УДК 681.3

ПІДХІД ДО АДАПТИВНОЇ КОМПОЗИЦІЇ СЕРВІСІВ В SEMANTIC WEB СЕРЕДОВИЩІ НА ОСНОВІ ПОТОКІВ РОБІТ ПРОГРАМНИХ АГЕНТІВ

В. Дерецький

Інститут програмних систем НАН України,
03187, Київ-187, проспект Академіка Глушкова, 40.
Тел.: +38044526 4342; dva@isofts.kiev.ua.

Веб-сервіси відіграють важливу роль у розробці розподілених систем. Зокрема забезпечують створення нових функціональних можливостей на основі вже існуючих веб-сервісів, що є перспективним підходом для створення складних розподілених застосувань і бізнес-процесів. Можливість динамічного створення веб-сервісів, спрямованих на роботу за умов зміни навколишнього середовища, є розширеним завданням композиції. Існуючі підходи до динамічної композиції, в основному не враховують встановлених специфікацій сервісів, наприклад, таких як WS-BPEL, оскільки вони пропонують модифікації у правилах специфікацій бізнес-процесів, накладаючі більше труднощів для реалізації такої схеми. Ця робота представляє підхід до створення адаптивних веб-сервісів через семантичну модифікацію специфікації сервісів. Розвиток нових технологій дасть змогу організаціям створювати єдине уніфіковане представлення даних у всіх застосуваннях, дозволить точно знаходити необхідну інформацію, спрощувати корпоративну інтеграцію і інтеграцію мережних застосувань. Запропонована схема для представлення та рішення задач в розподіленому середовищі Семантичної мережі. Розглянуто засоби та стандарти інтеграції та координації мережних сервісів на основі потоку робіт (WorkFlow).

The Web-services play an important role in development of the distributed systems. In particular provide creation of new functional possibilities on the basis of already existing Web-services which is perspective approach for creation of the difficult up-diffused applications and businesses processes. Possibility of dynamic creation of Web-services directed on work in the conditions of change of terms of environment is a dynamic task composition. Existent going near dynamic composition, mainly does not take into account the set specifications of services, for example such as WS-BPEL, as they offer to modification in the rules of specifications of businesses processes, imposing more difficulties for realization of such chart. This work presents going near creation of adaptation composition Web-services through semantic modification of specification of services. This paper examines proposals and standards for e-services from the perspectives of XML, data management, WorkFlow, and process models. Key areas for study are identified, including behavioral service signatures, verification and synthesis techniques for composite services, analysis of service data manipulation commands. The offered chart for presentation and decision of tasks in the distributed environment of the Semantic Web. Facilities and standards of integration and coordination of Semantic Web services are considered on the basis of stream of works (WorkFlow).

Вступ

В останні роки збільшується інтерес до адаптивных систем. В контексті веб-сервисів адаптивність слідує з того факту, що нові веб-сервисі можуть бути створені і стають доступні автоматичним способом, без зупинки та переналагодження системи в разі зміни умов її виконання. Прикладом може бути зміна трафіку в Інтернет, який не забезпечений надійним середовищем комунікації. У цьому контексті важливість використання адаптації є очевидною.

Адаптація веб-сервісів може бути статичною або динамічною, ручною або автоматичною, превентивною або реактивною [1]. Статична адаптація виконується через модифікації на етапі створення сервісу, тоді як динамічна змінює програму під час її виконання як шляхом прямого людського втручання засобами адаптації, так і автоматично, при якому зміна програми може бути виконана системою безпосередньо. Нарешті, превентивна адаптація відбувається перед специфічною подією і реактивною, коли адаптація здійснюється після події. Наприклад, система може зрозуміти, що один сервіс не доступний, і потім приймає заходи перед виконанням (превентивна адаптація). Якщо система приймає заходи після невдалого або помилкового запиту – це відповідає реактивній адаптації.

Крім вищезазначених типів адаптації слід розглянути три наступні категорії: як, коли і де [2] відбувається адаптація. "Як" звертається до механізму, який має дозволяти адаптацію. Ці механізми мають бути засновані на деякій методиці і мають бути запроваджені безпосередньо в систему. Час ("коли"), використовується для того, щоб визначати, час трансляції, час розгортання і час виконання адаптації. Якщо адаптація відбувається під час виконання, то її визначають як динамічну адаптацію на відміну від статичної, де адаптація відбувається на етапі створення нової функціональності сервісу.

Для ілюстрації типів композиції розглянемо реальні веб-сервіси і умови, при яких їх композиція має місце в розподілених застосуваннях. Припустимо, нам необхідно перекласти текст з української мови на китайську. Такої програми-перекладача немає. Але є програми перекладу з української на російську і з російської мови на китайську. Кожна з таких програм оформлена як веб-сервіс. Сервіс перекладу з української мови на китайську є композицією відповідних сервісів. На рис. 1 надані традиційні типи композиції сервісів: ітеративна композиція, послідовна композиція, композиція з вибором і паралельна композиція.

