УДК 004.7

О.А. Попхадзе

Національний технічний університет України «Київський політехнічний інститут», Київ

РОЗГЛЯД ПЕРСПЕКТИВНОЇ КОНЦЕПЦІЇ ПОБУДОВИ КОМПОЗИТНИХ ВЕБ-ДОДАТКІВ

У статті розглянута концепція побудови композитних веб-додатків. Визначені композитні веб-додатки, їх типи, архітектурні аспекти, проведено огляд технологій і протоколів, що полегшують створення mashup додатків, проведено аналіз існуючих композитних веб-додатків.

Ключові слова: *веб-технологія, композитні веб-додатки.*

Вступ

Рішенням для вирішення невеликих задач є використання веб-додатків через їх зручність, доступність з будь-якого пристрою. Але сервісів для таких цілей поки що мало, оскільки технології, що дозволяють їх проектувати, з'явились недавно, а кількість потенціальних користувачів – невелика.

Останнім часом веб-додатки витісняють інстальоване ПЗ, оскільки вони доступні з будь-якого комп'ютера, швидкість Інтернету дозволяє працювати без зайвих проблем, для їх функціонування потрібен лише браузер та вони не створюють ніякого навантаження на пристрій користувача, тому що всі вираховування проводяться на сервері. Поява композитних веб-додатків, які інакше називаються mashup, стала можливим завдяки сервіс-орієнтованій архітектурі програмного забезпечення. Mashup використовує інструментарії двох або більше сервісів та, використовуючи їх функціональність, створює новий. Розробка окремих незалежних компонентів з відкритим програмним інтерфейсом та поєднання їх у mashup має великий потенціал для використання у будь-яких сферах, такий підхід дозволяє легше розробити складний сервіс [1 – 5].

Мета статті – визначити характеристики перспективної концепції побудови композитних веб-додатків.

Результати досліджень

Поняття композитного веб-додатку. Композитний веб-додаток (mashup) – це веб-додаток, який використовує дані з більше ніж одного джерела для створення нового сервісу, що відображується одним графічним інтерфейсом. Наприклад, комбінуючи картографічні дані GoogleMaps з відгуками людей про подорожі до якихось місць, можна створити унікальний веб-сервіс з функціональністю, що не передбачало жодне джерело. Mashup створюється на основі веб-служб, які представляють розробникам інтерфейси прикладного програмування API. Зазвичай термін mashup застосовується тільки до тих проектів, які використовують відкриті інтерфейси API для отримання вище наведених послуг [1]. Іншими методами отримання інформації можуть бути веб-фіди (наприклад? RSS або Atom) або парсинг веб-сторінок. Побудова сервісів за такою концепцією базується на сервіс-орієнтованій архітектурі.

Типи композитних веб-додатків. Mashup можна розділити на три основні типи.

Mashup додатки користувача - об'єднують дані з різних відкритих джерел в браузері користувача. Користувач сам може брати участь у створення нових даних, приклад тому сайт earthalbum.com де об'єднуються картографічні дані з сервісу Google Map і фотографії з сервісу Flickr [2].

Mashup даних («enterprise» - mashup) змішують дані близькі за типом з різних джерел, наприклад, об'єднуючи дані з декількох RSS-фідів в один фід з графічним інтерфейсом. «Enterprise» - mashup зазвичай інтегрує дані із зовнішніх і внутрішніх джерел. Такий mashup може, наприклад, створювати звіт про зайнятої частини ринку, об'єднуючи зовнішній список всіх проданих за минулий тиждень будинків з внутрішніми даними про те, які будинки були продані окремим агентством [3].

Бізнес mashup додатки - створюються з використанням технології бізнес-бізнес (b2b). Добре підходять для швидкої розробки проектів, які вимагають співпраці розробників і замовників для визначення та реалізації бізнес-вимог. Збір та аналіз порівняльної інформації щодо ціни на торгових Інтернет-майданчиках дозволяє орієнтуватися в ціновому діапазоні, перебуваючи в одному веб додатку. eBay та Amazon – одні з перших компаній, які створили API для програмного доступу до свого контенту.

