XXIV Seminarium

ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE 2014

Oddział Gdański PTETiS

WYBRANE METODY EFEKTYWNEJ INTEGRACJI KOMPONENTÓW W SYSTEMACH ROZPROSZONYCH

Paweł KACZMAREK

Politechnika Gdańska, Wydział Elektroniki, Telekomunikacji i Informatyki
tel: 58 347 24 89 fax: 58 348 61 25 e-mail: pkacz@eti.pg.gda.pl

Streszczenie: W pracy przedstawiono problemy wytwarzania efektywnych aplikacji rozproszonych ze szczególnym uwzględnieniem wytwarzania zorientowanego na integrację komponentów. Opisano metody komunikacji stosowane w aplikacjach rozproszonych oraz architektury oprogramowania, takie jak: SOA, ESB i SCA. Metody komunikacji obejmują standardy usług sieciowych XML Web services oraz RESTful Web services. Ponadto opisano wykorzystanie interfejsu programistycznego API jako nadbudowy nad usługami sieciowymi, dzięki czemu możliwe jest korzystanie z usług sieciowych realizowanych w dowolnej technologii. Wskazano na rosnąca popularność RESTful Web services wynikającą między innymi z lepszych osiągów wydajnościowych i elastycznej konfiguracji.

Słowa kluczowe: Architektura zorientowana na usługi, XML Web services, RESTful Web services.

1. WSTĘP

Integracja gotowych komponentów pozwala na usprawnienie procesu wytwarzania przez ponowne użycie już istniejących modułów oraz wybór tych komponentów, które najlepiej odpowiadają potrzebom wytwarzanej aplikacji. Obecne integracyjne wytwarzanie aplikacji rozproszonych jest realizowane przy użyciu wielu różnych architektur określanych ogólnie jako architektura zorientowana na usługi (Service Oriented Architecture – SOA) [1]. Ogólne podejście SOA jest realizowane przez wiele specyficznych rozwiązań takich jak: kompozycja usług złożonych z usług prostych, Enterprise Service Bus (ESB) [2], Service Component Architecture (SCA) [3] i inne.

Celem opracowanych architektur jest uzyskanie efektywności zarówno w procesie wytwarzania jak i działania aplikacji, choć różnorodność rozwiązań może świadczyć o trudnościach w osiągnięciu tego celu. Każda z architektur stosuje nieco inne podejście do wytwarzania i używa nieco innego nazewnictwa integrowanych modułów.

Usługi XML Web services zostały opracowane jako standard komunikacji w celu zapewnienia integracji usług [4]. Zaproponowano wiele rozszerzeń dotyczących zagadnień bezpieczeństwa, transakcyjności, niezawodności i innych. XML Web services (XML WS) okazały się jednak stosunkowo mało wydajne i trudne do elastycznej konfiguracji [5]. RESTful Web services (RS WS) opierają swoje działanie na mniej restrykcyjnych zasadach formatowania przesyłanych danych [6]. Niezależnie od stosowanego rozwiązania, dostawca usługi i klient muszą być zgodni co do znaczenia przesyłanych danych.

W pracy przeanalizowano również wykorzystanie API jako metody komunikacji, co pozwala na ukrycie szczegółów komunikacyjnych przed programistą, jednak wymaga zastosowania dodatkowych bibliotek. Metoda ta jest stosowana głównie przez dużych dostawców systemów i usług informatycznych [7] [8].

W pracy przedstawiono analizę i porównanie stosowanych rozwiązań oraz wskazano na główne czynniki wpływające na wybór rozwiązań w procesie wytwarzania aplikacji. Wyniki porównania mogą być przydatne dla konsumentów usług i mogą usprawnić proces wytwarzania oprogramowania. Ponadto opisano ogólny proces wyboru metod integracji, który może być stosowany w przypadku różnych dostawców usług.

2. ARCHITEKTURY SYSTEMÓW ROZPROSZONYCH

Rozwój systemów rozproszonych wiązał się z opracowaniem wielu różnych rozwiązań architektonicznych obejmujących zarówno architektury sprzętowe jak i programowe. Rozwój architektur programowych miał na celu dostosowanie oprogramowania do bieżących wymagań i możliwości technologicznych. Dynamiczny rozwój internetu spowodował opracowanie metod wytwarzania bazujących na architekturze zorientowanej na usługi [1].

