УДК 004.72:004.75

Ю.Д. Свистунов

Національний технічний університет «Харківський політехнічний інститут», Харків

МЕТОДИ ОРГАНІЗАЦІЇ ВЗАЄМОДІЇ РОЗПОДІЛЕНИХ КОМП'ЮТЕРНИХ СИСТЕМ НА ОСНОВІ СЕРВІС-ОРІЄНТОВАНОЇ АРХІТЕКТУРИ

Обговорюються методи та підходи до інтеграції програмно-апаратних засобів в розподілену систему підприємства. Визначені поняття сервіс-орієнтованої архітектури та її основних компонентів. Обґрунтована побудова систем на основі сервіс-орієнтованої архітектури, та проаналізована надійність таких систем Отримані результати дозволяють стверджувати, що за допомогою веб-сервісів можна підвищити ефективність та надійність процесів обробки даних у слабко пов'язаних системах масштабу підприємства.

Ключові слова: *Сервіс-орієнтована архітектура, розподілені системи, Web-сервіс.*

Вступ

Постановка проблеми. Інформаційно-обчислювальна інфраструктура сучасного підприємства налічує сотні програмних додатків, що реалізують бізнес-процеси, що протікають як всередині окремої компанії, так і зв'язують окремі ланцюжки бізнес-процесів різних компаній або їхніх філій в єдиний бізнес-процес. Всі ці програми розроблені в різний час і різними людьми. Природно, що архітектура, мова програмування і програмна середа виконання кожного з додатків різні, що істотно ускладнює інтеграцію програмних додатків інформаційних систем підприємств у єдину інформаційно-обчислювальну середу. Термін інтеграція може мати достатньо широке визначення і означати об'єднання комп'ютерних систем, компаній і людей. У загальному випадку можна сказати, що інтеграція має на увазі забезпечення взаємодії між безліччю програм і програмних систем, що виконуються під управлінням різних програмно-апаратних платформ, розташованих в територіально віддалених місцях.

З часом підприємства розвиваються, розширюється спектр їхніх послуг, збільшується число партнерів, що спричиняє появу нових цілей і завдань, і, як наслідок, розробку, або закупівлю необхідних програмних додатків. Основне завдання інтеграції полягає у налагодженні функціональних зв'язків між додатками різних інформаційних систем.

Розподілені системи всередині підприємства є найважливішими компонентами їх інформаційного простору, але одночасно з широкими можливостями, що відкриваються перед компаніями з організації бізнес-процесів як всередині, так і зовні підприємства, з'явилася необхідність створення надійних і ефективних засобів програмної реалізації таких систем.

Метою роботи є підвищення ефективності та надійності процесів обробки даних у слабко пов'язаних системах масштабу підприємства за рахунок впровадження сервіс-орієнтованої архітектури.

1. Існуючі способи інтеграції розподілених систем

На сьогоднішній день існує три переважаючих способу інтеграції програмно-апаратних засобів в розподілену систему підприємства. І першим з них є використання клієнт-серверної архітектури.

Клієнт-серверна архітектура - обчислювальна або мережева архітектура, в якій завдання або мережеве навантаження розподілені між постачальником послуг, що зветься сервером, і замовниками послуг, що називаються клієнтами. Фізично клієнт і сервер - це програмне забезпечення. Зазвичай вони взаємодіють через комп'ютерну мережу за допомогою мережевих протоколів і знаходяться на різних обчислювальних машинах, але можуть виконуватися також і на одній машині. Програми - сервера, очікують від клієнтських програм запити і надають їм свої ресурси у вигляді даних (наприклад, завантаження файлів за допомогою HTTP, FTP, BitTorrent, потокове мультимедіа або робота з базами даних) або сервісних функцій (наприклад, робота з електронною поштою, спілкування за допомогою систем миттєвого обміну повідомленнями, перегляд web-сторінок) [6].