Інколи виділяють ще телекомунікаційний та навчальний mashup. Телекомунікаційний mashup – це телекомунікаційний сервіс, елементи якого зібрані з декількох джерел. Наприклад, хтось може отримувати базовий сервіс від компанії А, тон зворотного дзвінка від компанії Б, сервіс голосової пошти від компанії В тощо [4]. Навчальний mashup - («Training» -mashup) – це навчальний сервіс в Web,

© О.А. Попхадзе

який інтегрує дані з різних навчальних джерел в інтернеті.

Також mashup розрізняють за типом API, якій вони використовують, але вони можуть бути комбіновані між собою або інтегровані в інші сервіси. За типами використаних даних: індексовані дані; Картографічні або географічні дані; фіди, подкасти. За типами функцій: конвертери інформації; комунікації; візуалізація інформації; безпека; редактори.

Архітектурні аспекти. Архітектура будь-якого mashup складається з трьох основних частин, що не перетинаються між собою фізично та логічно:

Прикладний програмний інтерфейс. Провайдери даних. Це провайдери контенту, звідки береться інформація. Для полегшення обміну даними, провайдери зазвичай представляють контент через веб-протокли, такі як REST, SOAP та RSS/Atom. Тим не менш, багато потенціально цікавих джерел даних ще не представили власний API.

Mashup сайт. Це сервер на якому розміщується mashup та інформація від провайдерів контенту, інформація надходить за допомогою відкритих API. З одного боку, mashupи можуть працювати як традиційні веб-сервіси, використовуючи динамічну генерацію контенту на стороні серверу технологіями Java, CGI, PHP або ASP.

Зазвичай mashup використовують комбінацію серверної та клієнтської логіки для досягнення необхідної агрегації даних. Багато mashupів використовують дані, які поставляються напряму від їх бази користувачів, роблячи хоча б один з наборів даних локальним. Додатково, виконуючи комплекс запитів на різні ресурси, інформація потребує обчислень, що можуть бути виконані в браузері клієнта.

Клієнтський браузер, який за певних налаштуваннях браузера може генерувати інформацію як за мовними та регіональними налаштуваннями, так і за останніми пошуковими запитами. Завдяки цьому, при запуску браузера, він сам обирає потрібні налаштування веб-сервісів, позбавляючи користувача від зайвої інформації, незв'язаної з його мовними, регіональними та пошуковими запитами. Приклад реалізації налаштувань браузера добре спостерігаються в API GoogleMaps. При запуску браузера у користувача відображається інформація на його мові. Одним словом клієнтський браузер - це середовище, в якому додаток інтерпретується в графічному вигляді й взаємодіє з користувачем [5].

Огляд технологій і протоколів, що полегшують створення mashup додатків. Для того, щоб mashup додатки почали свою роботу, використовуються різні технології для їх реалізації:

Ajax - це скоріше модель веб-додатків, аніж специфічна технологія, складається з технологій:

- XHTML та CSS – результат співпраці між W3C та веб- додатків генеративної технології робочої групи WHATWG. Заснований на HTML, CSS, DOM, JavaScript;
- API моделі DOM, незалежна інтерфейсна модель - для динамічного відображення і взаємодії HTML;
- XML, розширювана мова розмітки - для асинхронного обміну даними;
- JavaScript - використання сценаріїв на стороні клієнтського браузера.

Використовуючи усі технології разом, досягається ціль створення гладкого, об'єднуючого веб-досвіду для користувача, замість повного перезавантаження та ри-рендеренгу сторінки йде обмін малими частками даних з серверами контенту. Враховуючи, що дані обмінюються між різними серверами і інтерпретуються на користувальницької машині, Ajax дозволяє це робити без перезавантаження браузера під час інтерактивної роботи користувача з веб-додатком.

Веб-протоколи **SOAP і REST** - це платформи незалежних веб-протоколів для зв'язку з віддаленими сервісами. Виступаючи в ролі частини сервіс-орієнтованої парадигми, клієнти можуть використовувати SOAP і REST для взаємодії з віддаленими сервісами без знання їх внутрішньої реалізації, функціональність сервісу цілком виражається описанням запитів та відповідей

SOAP - протокол доступу до об'єктів, фундаментальна технологія парадигми веб-сервісів. SOAP – це акронім Services-Oriented Access Protocol. Використовується для обміну довільним повідомленнями у форматі XML незалежно від платформи. Структура його повідомлень складається з заголовку та тіла запиту. У заголовку запиту міститься контекстна інформація, що не відноситься до корисної інформації для додатка, наприклад інформація для аутентифікації. Тіло запиту містить специфіковану інформацію для сервісу. SOAP API для веб-сервісів , що описується WSDL документами, які описують які операції розкриває сервіс, формат повідомлень які приймає, і як їх адресувати.