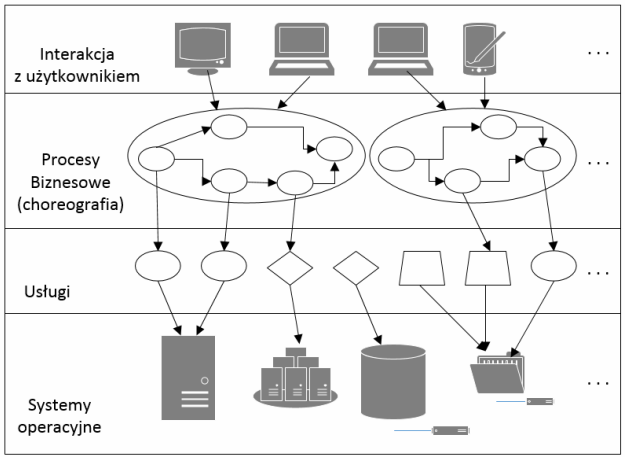
2.1. Kompozycja usług złożonych w SOA

W tym podejściu zakłada się, że złożona aplikacja jest realizowana przez proste usługi, które są ze sobą integrowane przy użyciu standardowego mechanizmu komunikacji [4]. Usługi udostępniają swoją funkcjonalność za pośrednictwem interfejsu dostępnego dla potencjalnych klientów.

W przypadku kompozycji usług złożonych twórca aplikacji wykorzystuje standardy wymiany danych i samodzielnie decyduje, w jaki sposób komponenty będą integrowane.

Typowo wykorzystywane są standardy usług sieciowych XML Web services lub RESTful Web services na niższych poziomach [4] [6]. Usługi działają w ramach serwerów aplikacji, które udostępniają biblioteki wspierające przetwa-

rzanie komunikacji. Wyższe warstwy architektury wykorzystują uzupełniające się podejścia choreografii i orkiestracji usług. W choreografii logika interakcji między usługami jest definiowana z globalnej perspektywy systemu. W orkiestracji zaś (np. w procesach biznesowych zdefiniowanych z wykorzystaniem języka BPEL) logika działania jest zdefiniowana z perspektywy lokalnej jednego z uczestników systemu. Ma rysunku 1 przedstawiono główne warstwy abstrakcji w kompozycji aplikacji.



Rys. 1. Główne warstwy abstrakcji w kompozycji aplikacji w architekturze SOA

Wśród stosowanych modeli komunikacji możemy wyróżnić między innymi [9]: zdalne wywołanie metod, przesyłanie wiadomości, model zasobowy i inne. Usługi XML Web services stosują zdalne wywołanie procedur, natomiast usługi RESTful Web services skupiają się na modelu dostępu do zasobów.

2.2. Enterprice Service Bus

Architektura integracji komponentów ESB [2], zwana też szyną usług, wykorzystuje wspólną szynę komunikacyjną łączącą wielu komunikujących się nadawców i odbiorców. Architektura ma zastosowanie w przypadku, gdy istnieje wiele modułów systemu oraz potencjalnie występuje konieczność komunikacji każdy z każdym.

Rozwiązanie jest stosowane w wielu szczegółowych wariantach [10]. Message Oriented Middleware bazuje na systemach kolejkowych oraz rozsyłaniu wiadomości na zasadzie publikuj−subskrybuj. Pośrednicy (ang. brokers) wiadomości bazują na konwertowaniu wiadomości pomiędzy komunikującymi się modułami. Ma to zastosowanie w przypadku integracji modułów, których interfejsy nie są zgodne. Kanały dołączane (ang. plug-in channels) wspierają heterogeniczną komunikację oraz dostarczają możliwość rozszerzania dostępnych modułów.

Zastosowanie ESB wiąże się z wykorzystaniem dedykowanych serwerów warstwy pośredniczącej, które odpowiadają za przesyłanie i konwersję wiadomości.