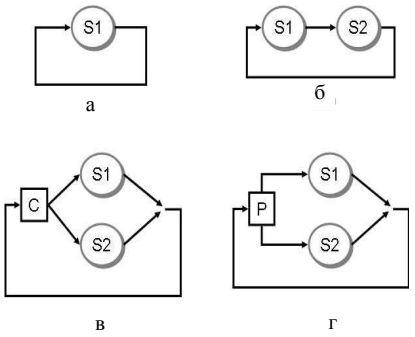

Рис. 1. Типи композиції сервісів

У ітеративній композиції заданий сервіс S1 запускається кілька разів послідовно. У послідовній композиції, якщо сервіс S2 виконується після S1, то мається на увазі, що S1 має завершитися з позитивним результатом, а потім може бути запущений сервіс S2. Композиція такого типу представлена на рис. 1, б. У композиції з вибором, де учасниками є два і більше сервісів (S1 і S2), але одночасно може бути вибраний для виконання тільки один з них. Вибір здійснюється на основі специфікованого критерію (критерій порядку). У композиції цього типу сервіси S1 і S2 можуть взаємодіяти або ні. Цей випадок представлений на рис.1, в, де оператор С здійснює вибір сервіса. Нарешті, в паралельній композиції два або більше сервісів виконуються в водночас незалежно. Рис. 1, г представляє цю ситуацію (оператор Р визначає паралелізм композиції). Ці різні типи композиції можуть бути вбудовані в мовні засоби композиції.

Не існує стандартів, які б специфікували б композицію. WS-BPEL є мовою, спрямованою на виконання бізнес-процесів, які дозволяють різним веб-сервісам виконуватися спільно, вирішуючи задане бізнес завдання. Спробою стандартизації композиції сервісів може бути проект OASIS (IBM, Microsoft, Oracle, Sun). До цих пір він служить стандартом де-факто.

Структура роботи є наступною. В розділі 1 визначаються семантична специфікація SWS, засоби створення Семантичні Web-сервіси (SWS) і Web-сервіс (WS) на концептуальному рівні, визначені вимоги та засоби композиції сервісів. У розділі 2 описується архітектура SWS застосувань, засобів для розгортання та виконання SWS шляхом координації потоку робіт, мережних агентів. В розділі 3 представлена модель адаптивної композиції сервісу на основі взаємодії агентів розглянуто та архітектура мультиагентної координації. Основні результати роботи підсумовуються в розділі 4.

1. Технологія та стандарти веб-сервісів

Сервісно-орієнтована архітектура SOA (Service Oriented Architecture) визнана революційною в технології побудови та інтеграції корпоративних інформаційних систем. За визначенням, SOA – це парадигма, яка призначена для проектування, розробки і керування застосуваннями в мережному середовищі. Використання цього підходу вимагає від розробників забезпечення проектування застосувань як набір сервісів. Розробники мають "вийти за межі" своїх застосувань і подумати, як скористатися вже існуючими сервісами або як їх сервіси можуть бути використані іншими розробниками та користувачами застосувань.

SOA спрямована на використання альтернативних технологій і підходів інтеграції застосувань (таких як обмін повідомленнями) за допомогою «зв'язування» сервісів, а не за допомогою написання нового програмного коду. В цьому випадку, при належному проектуванні, система може вчасно реагувати на зміни умов середовища, шляхом "настройки" процесу обміну повідомленнями, а не розробки нових програм. До того ж процес настройки може бути автоматичним.

SOA також розглядається як засоби активації Web-сервісів, які можуть викликатися іншими програмами, виступаючими як клієнти або споживачі цих сервісів. Ці сервіси можуть бути новими, або давно розробленими прикладними програмами, які можна активізувати як чорний ящик. Від розробника не вимагається знань, як працює програма, необхідно лише розуміти, які вхідні і вихідні дані потрібні, і як викликається ця програма на виконання.

В загальному вигляді модель SOA припускає наявність трьох основних учасників: постачальника сервісу, споживача сервісу і реєстру сервісів. Взаємодія учасників виглядає достатньо просто: постачальник сервісу реєструє свої сервіси в реєстрі, а споживач звертається до реєстру із запитом.

Насправді SOA – це всього лише інший стиль побудови сучасних корпоративних систем. Він орієнтується на сервіси і характеризується розподіленою архітектурою і слабо зв'язаними інтерфейсами [3].

Web-сервіс (WS) в поточному використанні – це не що інше, як одиниця робіт, що виконується сервіс-провайдером (службою, що надає сервіс) для забезпечення бажаного результату користувачеві сервісу. Саме сервіс, а не об'єкт, як в об'єктно-орієнтованому програмуванні, є тим модулем або програмним кодом, що

повторно використовується, і при цьому він не залежить від технологій, мовних середовищ і інших ресурсів. Інтегруючу роль між сервіс-провайдером і користувачем беруть на себе програмні агенти.