REST - архітектурний стиль програмного забезпечення для розподілу систем, таких як World-WideWeb, який використовується для побудови веб-служб допомогою HTTP і XML. REST – це акронім RepresentationalStateTransfer. На відміну від типових інтерфейсів, що базуються на різних наборах запитів для різних даних, які ви можете знайти у сучасних мовах програмування, REST підтримує лише декілька операцій, а саме POST, GET, PUT, DELETE, що приємні для усіх типів даних. Виразність у REST є однією з частин інформації самої по собі, званим ресурсом. Наприклад, ресурс співбесіди для майбутнього співробітника ідентифікується URI, повертається через GET, оновлюється через PUT і так далі. REST подібний до документально-

буквального стилю сервісів SOAP. Клієнти можуть використовувати SOAP і REST для взаємодії з віддаленими сервісами. Функції сервісу повністю передаються через опис повідомлень, за допомогою яких здійснюються запити і відповіді [5].

Як було зазначено вище, недостатність API від провайдерів контенту часто змушувало mashup-розробників парситивеб-сторінки для отримання необхідної інформації для композиції. Процес полягає в використанні спеціального ПЗ для парсингу та аналізу даних, що початково написані для сприйняття людиною, перетворення семантичних структур даних у форму, що якою може маніпулювати і використовувати програмно. Невелика кількість mashup використовують такий підхід для отримання необхідних даних, особливо з публічних ресурсів.

Парсинг сторінок вважається неелегантним рішенням, і не без підстав. Він має декілька невід'ємних недоліків. Перший, на відміну від даних отриманих з API, парсинг не має структурованої форми обміну інформацією між провайдером контенту і клієнтом. Клієнт має підлагоджувати власні інструменти під джерело контенту і сподіватися, що провайдер дотримувався саме цієї моделі відображення. Веб-сайти мають тенденцію до оновлення їх вигляду періодично, щоб залишатися свіжими і стильними, що доставляє велику головну біль через падіння системи парсингу.

Другий недолік – це недостача складного, повторно використовуваних інструментів для парсингу та аналізу даних. Такі інструменти підлаштовуються індивідуально під кожен сервіс. Це призводить до того, що розробники змушені деконструювати контент, розробляти моделі даних, парсити і агрегувати необроблені дані з сайту провайдера контенту [6].

Неелегантність аспектів парсингу відслідковується від факту, що контент створений для сприйняття людиною, а це в свою чергу робить його не пристосованим до автоматизованим сприйняттям комп'ютером. Рішенням є Семантичний веб, який передбачає, що контент сучасного веб може бути доповнений до стану, в якому дані є прийнятним для сприйняття людиною з еквівалентною інформацією, яку здатен сприймати комп'ютер. В цьому контексті Семантичного веб поняття інформації є відмінним від поняття даних, дані становляться інформацією у момент коли вони виражають суть, зміст. Семантичний веб має мету створення веб-інфраструктури, яка поповнює дані метаданими, щоб дати їх зміст, роблячи таким чином їх прийнятними для інтеграції і використання повторно.

Консорціум W3 створив набір специфікацій відомий як RDF. RDF – це акронім Resource Description Framework. Він служить для досягнення цілі забезпечення методології для встановлення синтаксис-структур, які описують дані. XML самого по собі недостатньо, він занадто довільний, ви можете закодувати одні і ті самі дані різними шляхами. RDF-схема додає до RDF можливість закодувати концепти у читабельний для комп'ютера спосіб. Як тільки об'єкти даних можуть бути описані в моделі даних, RDF забезпечує створення відношень між об'єктами даних через відношення суб'єкт-предикат-об'єкт. Комбінація моделей даних і графа відношень дозволяє створення онтологій, що є ієрархічними структурами інформації, які можуть бути знайдені і номінально обґрунтовані. Наприклад, ви можете визначити модель в якій «хижаки» підклас «тварин» з властивістю «з'їсти» іншу «тварину» і створити два екземпляри: перший наповнений даними описуючими гепардів і полярних ведмедів та їх звички, інший описує газелей та пінгвінів і відповідні їх звички. Система висновків може «замашурити» ці окремі джерела і зрозуміти, що гепарди полюють на газелей, але не на пінгвінів.