2.3. Service Component Architecture

Architektura SCA [3] jest jednym ze sposobów budowy systemu o modułowej strukturze. Rozwiązanie zostało zaczerpnięte z koncepcji wytwarzania urządzeń elektronicznych. W rozwiązaniu tym, każdy moduł może składać się z mniejszych modułów, które realizują potrzebną funkcjonalność. Wspiera to ponowne użycie kodu i tym samym usprawnia proces wytwarzania oprogramowania.

Główne koncepcje wykorzystywane w SCA to: Kompozyt (jednostka, która jest wdrażana w środowisku SCA), Usługa (punkt dostępu do jednostki), Komponent (zapewnia logikę działania jednostki) oraz Odwołanie (dostęp do innego serwisu). Wdrożenie i uruchomienie aplikacji SCA jest wspierane przez dedykowane serwery, które udostępniają warstwę pośredniczącą zgodną z zaproponowanym standardem.

3. STANDARDY XML WEB SERVICES I RESTFUL WEB SERVICES

Integracja usług wymaga zastosowania odpowiednich metod komunikacji między poszczególnymi modułami systemu. Metody komunikacji muszą zapewniać skuteczną integrację modułów, a jednocześnie ograniczać konieczny nakład pracy podczas implementacji i konfiguracji systemu.

3.1. Standardy XML Web services

Standardy XML WS [4], zwane też SOAP−based Web services, zostały zaprojektowane w celu integracji heterogenicznych usług pochodzących od różnych dostawców. Centralnym elementem rozwiązania jest grupa trzech standardów: SOAP, WSDL oraz UDDI. Standard SOAP definiuje format wymiany informacji między klientem i dostawcą usługi, co obejmuje wykorzystanie standardu XML oraz zdefiniowanie odpowiednich znaczników określających nazewnictwo oraz wartości parametrów. Standard WSDL (Web Dervices Description Language) definiuje, w jaki sposób usługi mają być opisywane w kontekście nazewnictwa, typów parametrów i wyników. Standard UDDI (Universal Description and Discovery Interface) wspiera udostępnianie opisów usług i umożliwia ich wyszukiwanie.

Równolegle z wykorzystaniem XML WS są opracowywane kolejne standardy, które rozszerzają podstawowy zbiór protokołów SOAP/WSDL. Rozszerzone standardy obejmują takie zakresy funkcjonalne jak: transakcje, bezpieczeństwo, niezawodność i inne. Opracowywane rozwiązania zwiększają funkcjonalność usług, ale jednocześnie wiążą się z trudnościami w uzyskaniu zgodności między różnymi dostawcami [5], wymagają złożonej konfiguracji oraz negatywnie wpływają na wydajność. Spowodowało to, że usługi wykorzystujące XML WS są stopniowo zastępowane usługami wykorzystującymi RESTful WS[6].

3.2. Standardy RESTFul Web services

REST (Representational state transfer) jest ogólnym stylem architektury, w którym głównym punktem odniesienia są zasoby udostępniane przez serwer za pośrednictwem zunifikowanego interfejsu. Usługi RESTful Web services realizują założenia REST w kontekście protokołu HTTP. Dostęp do zasobów bazuje na standardowych operacjach HTTP: GET (pobranie), PUT (aktualizacja), DELETE (usunięcie), POST (utworzenie nowego).

Zasoby są identyfikowane przy użyciu jednoznacznego identyfikatora URI. Transfer informacji związanej z zasobami i operacjami na nich może odbywać się przy użyciu różnych standardów takich jak XML, JSON, tekst i inne. Wykorzystanie zunifikowanego interfejsu ułatwia podział systemu na moduły i przekazywanie reprezentacji zasobów między modułami.

RS WS w praktyce zastąpiły XML WS ze względu na dużo większą prostotę konfiguracji [6]. Wykorzystanie RS WS

przez programistów w aplikacjach możliwe jest zarówno przez samodzielną implementację jak i przez wykorzystanie gotowych bibliotek. Samodzielna implementacja serwera lub klienta usług jest relatywnie mało kosztowna ze względu na proste zasady przesyłania danych.

RS WS charakteryzują się większą prostotą konfiguracji w porównaniu do standardów XML WS, gdyż nie wymagają instalowania i konfigurowania złożonych bibliotek zajmujących się konstruowaniem wiadomości SOAP. Uproszczenie konfiguracji odbywa się kosztem przerzucenia na programistę konieczności sprawdzenia i prawidłowej obsługi zawartości przesyłanych informacji.