За визначенням W3C, Web-сервіс (Web service) спрямований на забезпечення взаємодії „машина-машина" через мережу. Web-сервіс має інтерфейс, описаний у форматі, доступному для машинної обробки (WSDL). Інші сервіси також взаємодіють з Web-службою, так як це визначено їх описом, з використанням SOAP-повідомлень, що передаються за протоколом HTTP в поєднанні з іншими Web-стандартами [4].

Корпоративна інформаційна система, побудована на основі SOA, складається з набору сервісів, доступних через прикладні програмні інтерфейси, та вбудованого механізму пошуку. Перелік сервісів у загальному реєстрі дозволяє користувачеві визначити службу та сервіс, що пропонує шукану функцію.

Практичні аспекти сервісно-орієнтованої технології дозволяють розв'язати проблеми масштабованості, інтегрувати мережі передачі даних, спростити процедури проектування і керування мережами, а також створити інші розподілені застосування на засадах вже існуючих, прозоро взаємодіючи з ресурсами систем за допомогою прикладних програмних інтерфейсів і відкритих стандартів.

Web-сервіси спрощують Web-застосування. Загальні реєстри сервісів доступні будь-яким зацікавленим клієнтам у всьому світі, надбудова (на відміну від вбудови) Web-сервісів над корпоративними системами не вимагає їх істотних змін, а загальнодоступні API (прикладні програмні інтерфейси) вирішують проблеми повторного використання коду і масштабності його застосування.

Web-сервіси – це суть SOA з двома суттєвими обмеженнями: інтерфейси базуються на протоколах (HTTP, FTP, SMTP, TCP), а всі повідомлення описуються у форматі XML. Детальні описи стандарту Web-сервісів і специфікацій SOA наводяться на сайтах консорціуму W3C [5]. Веб-сервіси мають властивості та можливості бути знайденими та до них можна звернутися від розподілених застосувань. Веб-сервіси визначаються засобами мови опису (WSDL) [3] и можуть бути викликані на виконання засобами, побудованими на використанні протоколу доступу до об'єктів (SOAP) [5]. Традиційний WSDL підтримує синтаксичні форми визначення веб-сервисів, тоді як, семантичні веб-сервіси визначаються розширенням WEB машинно-читаною інформацією і автоматизованими сервісами.

Модель SOA, яка представлена набором рівнів, дозволяє забезпечити такі основні функції для підтримки WS:

1) публікація WS, де використовується специфікація UDDI, яка визначає можливості WS та характеризує постачальника сервісу;

2) опис WS засобами мови WSDL, щоб конкретизувати як сервіс може бути ініційований (вхідні та вихідні повідомлення), і засоби SOAP, як протокол комунікації для доступу до WS;

3) композиція WS, яка конкретизує як комплексний сервіс може складатися з інших сервісів засобами мови потоків даних веб-сервісів WSFL.

2. Семантичні веб-сервіси

Семантична мережа (Semantic Web) [6] надає засоби організації такого представлення даних в мережі, щоб допускалася не тільки візуалізація, але й ефективна автоматична обробка даних програмами різних виробників.

Суть ідеї можна представити таким чином: якщо розглянути пару „дані-програми", що використовується в традиційних програмних системах, то очевидно, що значення оброблюваних даних міститься в самих програмах, які ці дані обробляють, а їх коді містять алгоритм розуміння цих даних. У підходах Семантичної мережі вводяться метадані, тобто дані про дані, які розглядаються як знання про дані і, які винесені за межі програмного коду. В метадані вкладено смисл даних або частину смислу. Якщо метадані обробляти таким же чином і разом з даними, то програми зможуть стати універсальними та спрощеними, що є прогресивним чинником.

Парадигмою для побудови прикладних застосувань в Семантичній мережі є підхід, заснований на використанні Семантичних Web-сервісів [7]. Семантичні Web-сервіси (SWS) розширюють поняття звичайних Web-сервісів в частині використання семантичної інформації, а саме онтологій та семантичної розмітки як для прикладних, так і для системних потреб [1].

Підхід заснований на семантичних описах властивостей сервісів і використанні можливостей сервісів, які представляються в онтологіях, міркуванні про функціональну композицію сервісів [1].

Засоби побудови SWS повинні дозволити користувачу розробляти SWS шляхом моделювання сервісів на концептуальному рівні та забезпечити розгортання сервісів на фізичному рівні, шляхом композиції існуючих сервісів в середовищі Семантичної мережі. Моделювання сервісів на концептуальному рівні повинно забезпечувати перевірку коректності і завершеності сервісу, який створюється. Один раз перевірений сервіс може бути використаний на наступних етапах проекту.