RDF дані - це швидкий пошук необхідних даних різноманітті доменів, включаючи соціальні мережі і синдикати, такі як RSS. До того ж, ПЗ для технології RDF та його компоненти дозволяють досягти високого рівня завершеності, особливо при використанні мов запитів RDF, програмних фреймворків та невід'ємних двигунів.

Atom і RSS - це сімейство форматів синдикації на основі XML. RSS дозволяє за допомогою онлайн сервісів і різних агрегаторів, і каталогів імпортувати і експортувати інформацію, що дозволяє користувачу збирати потрібну для нього інформацію в зручному для нього відео з різних сайтів.

Формат Atom, який врахував недоліки RSS і реалізує обмін інформацією заснований на XML [2].

Ці технології синдикації добре підходять для mashup які агрегують контент, що постійно оновлюється, такий як новини або веб-блоги.

Проблеми інтеграції даних. Семантичне значення і якість даних. Опитування якості показують, що основний пріоритет IT підприємств є інтеграція даних в межах підприємства віртуальної організації. (У цьому контексті, я використовую термін віртуальна організація, означає складу об'єднання бізнес-одиниць, кожна міститься у його власному адміністративному домені.). Як і багато керівників підприємств IT, які знаходять себе в завдання інтеграції джерел даних (наприклад, створити корпоративні інформаційні панелі, які відображають поточні умови бізнесу), розробники mashupів стикаються з аналогічними проблемами отримання загального семантичного змісту між гетерогенними наборами даних.

Наприклад, системи переводу між моделями даних повинні бути розроблені. Коли конвертується інформація в звичайні форми, мають бути створені

припущення щодо певних параметрів, якщо відображення не є повним. Крім того це може бути ускладнено фактом, що розробник mashupy не є експертом у моделях джерела даних, а тому припущення можуть не бути інтуїтивними і зрозумілими.

В добавок до відсутніх даних або не повних відображень , розробник композитного веб-додатку може зустрітися з тим, що інформація, яку він хоче інтегрувати не є прийнятною для автоматизованого сприйняття комп'ютером, вона має бути підготовлена. Семантичні технології моделювання, такі як RDF, мають допомагати полегшити проблеми автоматичного сприйняття між різними наборами даних. Спадкові набори даних потребують більших людських зусиль в аналізі і підготовці даних перед тим як до них можуть бути застосовані семантичні технології моделювання. Частиною розробки будь-якого додатку є забезпечення вводу інформації користувачем. Це ніж з двома гранями, з однієї сторони це дуже потужна можливість, що забезпечує відкритий внесок в роботу, з іншої інформація може бути не коректною, не повною і т.д. Це визиває велику кількість сумнівів у введених даних, а mashup має забезпечити коректну їх інтерпретацію [5].

Проблеми компонентів. Модель Ajax розробки може забезпечити значно кращий і багатший досвід використання користувачем, а ніж повне перезавантаження сторінки, але вона також вносить додаткові складнощі. Фундаментально, Ajax використовує сценарні можливості браузеру клієнта і поєднанні з його DOM для досягнення методів отримання контенту, який не цілком передбачається дизайнерами браузеру. Тим не менш, суб'єкти сервісів побудованих за Ajax вирішують проблеми сумісності інформації, ще з часів Internet Explorer .

Використання JavaScript для асинхронного оновлення контенту сторінки можуть спричинити помилки у інтерфейсі користувача. Через те, що контент не зв'язаний безпосередньо з URL у панелі адреси браузеру, користувачі можуть не отримати функціонал зазвичай очікуваний при використанні кнопки повернення в браузері, або створення закладки. Також Ajax підвищує затримку відправляючи запити для оновлення контенту, поганий дизайн може справити погане враження на користувача, тому дизайнер має піклуватися про створення панелей прогресу під час таких операцій.