Zarówno dla usług XML WS jak i RESTful WS pojawiają się trudności ze sprawdzaniem poprawności w czasie kompilacji. Biorąc pod uwagę te trudności, dostęp do zasobów jest realizowany również za pośrednictwem bibliotek API, które stanowią wyższą warstwę względem istniejących interfejsów usług sieciowych jak opisano w kolejnym rozdziale.

4. INTEGRACJA Z WYKORZYSTANIEM API

Integracja z wykorzystaniem API jest kolejnym sposobem włączania istniejącego kodu do funkcji wytwarzanego systemu. Ten rodzaj integracji zyskuje stopniowo coraz większą popularność względem wywołań usług sieciowych Web services. Dostarczenie własnego API do oferowanych usług wiąże się zazwyczaj z dużym nakładem pracy związanym z wykonaniem oprogramowania i jego utrzymaniem.

Przykładem integracji za pośrednictwem API może być zestaw bibliotek udostępniany przez portal Google Maps [7], który udostępnia dedykowane biblioteki dla popularnych platform uruchomienia: iOS, Android, web (JavaScript, embed HTML). Ponadto dostępny jest surowy interfejs Web services. Biblioteki dla różnych platform uruchomienia posiadają zbliżoną funkcjonalność.

Serwis dostarcza wiele dedykowanych funkcji i alternatywnych opcji konfiguracyjnych, co powoduje, że jego wykorzystanie wymaga zapoznania i wyboru odpowiedniego rozwiązania. Zakres funkcjonalny obejmuje między innymi: Maps APIs, Maps Image APIs, Places API, Google Earth API.

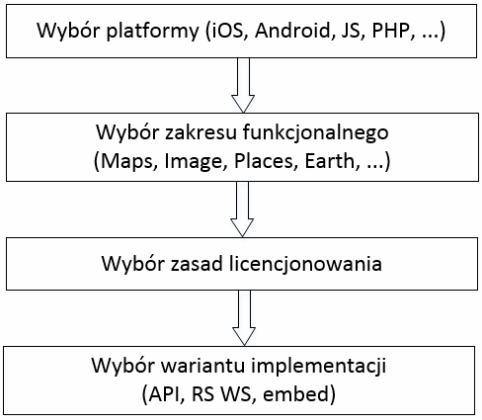
Duża liczba alternatywnych rozwiązań spowodowała, że firma udostępniła dedykowany serwis mający na celu wspomaganie użytkownika w samym procesie wyboru odpowiedniego rozwiązania. Przykładowo, wykorzystanie typowego rozwiązania Google Map na stronie internetowej może być zrealizowane na dwa alternatywne sposoby: wykorzystanie bibliotek API JavaScript lub rozwiązania Embed API. W przypadku Embed API wykorzystywany jest znacznik HTML iframe, aby do bieżącej strony wstawić treść strony przesłanej z Google. Ponadto firma określa szczegółowe zasady licencjonowania swoich produktów, które obejmują między innymi: maksymalną liczbę wywołań w ciągu dnia, maksymalną liczbę zdefiniowanych punktów na mapie, wykorzystanie w systemach wewnętrznych i inne.

Niezależnie od dostępnych API portal udostępnia interfejs bazujący na RESTful WS. Adresy URI wskazują zasoby, zaś operacje HTTP i ich parametry są wykorzystywane do przesłania danych do wywołanych usług. Usługi zwracają wynik w formacie XML lub JSON zależnie od konfiguracji w parametrach. Określenie typu wyniku w parametrze odbiega od stosowanego często mechanizmu Content negotiation, który umożliwia uzgodnienie typów danych przesyłanych między klientem i serwerem.

Użytkownik usługi jest odpowiedzialny za skonstruowanie odpowiedniego łańcucha wywołania URL oraz za prawidłowe przetworzenie wyniku. Jak stwierdza dokumentacja usług: dokładny format poszczególnych wyników usług sieciowych nie jest zagwarantowany i nie należy zakładać, że format wyniku dla żądania będzie taki sam dla różnych wartości zapytań.