До недоліків клієнт-серверних систем можна віднести те, що сервер є єдиною точкою збою. Це єдиний компонент, що надає клієнтам можливість обробки їх запитів і виконання завдань, у той час, як клієнти - взаємозамінні, і можуть підключатися і відключатися від сервера незалежно. Якщо сервер, з якихось причин, перестане працювати, то ціла система теж зупинить своє виконання. Таким чином, функціональна абстракція, створена клієнт-серверною архітектурою, теж робить систему вразливою до відмов. Іншим недоліком такої архітектури є те, що ресурси сервера обмежені, і якщо до нього буде підключено забагато клієнтів, то це теж може негативно позначитися на роботі цілої системи, не виключаючи її відмову.

Другим способом організації розподілених обчислювальних систем є архітектура «peer-to-peer». Це такий варіант архітектури системи, в основі якої стоїть мережа рівноправних вузлів.

Комп'ютерні мережі типу peer-to-peer (або P2P) засновані на принципі рівноправності учасників і характеризуються тим, що їх елементи можуть зв'язуватися між собою, на відміну від традиційної архітектури, коли тільки окрема категорія учасників, яка називається серверами може надавати певні сервіси іншим.

У P2P мережі не існує поняття клієнтів і серверів, тільки рівні вузли, які одночасно працюють як клієнти і сервери по відношенню до інших вузлів мережі. Ця модель мережевої взаємодії відрізняється від клієнт-серверної архітектури, в якій зв'язок здійснюється тільки між клієнтами і центральним сервером. Така організація дозволяє зберігати працездатність мережі при будь-якій конфігурації доступних її учасників. Однак практикується використання P2P мереж які все ж мають сервери, але їх роль полягає вже не в наданні сервісів, а в підтримці інформації з приводу сервісів клієнтами мережі. Вони також допомагають мережі залишатися пов'язаною, зберігають інформацію з приводу місця розташування інших комп'ютерів в мережі і допомагають новим комп'ютерам, що підключилися швидше налагодити обчислювальний процес.

У P2P системі автономні вузли взаємодіють з іншими автономними вузлами. Вузли є автономними в тому сенсі, що не існує загальної влади, яка може контролювати їх. У результаті автономії вузлів, вони не можуть довіряти один одному і покладатися на поведінку інших вузлів, тому проблеми масштабування і надмірності стають важливішими, ніж у випадку клієнт-серверної архітектури.

Сучасні P2P-мережі отримали розвиток завдяки ідеям, пов'язаними з обміном інформацією, що формувалися в руслі того, що кожен вузол мережі може надавати і отримувати ресурси, які надаються будь-якими іншими її учасниками. В якості ресурсів тут може виступати не тільки інформація і дані, а й обчислювальні ресурси, такі як процесорний час і пам'ять, які служать для розподілених обчислень.

Більшість додатків однорангових мереж це додатки для передачі та зберігання даних. Кожен комп'ютер може бути проміжною ланкою між відправником та одержувачем, отже, теж безпосередньо брати участь у передачі інформації по мережі. У випадку з зберіганням даних, обсяг даних, необхідний для зберігання може бути занадто великим для одного комп'ютера, або занадто значущим, щоб зберігати його в одному місці. У такому випадку, кожен комп'ютер зберігає частину даних і може бути таке, що кілька копій одних і тих же даних зберігаються на різних комп'ютерах. У такому випадку, коли комп'ютер виходить з ладу, або просто відключається від мережі, дані можуть бути з легкістю відновлені.

Найбільш популярним підходом до інтеграції програмно-апаратних засобів підприємств є створення слабкозв'язаних систем [8]. Термін слабке зв'язування, як і його переваги по відношенню до створення сильно зв'язаних систем, відомий вже давно. Основним принципом слабкого зв'язування є мінімізація числа припущень, які роблять розробники компонентів і додатків взаємодіючих сторін. Безумовно, наявність детальної інформації про протоколи і формати даних дозволяє підвищити ефективність взаємодії, але одержане при цьому сильно пов'язане інтеграційне рішення вкрай чутливо до змін.

