють в один з найбільш перспективних напрямків інтелектуальних інформаційних технологій, а результати робіт уже зараз дозволяють робити висновки про нову якість одержуваних оцінок при оптимальному прийнятті рішень.

Роботи першого періоду досліджень у даній області концентрувалися на розробці так званих «тямущих» (smart) агентів, що були розпочаті наприкінці 1970-х років і продовжуються до наших днів. Спочатку ці роботи були зосереджені на аналізі принципів взаємодії агентами, декомпозиції розв'язуваних задач на підзадачі і розподілі отриманих задач між окремими агентами, їх координації і кооперації, дозволі конфліктів шляхом узгодження обміну інформаційних потоків між окремими модулями тощо. Ціль таких робіт – аналіз, специфікація, проектування і реалізація систем агентів. На цьому рівні активно велися роботи з теорії, архітектурам і мовам для програмної реалізації агентів. З 1990 року стало ясно, що програмні агенти можуть використовуватися більш широко. Цьому послужив звіт консультативної фірми Ovum [Ovum, 1994], що пророчила, що сектор ринку для програмних агентів у США і Європі зросте, принаймні, до 3,9 більйонів доларів до 2005 року в противагу 1995, коли він оцінювався в 476 мільйонів доларів.

В даний час безліч дослідницьких лабораторій, університетів, фірм і промислових організацій працюють у цій області. До них можна віднести такі організації як університет Карнегі Меллон (CMU), фірма General Magic, транснаціональні компанії Apple, AT&T, BT, Daimler-Benz, DEC, HP, IBM, Lotus, Microsoft, Oracle, Sharp і ін. Найбільш відомими областями застосування цієї технології є: керування інформаційними потоками (workflow management), мережами (network management), керування транспортним повітряним рухом (air-traffic control), інформаційний пошук (information retrieval), електронна комерція (e commerce), навчання (education), електронні бібліотеки (digital libraries) тощо.

Необхідність використання МАС у міжнародній комп'ютерній мережі Internet

Існує безліч причин необхідності використання МАС. Основна з них полягає в тому, що агенти автономні і можуть виконуватися у фоновому (background) режимі від імені користувача при вирішенні різних задач, найбільш важливими з яких є збір інформації, її фільтрація і використання даних для прийняття рішень. Таким чином, основна ідея програмних агентів – делегування повноважень. Для того, щоб реалізувати цю ідею, агент повинен мати можливість взаємодії з власником чи користувачем процесу, що виконується, для одержання відповідних завдань і повернення отриманих результатів, орієнтуватися в навколишньому середовищі і приймати рішення, необхідні для виконання поставлених перед ним задач.

Існують також випадки, коли вирішення задач за допомогою стандартних методів побудови інтелектуальних систем (ІС) практично неможливі чи дуже складні. Наприклад, аналітик з питань охорони здоров'я регіону, який перевіряє ситуацію в районі за визначений період часу чи оператор ядерного реактора, що обстежує систему безпеки, повинні виконати аналіз багатьох показників. Нерідко дані, які можуть привернути увагу аналітиків, не обов'язково повинні мати максимальні чи мінімальні значення, а комбінації окремих факторів можуть залежати від деяких умов. Внаслідок цього виникає ситуація неможливості прогнозування послідовних дій

експертів, що призводить до складностей при побудові інтерфейсу користувача розроблювачами систем.

Тому до сучасних вимог побудови ІС з використанням МАС, з огляду на вищеописані проблеми, можна віднести: непередбачуваність дій користувача при формуванні запитів до баз даних чи знань; велика кількість наданих факторів, при яких складно спроектувати інтерфейс системи; значні відстані між користувачами, експертами-аналітиками й адміністраторами системи.

Методи побудови агентно-орієнтованих систем у міжнародній комп'ютерній мережі Internet

Побудова **агентно-орієнтованих** систем звичайно реалізується за допомогою двох підходів: створення єдиного автономного агента чи розробка мультиагентної системи. Автономний агент взаємодіє тільки з користувачем і реалізує весь спектр функціональних можливостей, необхідних у рамках агентно-орієнтованих програми. На противагу цьому, МАС є програмно-обчислювальним комплексом, де взаємодіють різні агенти для рішення задач, що складні в рішенні чи недоступні в силу своєї складності для одного агента. Призначення таких агентів укладається в спілкуванні, кооперації і виробленні угод між собою для пошуку оптимального рішення поставленої перед ними задачі. Агентні технології звичайно припускають використання визначених структур і їхніх моделей, вони спираються на відповідні бібліотеки і кошти підтримки розробки різних типів МАС.