Мова розмітки агентів для сервісів – (DAML-S) і мова WEB-онтологій для сервісів (OWL-S) [8] дозволяють автоматизувати підтримку агентів, в частині визначення місцезнаходження, вибору, використання, композиції і моніторингу сервісів. Розподілені регістри сервісів, такий як Universal Description Discovery and Integration (UDDI) [5], визначають специфікації розподілених сервісів. Крім того, інші мови на основі XML, такі як мова потоків даних веб-сервісів (WSFL), мова виконання бізнес-процесів для веб-сервісів (BPEL4WS), і мова для моделювання бизнес-процесів (BPML), визначають композицію сервісів на основі процесів [5, 9].

Проте, ці мови визначені текстовим способом, який знаходиться в суперечності з візуальними специфікаціями, розробками програмного забезпечення, що зазвичай приймаються розробниками, зокрема, використання мови моделювання (UML) [8, 10].

3. Моделювання адаптивної композиції сервісу на основі взаємодії агентів

Агент-орієнтовані технології з використанням SWS направлені на потреби великих гетерогенних систем, для забезпечення розподіленого планування і керування або координації, наприклад, в задачах логістики, де організаційна функція може мати певну перевагу, використовуючи методи SWS.

Проблемними аспектами, які мають загальний характер і які визначені як семантичні функції SWS, є:

ідентифікація SWS (matchmaking) та їх пошук;

автоматизована композиція SWS;

дослідження стану (моніторинг) процесів SWS;

протоколи взаємодії SWS, координація-інтерпретація і виконання;

семантичне посередництво (наприклад, знаходження SWS за змістом запиту або повідомлення, описів або заяв про своє існування);

планування виконання SWS (формування повідомлень або запитів) і інтерпретація відповідей;

переговори і укладання контрактів;

керування онтологіями, створення, пошук і доступ до онтологій;

керування процесами створення SWS (фабрики сервісів, інсталяція, переміщення);

посередництво процесам і делегування функцій;

послуги репутації (історія використання SWS).

Одна з проблем, що супроводжує створення та використання онтологій при створенні SWS, полягає у змістовному поєднані єдиного смислового простору. Сучасна теорія не дає надійного способу поєднання даних, визначених різними онтологіями. Тому основним підходом до рішення цієї проблеми є наявність єдиної базової онтології щодо загального поля не специфікованої інформації і її розширень. Таку онтологію природно назвати „моделлю світу”, підкреслюючи її фундаментальну роль щодо моделей предметних областей. Засобам представлення знань в Семантичній мережі властиві універсальні виразні можливості, синтаксична і семантична інтероперабельність. Семантична інтероперабельність реалізується, наприклад, в онтологіях шляхом встановлення відповідності між термінами, що використовуються [11].

Роль агентів в створенні та реалізації сервіс-орієнтованих застосувань. В запропонованому підході маємо розрізняти сервіси і агенти. Розглянемо основні відмінності між ними:

WS безпосередньо «знають» тільки про себе, вони не володіють будь-яким мета-горизонтальним усвідомленням;

WS можуть бути розроблені без урахування того, щоб використовувати або “розуміти” онтології;

WS можуть бути не здатними до кооперативної дії через комунікації, або не здатні до кооперативної поведінки [3].

Напроти до вищезазначеного, агенти володіють всіма цими властивостями. Агент визначається як сервіс, який володіє такими основними властивостями як: ситуативність, автономність, реактивність, pro-активність і соціальна здатність [12].

Метафора соціальної здатності надає особливої потужності агентно-орієнтованій парадигмі і є однією з особливостей, що робить агента спроможним для створення розподіленої системи. Якщо в мультиагентній системі кожен агент представляє індивідуальний сервіс, який в свою чергу отримує можливість координуватися з іншими сервісами через ці агенти, то WS також отримують соціальну властивість. Агенти забезпечують динамічні соціальні форми поведінки сервісів, через які вони розділяють зобов'язання для досягнення загальної мети [12].

Для цього кожен агент має визнати, що кращий шлях досягнення мети полягає в залученні допомоги інших агентів. Суспільні зобов'язання виникають відповідно до суспільної залежності, коли один агент робить зобов'язання щодо іншого.

Суспільні агенти, які мають доступ до семантичного опису (онтології), спроможні до координування своєї поведінки. Навіть без семантичної інформації, агенти можуть використовуватись для виконання локальної оптимізації потоку робіт. Агенти використовують семантичні описи поведінки для того, щоб обдумати свої власні можливості і можливості інших агентів в системі, і використати ці міркування для кооперативного вирішення задачі.

Ми пропонуємо підхід, який базується на використанні опису обчислювального процесу для того, щоб нав'язати певний порядок у поведінці колекції агентів та відповідних їм сервісів.

Архітектура системи інтеграції SWS з використанням мультиагентного потоку робіт показана на рис. 2.