2. Сервіс-орієнтована архітектура: визначення та основні компоненти

З появою сервіс-орієнтованої архітектури (англ. Service-Oriented Architecture, SOA), технології веб-сервісів і стрімким розвитком Інтернет, перспектива створення глобально-розподілених слабкозв'язаних систем стала реальною. В основі даної архітектури лежать принципи багатократного використання програмних компонентів, що дозволяють виключити дублювання функціональності програмного забезпечення [7]. З погляду розробки та інтеграції інформаційних систем сервіс-орієнтована архітектура є парадигмою організації та використання бізнес-процесів і ресурсів, що належать різноманітних галузей виробничої та економічної діяльності. Вона дозволяє структурувати розгортання та організацію інформаційної технології в рамках конкретного підприємства або консорціуму підприємств, яким необхідно взаємодіяти один з одним. Тим не менше, основні концепції сервіс-орієнтованого підходу, як правило, реалізуються для надання електронних послуг за допомогою веб-сервісів, що взаємодіють один з одним через Інтернет, на відміну від бізнес-послуг, які можуть бути виконані вручну. У зв'язку з цим, SOA також можна розглядати як принцип для розробки архітектури програмного забезпечення, який будується навколо поняття веб-сервісу [12].

Веб-сервіси (Веб-служби) - це програмні компоненти, за допомогою яких можна створювати незалежні масштабовані слабкозв'язані додатки. В основі технології веб-сервісів лежить процес обміну повідомленнями у форматі XML-документів. Передача повідомлень відбувається з використанням протоколів HTTP, XML, XSD, SOAP, WSDL, UDDI. Специфікація визначає три основні стандарти, що використовуються для підтримки уявлення, пошуку та обміну інформацією між веб-сервісами - це

WSDL, UDDI і SOAP, які утворюють так званий «трикутник SOA» [9].

Процес взаємодії між клієнтом і постачальником веб-сервісу представлений на рис 1.

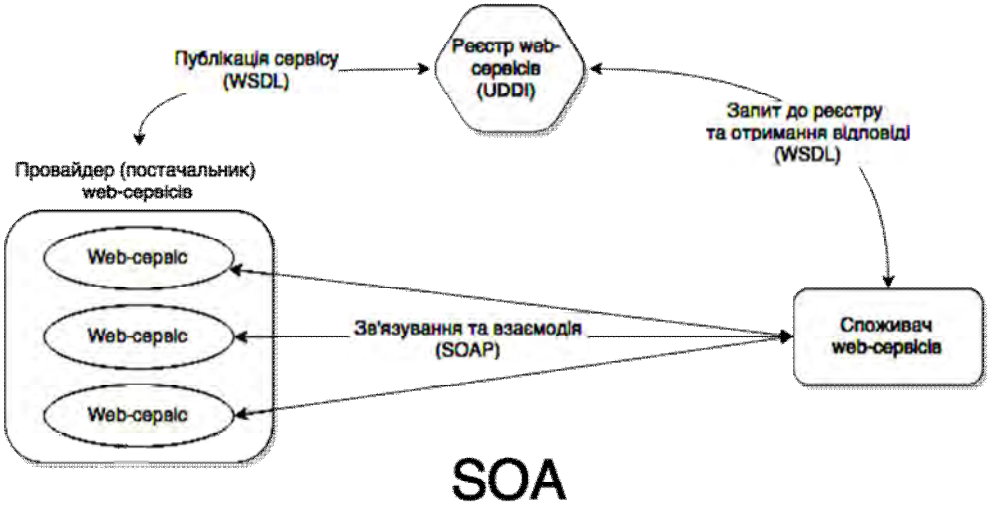

Рис. 1. Взаємодія користувача і провайдера веб-сервісу

Протокол SOAP (Simple Object Access Protocol) призначений для організації взаємодії віддалених систем за допомогою асинхронного обміну XML-відформатовані документами, що складаються з трьох частин: конверта (обгортки), заголовка і тіла. SOAP формує базовий шар стека протоколів веб-сервісів, забезпечуючи інфраструктуру обміну повідомленнями між ними.

Мова WSDL (Web Services Description Language) описує сервіси у вигляді деяких абстрактних ресурсів, здатних приймати на вхід документи певних типів та ініціювати відправлення документів інших типів. WSDL використовується для опису веб-сервісів і для визначення їх розташування. WSDL визначає сервіс з двох точок зору: абстрактної і конкретної. На абстрактному рівні сервіс задається в термінах повідомлень, що він відправляє та приймає, які описуються засобами XML Schema у вигляді, незалежному від конкретного транспортного протоколу. На конкретному рівні визначаються прив'язки до транспортних форматів і точок фізичного розміщення.