Реалізацію МАС за допомогою коштів міжнародної комп'ютерної мережі Internet можна здійснити за допомогою двох підходів: традиційного, котрий передбачає застосування HTML – технологій і нестандартного, що укладається у використанні сучасних інформаційних, і мультимедіа технологій.

Необхідно відзначити, що найбільше поширення одержали використання інформаційних ресурсів на основі HTML стандартів. Застосування альтернативних методів, заснованих на використанні на стороні клієнта і сервера мов високого рівня (у даному випадку C++), є більш молодим і перспективним напрямком у побудові інтелектуальних технологій Internet. Природно, в обох методах існують як переваги, так і недоліки. Деякі з них наведені в таблиці 1.

Таблиця 1

	Класичний метод	Альтернативний метод
Недоліки	Труднощі в гнучкому контролі БД/З на рівні Apache сервера Необхідність у перезапуску Apache сервера при зміні параметрів системи Труднощі в рішенні багатокритерних задач (обмеження) Відсутність різноманітних елементів графічного інтерфейсу	Використання дорогих програмних продуктів при розробці інтерфейсу користувача
Переваги	Використання безкоштовних чи дешевих програмних продуктів при розробці інтерфейсу користувача	Можливість якісного рішення багатокритерних задач. Розвинутий інтерфейс користувача, можливість побудови якісних графіків, діаграм.

Як видно з табл. 1, альтернативні методи побудови ІС має сенс застосовувати при необхідності рішення багатокритерних задач. Привертає увагу застосування альтернативних методів у дистанційному навчанні завдяки наявності мультимедіа засобів і можливості більш якісної побудови інтерфейсу користувача.

Порівняльна характеристика методів проектування МАС

Упровадження МАС є актуальною задачею в зв'язку з тим, що створення інтелектуальних агентів дозволяє фахівцям делегувати свої повноваження за рішенням складних задач. Однак розробка МАС та інтелектуальних агентів вимагає спеціальних знань і є складною ресурсомісткою задачею тому що програмні агенти є особливим класом систем програмного

забезпечення, що діє від імені користувача. Вони є могутньою абстракцією для «візуалізації» і структурування складних об'єктів.

Разом з тим, розвиток і впровадження програмних агентів був би неможливий без попереднього досвіду розробки і практичного освоєння концепції відкритих систем [Орлик, 1997], що характеризуються властивостями:

- розширюваності/маштабування (можливість зміни набору складових системи);
- мобільності/переносності (простота перенесення програмної системи на різні апаратно-програмні платформи);
- інтероперабельності (здатність до взаємодії з іншими системами);
- дружелюбності до користувача/легкої керованості.

Одним з результатів впровадження концепції відкритих систем на практиці стало поширення архітектури «клієнт-сервер» [Орлик, 1997].

В даний час виділяються наступні моделі клієнт-серверної взаємодії:

- «Товстий клієнт – тонкий сервер». Найбільш часто зустрічається варіант реалізації архітектури клієнт-сервер. Серверна частина реалізує тільки доступ до ресурсів, а основна частина додатка знаходиться на клієнті.
- «Тонкий клієнт – товстий сервер». Модель, активно використовувана в зв'язку з поширенням Інтернет-технологій і, у першу чергу, Web-броузерів. У цьому випадку клієнтський додаток забезпечує реалізацію інтерфейсу, а сервер поєднує інші частини додатків.

При створенні МАС можуть з успіхом використовуватись обидві моделі, хоча в даний час частіше застосовується друга. Однак, незалежно від використовуваної моделі засобів розробки і виконання розподілених додатків, якими, як правило, є МАС, спираються на статичний підхід (дозволяють передавати тільки дані додатків) чи динамічний підхід (забезпечують можливості передачі коду, що виконується,).

При динамічному підході Мас-додатки використовують парадигму мобільних агентів, що переміщаються в напрямку до серверної і клієнтської частин локальної чи корпоративної мережі з використанням протоколу TCP/IP. Деякі дослідники вважають, що мобільні агенти забезпечують більш прогресивний метод роботи в мережних додатках. Інші автори відзначають, що мобільні агенти приносять небезпеку в області забезпечення таємності інформації і завантаженості мережі [Chess et. al. 1995].