Ключовим значенням для вирішення проблеми інтеграції сервісів є забезпечення стандарту взаємодії сервісів. Для вирішення цієї задачі використовуються засоби BPEL (Business Process Execution Language). Засоби BPEL є новим стандартом для інтеграції гетерогенних застосувань і сервісів, і є мовою потоків робіт, процесів і даних. Основу BPEL складають три ключові властивості: асинхронність, координація потоків і керування винятковими ситуаціями [2, 9, 13].

Стандарти керування потоками робіт забезпечують ряд очевидних переваг таких як:

строга формалізація процесу опису потоків робіт, що забезпечує стандартний процес аналізу і побудови нових моделей складних робочих процесів;

переносимість і інтероперабельність: моделі процесів, створені в межах однієї системи, можуть частково або повністю працювати під керуванням іншої системи;

універсальність: розширюваність стандартів робить можливим застосування єдиного механізму опису управління потоками робіт у різних сферах діяльності.

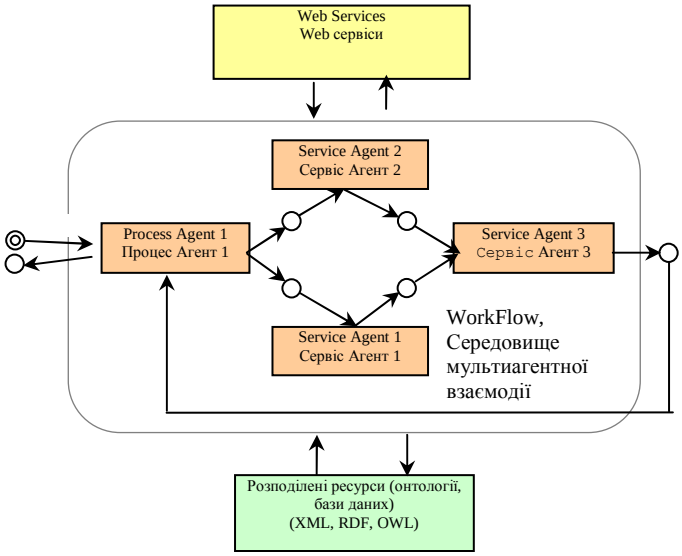

Рис. 2 Схема мультиагентної координації сервісів

Головною характеристикою запропонованої потокової архітектури для адаптивної композиції сервісів є використання підходів агент-орієнтірованного програмування до реалізації WorkFlow-композіції, з використанням властивостей адаптивності агентів. Програмні об'єкти були визначені як агенти, які володіють певними характеристиками. Групи характеристик визначають рівні агентів [14]. Рівні, класифіковані як слабкі агенти, мають тенденцію до володіння такими характеристиками, як автономність, соціальна здатність реактивність і/або проактвність. Проте, сильні агенти на додаток до характеристик слабких агентів, мають також володіти здібностями до міркування, навіть такими як емоції, які, зазвичай, приписуються людській поведінці.

Агенти в розглянутій моделі можуть бути класифіковані як слабкі агенти, оскільки автономні програмні об'єкти мають знання про середовище для реалізації властивостей реактивного і проактивного керування сервісом і процесами. Агенти мають можливість виклику, що визначається сервісами або бізнес-процесами, реагуючи як на функціональні, так і нефункціональні умови. Крім того ці агенти запрограмовані із загальними властивостями, щоб реагувати на нестандартні або аварійні ситуації.

Для пошуку потрібних сервісів використовуються засоби UDDI-сервер, який містить перелік доступних Web-сервісів. І хоча програма або агент може знайти Web-сервіс без допомоги людини, вона не в змозі зрозуміти, як саме ним користуватися і навіть для чого він призначений. Мова опису Web-сервісів (WSDL) дає нам інструмент для опису того, яким чином можна взаємодіяти з тим або іншим Web-сервісом, тоді як семантична розмітка, що є засобом SWS, забезпечує нас інформацією про те, що і як робить даний сервіс [15].

Онтології сервісів, використовують для кодування класів і підкласів понять і їх відношень. Онтології надають індивідуальному сервісу можливість успадковувати розділені поняття і словники із специфічної області. Наприклад, ми можемо визначити онтологію, що містить клас „Купувати" з підкласом „Купувати мобільний телефон" і т. д.

Щоб SW-сервіси стали реальністю, мова розмітки має бути достатньо інформативною в тому сенсі, щоб комп'ютер був здатний самостійно розуміти значення записаних виразів. Вимоги, яким має відповідати така мова, полягають в наступному.

Необхідність пошуку сервісів (виявлення – discovery). Програми мають можливість самостійно знаходити необхідні їм Web-сервіси. Слід зауважити, що ні WSDL, ні UDDI не дають можливості програмі зрозуміти, для чого саме з погляду клієнта служить той або інший Web-сервіс. Семантичний Web-сервіс зможе пред'явити опис своїх властивостей і можливостей з тим, щоб програми могли самі розпізнавати його призначення.