Група специфікацій WSDL 2.0 складається з трьох основних документів – WSDL Part 1: Core language («Основна мова»), WSDL Part 2: Message exchange patterns («Шаблони обміну повідомленнями»), WSDL Part 3: Bindings («Прив'язки»).

Протокол UDDI (Universal Description Discovery & Integration) являє собою стандарт на внутрішній устрій і зовнішні інтерфейси бази даних (репозиторію), що зберігає опис сервісів. Всі описи в БД зберігаються у вигляді XML-записів. Остання версія UDDI забезпечує реплікацію репозиторіїв зі складними моделями їх підпорядкованості один одному, побудова сховища з декількох вузлів (і реплікацію даних між ними), глобальну унікальність записів і ключів, API публікації описів і підписки на зміни, засоби забезпечення цілісності даних, інтернаціоналізації записів, шифрування вмісту. У той час як версія UDDI 2.0 призначалася для підтримки каталогів електронного бізнесу, версія 3.0 орієнтована і на внутрішньо корпоративне використання для побудови корпоративних систем в рамках ідеології сервіс-орієнтованої архітектури. Тому вона допускає створення реєстрів декількох типів (загальнодоступний, приватний і з розділеним доступом).

Архітектуру веб-сервісів становить маса протоколів і специфікацій [13]. Їх можна розбити на чотири частини (процес, опис, повідомлення, зв'язок), що утворюють стек протоколів, в якому кожен верхній рівень спирається на нижній рівень (рис. 2).

Рис. 2. Стек протоколів web-сервісів

3. Обґрунтування побудови систем на основі сервіс-орієнтованої архітектури

Як правило, кожен логічний веб-сервіс (позначений як i) розгортається у бізнес-середовищі щоб забезпечити корисну функціональність F_i, виражену у вигляді програмного інтерфейсу I_i. Важливою можливістю сервісу є його здатність до динамічної взаємодії в даному середовищі з іншими сервісами та сутностями, що не є сервісами (наприклад: кінцеві користувачі) [9].

Реалізація логічного веб-сервісу – набір скоординованих взаємодіючих процесів:

$$S_s = \left\langle P_1^i, P_2^i, \ldots, P_n^i, \Lambda \right\rangle, \quad (1)$$

де S_i – логічний екземпляр сервісу; P_k^i – k-й процес, що реалізує функціональність веб-сервісу F_i за допомогою програмного інтерфейсу I_i; Λ – мережева функція зв'язку між окремими процесами.

Процеси сервісу S_i можуть працювати:

- на одному процесорі; в цьому випадку логічний сервіс ідентичний екземпляру фізичного процессу, або $S_i = P_1^i$, та функція координація мережі Λ дорівнює нулю.
- на мультипроцесорному хості, в цьому випадку конкретний логічний сервіс містить в собі набір внутрішніх паралельних фізичних процесів, де функція Λ визначена апаратним та програмним забезпеченням поточного хосту.
- у гетерогенних розподілених системах, де функція Λ реалізована на основі протоколів загального призначення, нейтральних для апаратного та програмного забезпечення [2].

Сервіс-орієнтоване бізнес-середовище складається із закінченого числа всіх доступних реалізацій логічних сервісів на даний момент часу τ:

$$\mathrm{Env}(\tau) = \langle S_1, S_2, \ldots, S_n \rangle, \quad (2)$$

де n – кількість логічних сервісів розгорнутих у середовищі.

Звичайно, середовища сервісів можуть бути класифіковані за різними показниками, наприклад:

- на основі протоколу зв'язку (HTTP, або FTP);
- на основі контенту сервісу.

В цілому функціональність F сервіс-орієнтованого додатку A визначається логічними сервісами, залученими для забезпечення роботи додатку у даному середовищі на поточний момент часу τ:

$$F_A = \left\langle S_1^A, S_2^A, \ldots, S_n^A \right\rangle. \quad (3)$$

Більше того, можна побудувати графік направленості функціональності додатку, визначений таким чином:

$$V_A = (F_A, G), \quad (4)$$

де маємо вершини із набору F_A та набір граней G, що формалізують координацію між окремими логічними сервісами з набору F_A.