Необхідно також відзначити, що ті самі функціональні можливості в більшості випадків можуть бути реалізовані як за допомогою мобільних, так і статичних агентів. Використання мобільних агентів може бути доцільним, якщо вони:

- зменшують час і вартість передачі даних (наприклад, при великих обсягах даних замість передачі всієї неопрацьованої інформації з мережі на хост-джерело посилається агент, що вибирає тільки необхідну інформацію і передає її користувачу);
- дозволяють вирішувати проблеми обмеження локальних ресурсів (наприклад, якщо можливості процесора й обсяг пам'яті клієнтського комп'ютера малі, доцільніше використання мобільних агентів, що виконують обчислення на сервері);
- більш ефективно виконують координаційну функцію (наприклад, запити до вилучених серверів виконуються мобільними агентами як окремі задачі, у зв'язку з тим, що не має потреби в координаційному керуванні);
- дозволяють виконувати асинхронні обчислення (наприклад, запустивши агента, можна переключитися на інший додаток і навіть від'єднуватися від мережі, а результат буде доставлений агентом адресату після виконання завдання).

Мобільні агенти є перспективними для МАС, але в даний час немає єдиних стандартів їхньої розробки й усе ще залишається невирішеним ряд проблем, таких як легальні способи переміщення агентів по мережі, верифікація агентів (зокрема, захист від переданих по мережі вірусів), дотримання агентами прав приватної власності і збереження конфіденційності інформації, якими вони володіють, перенаселення мережі агентами, а також сумісність коду агента і програмно-апаратних засобів мережної машини, в якій він виконується [Wayner, 1995].

Найбільш відомими технологіями реалізації статичних і динамічних розподілених додатків є програмування сокетів, виклик вилучених процедур – RPC (Remote Procedure Call), DCOM (Microsoft Distributed Component Object Model), Java RMI (Java Remote Method Invocation) і CORBA (Common Object Request Broker Architecture) [Maurer et al., 1998]. Разом з тим, на

думку деяких авторів [Gopalan, 1999], з погляду розробки і реалізації МАС найбільш важливими є останні три – DCOM, Java RMI і CORBA.

Модель Microsoft DCOM є об'єктною моделлю, що підтримується Windows 95, Windows NT, Sun Solaris, Digital UNIX, IBM MVS і ін. Основна її цінність – у наданні можливостей інтеграції додатків, реалізованих у різних системах програмування.

Java RMI-додатка звичайно складаються з клієнта і сервера. При цьому на сервері створюються деякі об'єкти чи їхні методи, які визначаються як доступні для виклику вилученими і локальними додатками. На клієнтській частині реалізуються додатки, що використовуються вилученими об'єктами. Відмінною рисою RMI є можливість передачі в мережі не тільки методів, але і самих об'єктів, що забезпечує в кінцевому рахунку реалізацію мобільних агентів.

CORBA є частиною OMA (Object Management Architecture), розробленої для стандартизації архітектури й інтерфейсів взаємодії обєктно-орієнтованих додатків. Інтерфейси між CORBA-об'єктами визначаються через спеціальну мову IDL (Interface Definition Language), що є мовою опису інтерфейсів. Самі інтерфейси можуть при цьому бути реалізовані на будь-яких інших мовах програмування і приєднані до CORBA-додатків. У рамках стандартів передбачається, що CORBA-об'єкти можуть спільно працювати з DCOM-об'єктами і через спеціальні CORBA-DCOM мости (Bridges).

Технології Java RMI і CORBA є на думку деяких авторів самими гнучкими й ефективними засобами реалізації розподілених додатків [Gopalan, 1999]. Ці технології дуже близькі по своїх характеристиках. Основною перевагою CORBA є інтерфейс IDL, що уніфікує засоби комунікації між додатками і інтероперабельність з іншими додатками. З іншого боку, Java RMI є більш гнучким і могутнім засобом створення розподілених додатків на платформі Java, включаючи можливість реалізації мобільних додатків.

Таким чином, у даний час існує безліч напрямків для реалізації МАС. На нашу думку, для побудови МАС є необхідно використовувати наступні компоненти:

– мультиплатформені системи, що використовують позитивні сторони різних операційних систем;