Необхідність запускати сервіси (запуск – invocation). Програми мають вміти самостійно розпізнавати, яким чином слід запускати і виконувати даний сервіс. Наприклад, якщо виконання сервісу є багатокроковою процедурою, то програма має знати, як їй слід взаємодіяти з сервісом, щоб виконати послідовність кроків. SW-

сервіс має забезпечити вичерпний перелік того, що має уміти агент для запуску і виконання даного сервісу. Він повинен також містити опис вхідних і вихідних даних.

Необхідність спільного використання кількох сервісів (композиція – composition). Програми мають вміти відбирати потрібні їм Web-сервіси і комбінувати їх для досягнення своєї мети (виконання функції). Сервісам необхідно буде тісно взаємодіяти один з одним, щоб були прийнятними результати спільного виконання функції. Так, програмні агенти зможуть будувати абсолютно нові сервіси, комбінуючі вже існуючі сервіси в мережі.

Необхідність контролювати стан виконання, що відбувається після запуску сервісу (моніторинг – monitoring). Програмний агент має уміти визначати властивості даного сервісу і стежити за його виконанням. Деяким сервісам потрібен певний час для виконання роботи, за який може бути змінено стан середовища, і агенти мають стежити за ходом виконання сервісу.

Семантичні Web-сервіси описуються з використанням мови семантичної розмітки. Такий семантичний опис дає змогу агентам „розуміти" SWS як за складом функцій, що вони виконують, так і за внутрішньою структурою, і тому агенти можуть знаходити, створювати композитний (складний) сервіс, запускати і контролювати такі сервіси.

Семантична розмітка WS забезпечується засобами DAML+OIL або OWL. Ці мови комбінують із стандартними мовами опису Web-сервісів WSDL та SOAP. Хоча і в SW-орієнтованих мовах програміст має створювати сервіси на концептуальному рівні для того, щоб уникати структурних помилок та невідповідностей, або помилок впродовж створення та розробки архітектури WS. У цьому сенсі розробка SWS потребує моделювання на концептуальному рівні і включає специфікацію опису сервісу, функціональні і структурні засоби.

SWS-композиція полягає в комбінуванні кількох сервісів для створення нового сервісу. Тому розробка SWS і композиція SWS мають схожі функції. Проте відмінність їх полягає у тому, що композиція використовує сервіси, які вже існують, і є напівавтоматичним процесом. При розробці SWS користувачі вручну створюють сервіси, використовуючи графічний інтерфейс.

Процес композиції сервісу з використанням агетно-орієнтованого підходу в загальному сенсі в запропонованій моделі, складається з п'яти кроків, як показано на рис. 3. У першому кроці, визначаються сервіси, які потрібні для організації технологічного процесу композиції.

Пошук сервісів відбувається посеред реєстру UDDI або в локальному середовищі, в якому зареєстровані сервіси. На другому кроці, зберігаються характеристики сервісу і фіксуються в потоковій агент-орієнтірованной моделі сервісів. Так само на другому кроці визначаються характеристики сервісів і формується відповідна агентна модель, яка готова для подальшої роботи. На третьому кроці проектувальник потоку даних створює проектну модель, як для сервісів, так і для процесів.

Як тільки модель процесів створена, на четвертому кроці, забезпечується фіксація інформації процесу. Ця інформація запам'ятовується в процес-орієнтіроваій моделі даних, яка готова до роботи з агентами.

У п'ятому і завершальному кроці, агенти здійснюють доступ до потокової моделі даних – WorkFlow і працюють з нею для керування сервісами, реагуючи на зміни середовища.