Розробники mashup та провайдери контенту мають забезпечити передачу захищених даних, як і для будь-якого розподіленого, кросс-доменого сервісу. Поняття ідентифікації може виявитися проблемним, бо традиційний веб в першу чергу створений для анонімного доступу. Одноразовий вхід без аутентифікації в систему є бажаним, але є безліч конкуруючих технологій (від Microsoft Passport до LibertyAlliance), таким чином, створюючи розрізнені простори імен, що необхідно інтегрувати також. Контент-провайдери, швидше за все, використовують схеми аутентифікації і авторизації (які вимагають поняття безпечної ідентифікації або надійно ідентифікованих атрибутів) у своїх API, для забезпечення бізнес-моделі, які включають платні підписки або конфіденційних даних. Конфіденційні дані, ймовірно, також вимагають шифрування, і розробник повинен дбати, коли ви компонуєте їх з іншими джерелами, щоб не скомпрометувати їх. Крім того, інтеграція даних з відбувається як на сервері і на стороні клієнта, тому ідентичність та облік даних при передачі даних від користувача до mashup-сервісу може стати вимогою [5].

Аналіз існуючих композитних веб-додатків. Використання mashup- технології стає все ширшим кожен день, тому сформувалося декілька категорій і типів таких веб-сервісів. Ми можемо виділити чотири основних: картографічні, мультимедійні, пошукові та для шопінгу, новинні. Наукові mashup не попадають в окрему категорію, бо їх процент занадто малий і вони внесені в категорію Інші. Проаналізуємо найпопулярніші категорії.

Картографічні композитні веб-додатки є найпопулярнішими поміж існуючих. Вони потребують джерела карт, що забезпечить графічне відображення території, що відноситься до процесу. Коли найпопулярніші в світі карти GoogleMaps відкрили API – це дозволило веб-розробникам легко інтегрувати карти у власні сайти. Цей крок мав великий вплив на розвиток mashup і одразу за Google свої API представили такі сервіси як YahooMaps, MapQuest, Microsoft'sVirtualEarth, роблячи процес використання карт будь-якого рівня тривіальним.

Ось кілька прикладів mashup карт:

- Places.ae – це веб-програма для пошуку практично будь чого Об'єднаних Арабських Еміратах. Вона використовує GoogleMaps та Twitter.
- Gis.chicagopolice.org – це mashup, що інтегрує власну базу злочинів з GoogleMaps для запобігання злочинам у територіях і попередження громадян про їх місця.
- Iamcaltrain.com – це mashup на базі Yahoo Maps. За допомогою розкладів потягів та фотографій сервісу Flickr він допомагає спланувати подорожі Каліфорнією. Інформує про особливості зупинок потягів.

Фотографії та відео mashup базуються на відповідних провайдерах контенту та інших джерел даних, що можуть бути віднесені до відповідного мультимедійного контенту. Цей контент включає місця та локації де були зроблені відповідні фото та відео, що дозволяє відобразити ці місця на карті. Існують багато API для фото та відео, такі як Flickr, YahooPhotos, Youtube та інші. Ось приклади відео та фото mashup-сервісів:

- ReelzReview.com – ціль роботи цього веб-додатку – скомпонувати інформацію про певний фільм, щоб допомогти користувачу вирішити, чи варто купувати певний фільм, чи ні. Він використовує відкритий API Amazon, YouTube, Yahoo, Ebay та інших ресурсів, щоб отримати зв'язані дані про нові релізи, огляди, ціни, пропозиції на одній веб-сторінці.
- FlickrMaps це mashup карт Yahoo та Flickr який перетворює вас у віртуального туриста. Оберіть місто і ви отримаєте відповідні фото з Flickr.

Композитні веб-додатки для шоппінгу існували задовго до того, як виникло поняття mashup. Замість використання API, існували порівняльні інструменти, такі як BizRate, PriceGrabber, Google'sFroogle, які використовували комбінації business-to-business технологій для агрегації даних для порівняння.

Ідея mashup для пошуку та шоппінгу в порівнянні цін і специфікацій об'єктів пошуку різними методами. Дані з декількох джерел відображаються разом для впевненості користувача.