Нарешті, сервіс-орієнтований додаток A може бути формалізований у вигляді такого набору:

$$A = \langle F_A, V_A \rangle \quad (5)$$

Визначений сервіс-орієнтований додаток A характеризується властивостями, котрі наводяться нижче.

Для досягнення мети обчислення, повинно бути залучено щонайменш два логічні сервіси (в іншому випадку (СОА деградує у клієнт-серверну архітектуру) [11].

Сервіси, залучені у систему, повинні координувати їх роботу для вирішення мети обчислення. Під координацією мається на увазі будь-який тип взаємодії між залученими сервісами для досягнення цілей додатку. Координація може бути реалізована різними способами, включаючи, але не обмежуючись, обмін даними, обмін повідомленнями, резервування сервісів, управління сервісами, моніторинг сервісів та ін.

Наявність можливості координації вимагається сервіс-орієнтованим додатком у зв'язку із фактом, що послідовна реалізація сервісу повинна бути контекстно-інваріантною, тобто, конкретний сервіс не повинен знати про додаток, або систему, у яку він залучений.

Графік направленості функціональності додатку V_A, визначений у (4), є формалізацією таких можливостей.

У конкретному додатку V_A може бути вираженим за допомогою існуючих бібліотек для управління процесами сервісів, таких як BPEL, та WS-Coordination.

4. Надійність функціонування слабкозв'язаних сервіс-орієнтованих систем

При організації взаємодії інформаційних систем за допомогою веб-сервісів для залучення в бізнес-процес різних підприємств, орієнтованих на електронні форми ведення бізнесу, особливо важливим є питання забезпечення надійності функціонування виникаючих при цьому слабкозв'язаних сервіс-орієнтованих систем. У порівнянні з традиційними автономними системами програмного забезпечення створення надійних сервіс-орієнтованих систем набагато складніше з наступних причин:

- веб-сервіси розосереджені по Інтернет;
- веб-сервіси розроблені і розміщені різними власниками без узгодження будь-яких внутрішніх деталей розробки та реалізації;

- продуктивність веб-сервісів може мінятися (наприклад, з причини змін робочого навантаження серверів, внутрішніх оновлень веб-сервісів, зміни продуктивності каналів передачі даних);
- веб-сервіси можуть стати недоступними без будь-якого попереднього повідомлення.

Оскільки ініціатор формування бізнес-процесу відповідає тільки за належне йому програмне і апаратне забезпечення, питання забезпечення надійності функціонування може бути вирішене будь-якими доступними йому і добре всім відомими методами і засобами самостійно [4]. Те ж справедливо і для інших учасників динамічно утворюючихся при цьому композиції веб-сервісів. Забезпечення ж надійності функціонування динамічно мінливої композиції веб-сервісів і утвореної ними слабкозв'язаної системи в цілому не входить до компетенції жодного з учасників і повністю лягає на ініціатора утвореної композиції.

На надійність функціонування сторонніх веб-сервісів ініціатор композиції вплинути не може, а вплинути на надійність композиції в цілому може, забезпечивши надмірність функціональних можливостей веб-сервісів композиції за рахунок резервування функціонально подібними веб-сервісами.

Для перемикання в разі відмови основного веб-сервісу на резервний веб-сервіс (резервні веб-сервіси) необхідні програмні засоби, що реалізують стратегію відмовостійкості - сукупності методів резервування і алгоритмів вибору резерву [6].

Існуючі методи резервування програмних компонентів (веб-сервісів) сервіс-орієнтованих систем можна умовно розділити на пасивні та активні методи. Пасивні методи використовують резервні веб-сервіси тільки в разі відмови від обслуговування основного веб-сервісу - резервування заміщенням. Активні методи використовують різні варіанти паралельного виклику резервних веб-сервісів з безлічі функціонально подібних - постійне резервування. Змішане резервування забезпечує виконання необхідної функції кількома різними методами резервування веб-сервісів.