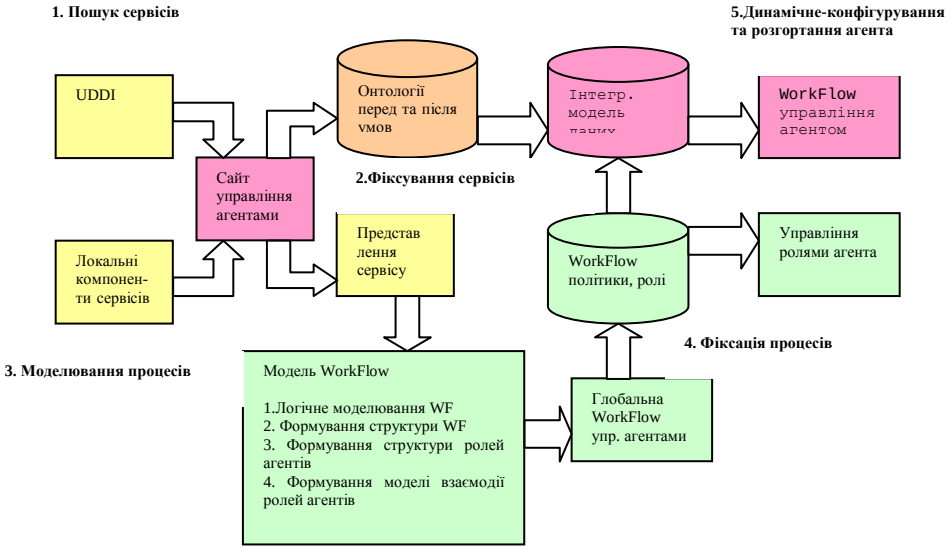

Рис. 3. Загальна схема динамічної WorkFlow - композиції сервісів

Формальне визначення агентів прикладного рівня будується на основі ролей потокової моделі, які за суттю є запитами до проксі-сервера. Операційні запити містять інформацію про запуск розподілених сервісів у відповідний час. Інформація про завершення отримується на основі моніторингу сервісів, визначенні помилок, і

профілактичні повідомлення про нестандартні ситуації. Формально, керування ролями агента може визначити потік даних набору ролей WR. Роль агента визначається як:

$$Rma = \{Wr1, Wr2. . .,WRn\}.$$

Тому, роль усієї WorkFlow визначається як кортеж WR: $W_R = (R_{ni}, S, G_i)$, де *Rni* – унікальне ім'я для ролі WorkFlow, *S* – набір сервісів: $S = \{s1, s2, s3, . ., sn\}$.

Gi – група або перехід для множини сервісів, які можуть бути зв'язані через умови AND або XOR. Умова AND конкретизує, що багаторазові сервіси мають виконуватися паралельно, тоді як тип XOR конкретизує, що певний сервіс має бути вибраний серед можливих сервісів (зазвичай на підставі історії попередніх запусків сервісів).

Атомарні сервіси можуть бути представлені трійкою: $S_i = (N_{si}, O_i, T_i)$, де *Nsi* – унікальна комбінація імен сервісу в наборі доступних сервісів, *Oi* – ім'я напрямку сервісу, за яким сервіс доступний, і яке використовується для доступу до сервісу, *Ti* – вид сервісу, за яким сервіс може бути викликаний на виконання. Сервіс доступний через регєстр UDDI і виконується з використанням SOAP повідомлень. Для локального запуску сервісу використовується рефлективна функціональність.

Визначення керування в WorkFlow визначається як послідовність ролей і ім'я WorkFlow. Проте WorkFlow мають глобальні знання про процеси потоку даних. Ця інформація представлена у відповідних онтологіях. Використання онтологій для динамічного керування сервісами відрізняє даний підхід від інших. У даній моделі не існує централізованого диспетчера. Головна відповідальність WorkFlow полягає в ініціації потоку процесів і контролювання змін. Отже, потік керується через координацію ролей.

Загальна схема WorkFlow агентів може бути визначена як кортеж WMA: $WMA_R = (W_{ni}, Jr_i, Fr_i, E)$, де *Wni* – унікальне ім'я для WorkFlow, *Jri* – набір подій: $J = \{j1, j2, j3, . . .jn\}$; *ri* є атомна подія, яка запускає WorkFlow агента для ініціювання нового процесу, *Fri* – ім'я кінцевої ролі в WorkFlow, *E* – набір виключень WorkFlow: $E = \{e1, e2, e3, . . . en\}$.

Агенти мають програмуватися, так, щоб керувати помилками або виключеннями при виконанні WorkFlow. Припущенням в цьому підході є те, що керування для виключення WorkFlow атомарні. Керування для виключення WorkFlow визначається як кортеж: $e_i = (W_{ni}, V, Lr_i)$, де *Wni* і *Lri* – ім'я WorkFlow і ім'я провідної ролі, *V* – набір імен значень: $V = \{v1, v2, v3, . . . vn\}$. *Vi* визначається з ідентифікатором Value_name і Workflow. Проектувальником визначається код помилки або невідповідність значення даних.

$$v_i = (N_{vi}, D_i).$$

Таке інтегроване рішення забезпечує масштабність процесу створення та композиції сервісів.

4. Створення SWS прикладних застосувань

Поряд з технологіями та засобами представлення даних та знань в Семантичній мережі народжується нова парадигма створення та інтеграції прикладних застосувань, яка відповідає новим вимогам. Концепції процедурного, модульного і об'єктно-орієнтованого програмування – це приклади минулого, майбутнє ж, схоже, за сервісно-орієнтованою та агентно-орієнтованою архітектурою [12].

Архітектура застосувань Семантичної мережі складається з розподіленої системи інформаційних ресурсів (XML-документів, RDF-даних, OWL), опублікованих в мережі, інтерфейсів, що дозволяють людям та агентам використовувати і змінювати їх у межах своїх повноважень. Програмні застосування складаються з мережних сервісів, що представляють собою стандартизовані прикладні програми, які апробовані і спрямовані на вирішення певної бізнес чи наукової задачі, а також універсальних сервісів та агентів, які вирішують задачі керування, забезпечуючи цілісність даних і коректність функціонування програмного середовища. Окремою складовою архітектури є засоби, що забезпечують пошук та доступ до сервісів мережі, запуск сервісів, композиції та моніторингу виконання сервісів [14].