- iPodRadar.com – це композитний веб-додаток для шоппінгу, який допомагає знайти iPod та аксесуари до нього. Окрім того, він повідомляє користувачів про новини пов'язаних блогів, посилань, програмного забезпечення та іншого. Він використовує API Google, AmazoneCommerce, BackPack, eBay, GoogleAdWords та Technorati API.

Композитні веб-додатки для новин полягають у тому, що вони відображують лише ті категорії новин та з тих ресурсів, які користувач хоче бачити. Це щось накшалт створення власної газети відповідно до його інтересів. У цих випадках використовуються технології RSS та ATOM, про які писалося вище. Приклади таких сервісів:

- Dogghot.us комбінує IT новини з Digg, Slashdot та del.icio.us, об'єднує або видаляє подібні записи та додає кілька власних особливостей.
- Aggreget.com індексує багато сторінок з таких ресурсів як Digg, Stumble, Delicious та подібних та відображає топ-10 популярних посилань, що перетинаються на цих ресурсах між собою, видаючи дійсно найпопулярніші теми.

Висновки

Mashup – це комбінація технологій моделювання, що відслідковуються від семантичного вебу та сервіс- орієнтованими, платформо-незалежними архітектурами і протоколами обміну даними, що забезпечують інструментами, необхідними для розробки сервісів, що можуть оперувати великими масивами даних, що доступні у веб. Композитні веб-додатки забезпечують і потребують кращу прозорість роботи, цікаво спостерігати за тим як цей клас сервісів впливає на права використання та інтелектуальну власність, тоді як інші сервіси інтегрують дані в межах організацій, наприклад грід-обчислення та business-to-business управління.

Список літератури

1. Things You Should Know About Mapping Mashups [Електронний ресурс]. Режим доступу: http://www.educause.edu/library/resources/7-things-you-should-know-about-mapping-mashups. – Дата доступу: 10.03.2015.

2. Немного о Mashup [Електронний ресурс].– Режим доступу: http://digipo.eu/nemnogo-o-mashup.html. – Дата доступу: 21.04.2015.

*3. Personalblog: WhyMashups = (REST + 'Traditional SOA') * Web 2.0: [Електронний ресурс].– Режим доступу: http://blog.sherifmansour.com/?p=187. – Дата доступу: 28.05.2015.*

4. WEBMASHUP: [Електронний ресурс].– Режим доступу: http://www.webmashup.com/. – Дата доступу: 14.05.2015.

5. MerrillD. Mashups: Thenewbreedofwebapp. / DuaneMerrill / [Електронний ресурс].– Режим доступу: http://www.ibm.com/developerworks/library/x-mashups/. – Дата доступу: 23.04.2015.

6. Mashup-приложения - эволюция SOA: Часть 2. Ситуативные приложения и mashup-экосистема [Електронний ресурс] / Стивен Уатт. – Режим доступу: http://www.ibm.com/developerworks/ru/library/ws-soa-mashups2/. – Дата доступу: 09.03.2015.

7. PROGRAMMABLE WEB RESEARCH CENTER: [Електронний ресурс]. – Режим доступу: http://www.programmableweb.com/api-research. – Дата доступу: 15.02.2015.

Надійшла до редколегії 30.03.2016

Рецензент: д-р техн. наук, проф. О.О. Можаєв, Національний технічний університет «ХПІ», Харків.

РАССМОТРЕНИЕ ПЕРСПЕКТИВНОЙ КОНЦЕПЦИИ ПОСТРОЕНИЯ КОМПОЗИТНЫХ ВЕБ-ПРИЛОЖЕНИЙ

А.А. Попхадзе

В статье рассмотрена концепция построения композитных веб-приложений. Определены композитные веб-приложения, их типы, архитектурные аспекты, проведен обзор технологий и протоколов, облегчающие создание mashup приложений, проведен анализ существующих композитных веб-приложений.

***Ключевые слова**: веб-технология, композитные веб-приложения.*

REVIEW POLICY CONCEPT OF MASHUP WEB APPLICATION CONSTRUCTION

O.A. Popkhadze

In the article the concept of building mashup Web applications. Defined mashup web applications, their types and architectural aspects of the review of protocols and technologies that facilitate create mashup applications analysis of existing mashup Web applications.

***Keywords:** web technology, mashup Web applications.*