Належний рівень надійності функціонування композиції веб-сервісів і слабко пов'язаних сервіс-орієнтованих систем в цілому може бути забезпечений гнучкою стратегією відмовостійкості - сукупністю методів резервування і алгоритмів вибору резерву, що враховують вимоги користувачів і динаміку зміни значень показників якості обслуговування веб-сервісів [1].

Пропоновані виробниками програмних засобів реалізації перерахованих вище методів і видів резервування дозволяють враховувати окремі показники якості обслуговування веб-сервісів і вимоги користувача, що відносяться тільки до надійності, не зачіпаючи при цьому показники, які стосуються ефективності та супроводжуємості програмного забезпечення. Дослідження, проведені в області розробки проміжного програмного забезпечення, що враховує динаміку зміни показників якості обслуговування веб-сервісів, і опубліковані в технічній та науковій літературі результати експериментів, стосуються тільки окремих веб-сервісів і не розглядають рішень, що застосовуються до слабкозв'язаних систем масштабів підприємств.

Дослідження, проведені в області оптимізації стратегії відмовостійкості по одному з показників надійності не зачіпають питання, пов'язані з внутрішнім станом викликаємого веб-сервісу, збереження проміжних результатів роботи якого необхідно для його повторного виклику, тобто не враховують так звані непоправні витрати на його використання.

Сучасні інформаційні системи масштабу підприємства налічують сотні, а то й тисячі, програмних додатків (комерційних, власної розробки, успадкованих і т.д.), розроблених для різних програмних платформ (Java EE, Microsoft .NET, CORBA), що виконуються під управлінням різних операційних систем, таких як Windows, Mac OS, Solaris. При цьому корпоративні додатки не можуть існувати окремо один від одного. Для їх інтеграції на рівні даних можуть використовуватися файли або бази даних.

Для організації взаємодії на програмному рівні можуть використовуватися віддалені виклики процедур або системи обміну повідомленнями. З розвитком інтернет-орієнтованої діяльності підприємств задача інтеграції корпоративних додатків поширилася на середу Інтернет і розширилася до організації взаємодії корпоративних інформаційних систем підприємств за принципом слабкого зв'язку, що дозволяє їм продовжувати роботу, не чекаючи відповіді від викликаємої сторони [8]. Це зумовило прояв особливого інтересу розробників інтернет-орієнтованих інтеграційних рішень до сервіс-орієнтованого підходу, який давно і широко використовується в корпоративних інформаційних системах загального призначення, таких як Microsoft Dynamics AX, SAP R3, Baan ERP та інших. У його основу покладена сервіс-орієнтована архітектура, що припускає використання слабкозв'язаних замінних програмних компонентів, оснащених стандартними інтерфейсами для взаємодії по стандартним протоколам.

З поширенням сервіс-орієнтованого підходу на інтеграційні рішення інтернет-масштабу стало збільшуватися кількість функціонально подібних веб-сервісів корпоративних інформаційних систем підприємств, що надають споріднені послуги. У розвитку інтернет-орієнтованої діяльності підпри-

ємств і організації міжкорпоративних бізнес-процесів спостерігається тенденція динамічного перерозподілу бізнес-процесів між виконавцями, створюючими по суті єдине віртуальне підприємство, а також зміни складу самих виконавців, що ускладнює завдання організації взаємодії програм і програмних систем підприємств і підвищує її значення.

Висновки

У даній статті знаходить своє відображення методи та підходи до інтеграції програмно-апаратних засобів у розподілені системи масштабу підприємств.

Як один із способів інтеграції була розглянута побудова комп'ютерних систем на основі сервіс-орієнтованої архітектури. Був проведений аналіз надійності та математичне обґрунтування побудови таких систем.

Отримані результати показали, що за допомогою веб сервісів та сервіс-орієнтованої архітектури в цілому, можна підвищити ефективність та надійність процесів обробки даних у слабкопов'язаних системах масштабу підприємства.

Список літератури

1. Papazoglou M.P. Web Services: Principles and Technology / M.P. Papazoglou // Prentice Hall. – 2007. – Vol. 21. – P. 139-145.