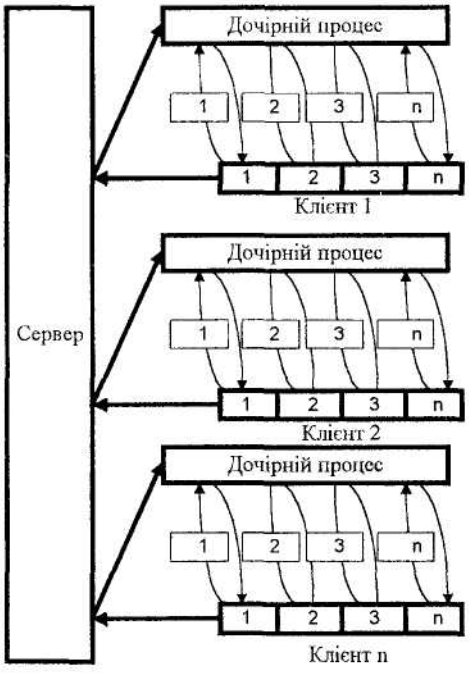
– засоби міжнародної комп'ютерної мережі Internet;

– сучасні засоби керування базами даних/знань;

– спеціальні компоненти бібліотек аналізу багатофакторної інформації.

Реалізація МАС

Звичайно, при реалізації мережних клієнт-серверних програм на базі операційної системи Unix за допомогою системної утиліти - inetd обробляються вхідні TCP - з'єднання чи UDP - датаграми. При цьому використовуються TCP чи UDP протоколи.



Мал. 1 Схема механізму розгалудження (відпачковування)

Для реалізації TCP – сервера inetd обробляє відомі порти, очікуючи запиту на з'єднання, потім з'єднує, асоціює з ним файлові дескриптори stdin, stdout і stderr після чого виконує додаток.

При реалізації UDP – сервера немає необхідності установки попереднього з'єднання. У цьому випадку, програма виконує запит до операційної системи за допомогою виклику select про прихід нових датаграм у порт UDP – сервера. Одержавши повідомлення, програма дублює дескриптор сокета на stdin, stdout і stderr, після чого запускає UDP – сервер.

У даній роботі використовувався UDP – сервер, що виконував прослуховування 5993 порту. У цьому випадку, програма виконувала обробку даних які надійшли в порт навіть у тому випадку, якщо попереднє повідомлення не завершило процес. Це досягалося за допомогою механізму

розгалуження (відпачкування) процесу, при якому оброблялися додаткові екземпляри UDP – сервера (мал. 1). У цей момент програма inetd може відновити прослуховування відомого порту UDP – сервера в чеканні нових повідомлень. Завдяки цьому з'являється можливість у межах одного запиту обробляти практично необмежену кількість діаграм, що надходять від одного клієнта.