Позитивною якістю ідей, закладених в основу Семантичної мережі, є гнучкість і відносна простота базових блоків, з яких може бути створена інформаційна система. Архітектура SWS застосування в Семантичній мережі показана на рис. 2.

Відповідно до концептуальної схеми для функціонування SWS і WS необхідні чотири типи засобів, що дають змогу зовнішнім програмам або агентам знаходити, запускати та створювати нові сервіси шляхом композиції вже існуючих:

засоби доступу (комунікації), що описують протоколи комунікацій, такі як SOAP або HTTP, необхідні для забезпечення запуску сервісів на виконання;

дескриптивні засоби SWS, що описують джерела сервісів. Це певні довідники та класифікатори, необхідні для виконання функцій WS;

функціональні засоби специфікують потреби SWS, відповідно до вхідних та вихідних даних, і умови, які описують результати, отримані при виконанні даного сервісу;

структурні засоби описують внутрішню структуру композиційного сервісу, яка складається з структурних компонентів (складових сервісів) та інформації, як ці компоненти пов'язані між собою.

Ці засоби використовуються для композиції сервісів, так як вони визначають, чи існує зв'язок між складовою сервісу і другим сервісом, який використовується для композиції нового сервісу. Вони представляють різні, але комплементальні аспекти сервісів, які необхідні агентам для виконання операцій (запуску, пошуку/публікації, тощо). Наприклад, для запуску SWS агент має визначити як забезпечений доступ до сервісу, так і місце знаходження сервісу, вхідні та вихідні дані, проте йому не треба знати внутрішню структуру сервісу (як саме працює сервіс).

Висновки

У даній роботі були представлені основні підходи та засоби, що закладені в основу технології адаптивної композиції сервісів на основі агентної інтеграції в середовищі Семантичної мережі. Семантичні технології забезпечують існування певного рівня абстракції над існуючими інформаційними технологіями. Цей рівень дозволяє здійснювати зв'язок даних, семантики і процесів, що виконуються в розподіленому середовищі.

Розвиток цих технологій дасть змогу створювати єдине уніфіковане представлення SWS у широкому колі застосувань, дозволить точно знаходити необхідну інформацію, спрощувати корпоративну інтеграцію і інтеграцію мережних застосувань в розподіленому середовищі.

Розглянуто засоби та стандарти інтеграції та координації мережних сервісів на основі моделей потоку робіт.

1. *Sheila A. McIlraith, Tran Cao Son, and Honglei Zen.* Semantic Web Services, Stanford University. http://www.ksl.stanford.edu
2. *Workflow* Management Coalition standards // http://www.wfmc.org/standards/ standards.htm
3. *Hull R.*. Web Services Architecture, W3C Working Group Note 11, February, W3C Technical Reportsand Publications. http://www.w3.org/TR/ws-arch/, http://www.informatik.uni-trier.de/~ley/db/conf/icws/icws2005.html
4. *http://developers.sun.com/techtopics/webservices/wscaf/wscf.pdf*
5. *A Comparison* of XPDL, BPML and BPEL4WS. // http://xml.coverpages.org/Shapiro-XPDL.pdf
6. *Tim Berners-Lee, James Hendler and Ora Lassila*. The Semantic Web http://www.sciam.com/article. cfm?articleid =000A0919
7. *http://www.semwebcentral.org*
8. *OWL* Technical Committee. Web Ontology Language (OWL). http://www.w3.org/TR/2004/WD-owlref
9. *Asunción* Gómez-Pérez and Rafael González-Caber. ODE SWS: A Framework for Designing and Composing Semantic Web Services Technical University of Madrid. www.delicias.dia.fi.upm.es
10. *http://www-128.ibm.com/developerworks/websphere/zones/was/wpc.html*
11. *Калинченко Л.А*. Методология организации решения задач над множественными распределенными неоднородными источниками информации. ИПИ РАН. http://www.rfbr.ru/default.asp?doc_id=20786
12. *Wooldridge M.* – On the Sources of Complexity in Agent Design – In Applied Artificial Intelligence. – 2000. – 14(7). – P. 623 – 644.
13. *Workflow* Process Definitio. http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf
14. *Andon Ph., Deretsky V.* Control Oriented Ontology and Process Description for Cooperation Agents in Information Retrieval // Sixth International Scientific Conference „Electronic Computers and Informatics ECI'2004". − Kosice – Herlany: Slovakia. September 22-24, 2004. – P. 14 – 18.
15. *Web* Services. // http://www-306.ibm.com/software/solutions/webservices/uddi