2. Kharchenko V. Web Systems Availability Assessment Considering Attacks on Service Configuration Vulnerabilities / V. Kharchenko, Alaa Mohammed Ab-dul-Hadi, A. Boyarchuk, Y. Ponochovny / Seria "Advances in Intelligent Systems and Computing", Vol. 286, / W. Zamojski et al (edits), Springer International Publishing Switzerland, 2014. – P. 275-284.

3. Chatterjee S. Developing Enterprise Web Services: An Architect's Guide / S. Chatterjee / Prentice Hall PTR. – 2003.

4. Gustavo Alonso H. Web services: Concepts, Architectures and Application / H.K. Gustavo Alonso, Fabio Casati and V. Machiraju, / Springer, 2004.

5. Papazoglou M. Design methodology for web services and business processes / M. P. Papazoglou, J. Yang // London, UK: Springer-Verlag. – 2002. – P. 54-64.

6. Радченко Г.И. Распределенные вычислительные системы / Г.И. Радченко. – Челябинск: Фотохудожник, 2012. – 184 с.

7. Угрин Д. Дослідження сервісно-орієнтованої архітектури як інтеграційної платформи / Д.І. Угрин, В.В. Литвин / Вісник Національного університету "Львівська політехніка": збірник наукових праць. Тематичний випуск "Інформаційні системи та мережі". – Львів: Національний університет "Львівська політехніка", 2011. – № 699. – С. 260-270.

8. Плющенков Р. Способы решения интеграционных задач / Р.А. Плющенков. – Москва, 2006.

9. Elements of Service-Oriented Analysis and Design: an interdisciplinary modeling approach for SOA project [Електронний ресурс] / Режим доступу до ресурсу: http://www-128.ibm.com/developerworks/library/ws-soad1.

10. Business process execution language for web services. [Електронний ресурс] / Режим доступу до ресурсу: http://www-106.ibm.com/developerworks/library/ws-bpel.

11. IBM Redbook Patterns: Service-Oriented Architecture and Web Services. [Електронний ресурс] / Режим доступу до ресурсу: http://redbooks.ibm.com/abstracts/sg246303.html.

12. Сервіс-орієнтована архітектура [Електронний ресурс] / Режим доступу до ресурсу: http://www.citforum.idknet.com/internet/webservice/soa/.

13. Подход к композиции сервис-ориентированных прикладных приложений [Електронний ресурс] / Режим доступу до ресурсу: http://studbooks.net/47748/informatika/ podhod_kompozitsii_servis-orientirovannyh_prilozheniy.

Надійшла до редколегії 3.02.2016

Рецензент: д-р техн. наук, проф. Г.А. Кучук, Харківський університет Повітряних Сил ім. І. Кожедуба, Харків.

МЕТОДЫ ОРГАНИЗАЦИИ ВЗАИМОДЕЙСТВИЯ РАСПРЕДЕЛЕННЫХ КОМПЬЮТЕРНЫХ СИСТЕМ НА ОСНОВЕ СЕРВИС-ОРИЕНТИРОВАННОЙ АРХИТЕКТУРЫ

Ю.Д. Свистунов

В статье обсуждаются методы и подходы к интеграции программно-аппаратных средств в распределенную систему предприятия. Определены понятия сервис-ориентированной архитектуры и ее основных компонентов. Математически обоснованно построение систем на основе сервис-ориентированной архитектуры и проанализирована надежность таких систем. Полученные результаты позволяют утверждать, что с помощью веб-сервисов можно повысить эффективность и надежность процессов обработки данных в слабосвязанных системах масштаба предприятия.

Ключевые слова: *сервис-ориентированная архитектура, распределенные системы, Web-сервис.*

THE METHODS OF INTERACTION ORGANIZATION BETWEEN DISTRIBUTED COMPUTER SYSTEMS BASED ON SERVICE-ORIENTED ARCHITECTURE

Y.D. Svystunov

The paper describes methods and approaches to the software and hardware integration into enterprise distributed system. It defines the concepts of service-oriented architecture and its main components. In this paper the development of the systems based on service-oriented architecture was mathematically proved, and reliability of such systems was analyzed. The obtained results allow to state that using Web services can increase efficiency and reliability of data process in weekly-referenced distributed systems across the enterprise.

Keywords: *Service-oriented architecture, distributed systems, Web-service.*