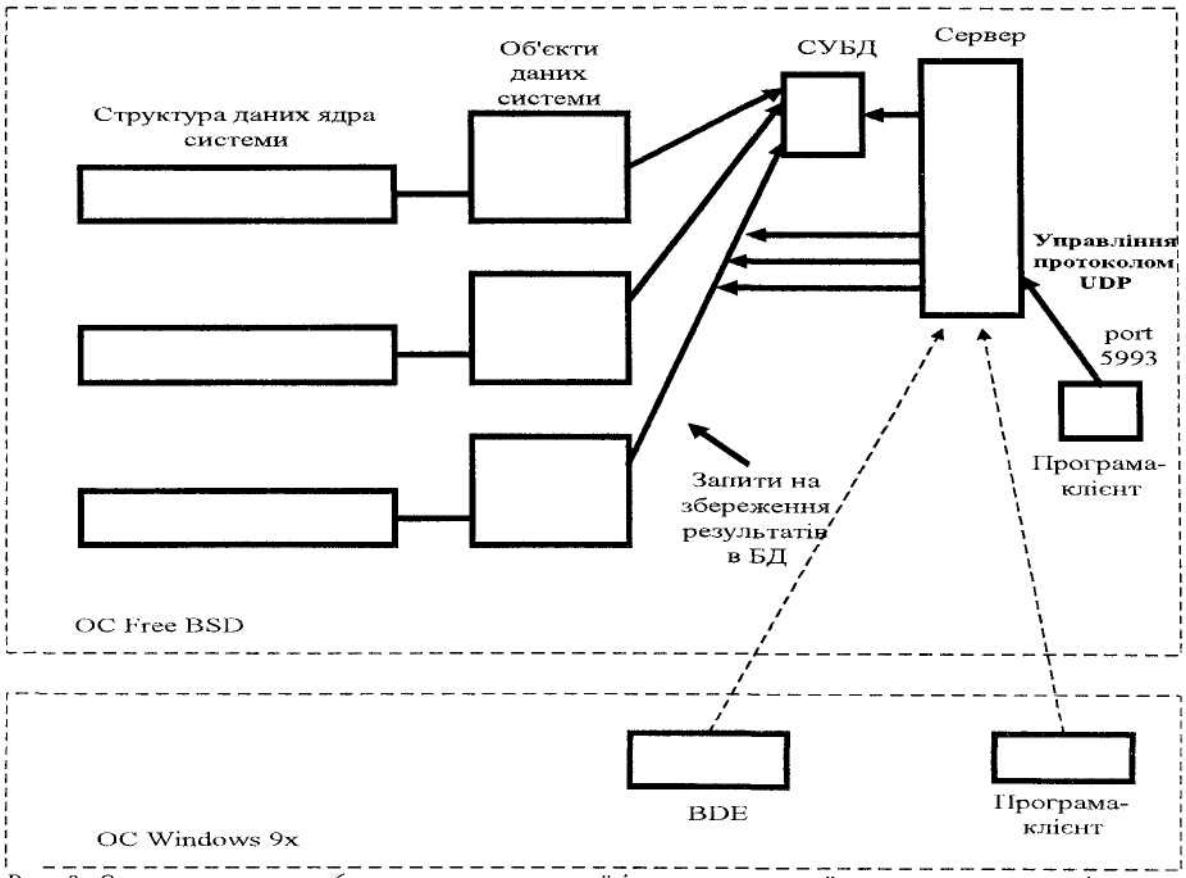


Рис. 2. Загальна схема роботи мультиагентної інтелектуальної системи по управлінню та моніторингу трафіка вузла Internet

Ознакою закінчення роботи з даним клієнтом служить спеціальна пауза чи пауза 30 сек. У роботі цей прийом унеможливлював утворення нескінченного циклу (зависання) процесу.

Необхідно також відзначити, що на кожну, яка прийшла і була оброблена, датаграму UDP – сервер відправляв відповідне повідомлення програмі – клієнту. Цим забезпечувалася надійність роботи системи у випадку втрати датаграм.

Розробка даного механізму одержання/відправлення повідомлень дозволила побудувати клієнт-серверну програму по обробці вхідного і вихідного трафіка вузла Internet. Для цього були створені об'єкти, що відповідають за моніторинг і керування структурами даних ядра операційної системи (мал. 2). Інформація, оброблювана об'єктами надходила в базу даних. Керування діями об'єктів виконувалися за допомогою програм – клієнтів, написаних мовою високого рівня C++ для операційних систем Unix і Windows 9/x, 2000, NT. У число таких операцій входили: запис поточного стану об'єктів з інформацією про вхідні і вихідні потоки даних, командами на закриття і відкриття необхідних портів і протоколів. При виконанні таких операцій програма-сервер звертається до конфігураційного файла, що містить інформацію про можливість виконання будь-якої операції таким чином, щоб було неможливим помилково закрити важливий порт чи протокол.

Таким чином, використовуючи технологію побудови МАС у Вінницькому інформаційно – аналітичному медико-статистичному центрі впроваджені програмні засоби з моніторингу трафіка вузла Internet корпоративної мережі health.org.ua. Розроблені програми мають інтерфейс, що підтримується в операційній системі Windows 9x/2000/NT і Unix, за допомогою якого виконувалися запити на пошук необхідних записів і одержання підсумкових результатів за визначені проміжки часу.

ЛІТЕРАТУРА:

1. *Хошаба А.М.* Впровадження програмних засобів моніторингу WEB– та FTP–орієнтованих інформаційних ресурсів міжнародної комп'ютерної мережі INTERNET // Технологічні системи. – Київ, 2000. – № 4 (6). – С. 91–93.
2. *Хошаба О.М.* Застосування програмних засобів по моніторингу WEB– та FTP–орієнтованих інформаційних ресурсів міжнародної комп'ютерної мережі INTERNET // Вимірювальна та обчислювальна техніка в технологічних процесах. – Хмельницький, 2000. – № 4. – С. 97–102.
3. *Локазюк В.М., Поморова О.В., Домінов А.О.* Інтелектуальне діагностування мікропроцесорних пристроїв та систем: Навч. посібник для вузів, 2001. – 286 с.
4. *Хошаба А.М., Король М.П.* Альтернативні методи побудови інтелектуальних систем міжнародної комп'ютерної мережі Internet // Вісник Херсонського державного технічного університету. – Херсон, 2002. – № 14.– С. 206–211.
5. *Хошаба О., Борей З., Войцех О.* Розробка програм по моніторінгу інформаційних ресурсів мережі Internet // Контроль і управління в складних системах. Тези доповідей шостої міжнародної науково-технічної конференції. Універсам. – Вінниця, 2001. – С. 163.

Подано 19.09.2002