Takie rozwiązanie wymaga zastosowania bibliotek wspierających przetwarzania i wyszukiwanie informacji w formatach XML i JSON. Przykładowo dla formatu XML wykorzystywane są biblioteki implementujące standard XPath, zaś format JSON jest dobrze obsługiwany w języku JavaScript, który posiada wbudowane funkcje wydobywania danych.

Wybór optymalnego rozwiązania obejmuje kilka etapów jak przedstawiono na rysunku 2:



Rys. 2. Kroki decyzyjne podczas integracji usług dostępnych w alternatywnych wariantach konfiguracji na przykładzie Google Maps API / RS WS

Innym, analogicznym przykładem dostępu za pośrednictwem API jest portal Facebook [8]. Portal udostępnia dedykowane API dla platform uruchomienia, które są potencjalnym celem biznesowym: iOS, Android, web (JavaScript, PHP), Unity. Dostępnych jest też wiele bibliotek wykonywanych przez zewnętrzne podmioty m. in. dla języków C#, Java i inne. Podobnie jak w poprzednim przypadku, biblioteki dla różnych platform posiadają zbliżoną, choć nie identyczną funkcjonalność.

Dostępne funkcje API tworzą rozbudowane biblioteki, w których dostępnych jest wiele alternatywnych rozwiązań. Przykładowo implementacja współdzielenia (Share) w API JavaScript może zostać realizowana na kilka sposobów:

- Social Plugins – Like / Share Buttons – wymagają skopiowania fragmentu HTML, nie wymagają obsługi logowania Facebooka
- Share Dialog – wywoływany przez Facebook SDK lub przez przekierowanie URL, UI spójne z FB, nie wymagają obsługi logowania do Facebooka
- Graph API – bardziej elastyczne, umożliwiają publikowanie własnych historii wydarzeń, pozwalają na stworzenie własnego UI, ale wymagają implementacji logowania do Facebooka

We wcześniejszych latach portal Facebook udostępniał możliwość wywoływania zadań za pośrednictwem usług RESTful Web Services. Obecnie większy nacisk jest kładziony

na wykorzystanie API zaś usługi RS WS stanowią wewnętrzny mechanizm komunikacji

Analogicznie jak w przypadku poprzedniej platformy, wybór alternatywnych opcji implementacji obejmuje: wybór platformy, wybór zakresu funkcjonalnego, wybór zasad licencjonowania, wybór metod implementacji.

5. PORÓWNANIE ALTERNATYWNYCH ROZWIĄZAŃ

Przedstawione metody (XML WS, RS WS, API) mają na celu realizację tych samych zadań związanych z integracją istniejących usług w nowo wytwarzanych aplikacjach. W tabeli 1 przedstawiono główne cechy metod w kontekście procesu wytwarzania aplikacji oraz potencjalnie ich dalszego utrzymania.

Tablica 1. Porównanie rozwiązań integracji modułów

	XML WS	RS WS	API
przenośność	wysoka	wysoka	niska
standard komunikacji	tak	częściowo	częściowo
łatwość konfiguracji	niska	wysoka	wysoka
łatwość programowania	średnia	średnia	wysoka
wydajność	niska	wysoka	wysoka
zgodność formatu	częściowo	po stronie programisty	tak

XML WS w swoich założeniach przesyłają dane, które są zgodne ze standardem SOAP. Umożliwiają również wybór wersji standardu komunikacyjnego lub jednego z jego rozszerzeń. Wykorzystanie SOAP odbywa się kosztem wysokich nakładów pracy związanych z konfiguracją środowisk uruchomienia. Pomimo trudności z konfiguracją, łatwość programowania XML WS jest zwiększona dzięki wsparciu środowisk programistycznych. Integracja API wykazuje najmniejszą przenośność i zgodność ze standardami, jednak wykazuje wysoką prostotę w konfiguracji i programowaniu. Jest to jednak warstwa wyższa względem komunikacji XMLWS lub RESTfull WS.

Tendencje technologiczne wskazują na fakt, że standardy XML WS okazały się zbyt trudne w wykorzystaniu pomimo formalnie wysokiej zgodności przesyłanych danych pomiędzy klientem i serwerem. Analiza popularnych serwisów, takich jak Facebook i Google Maps, potwierdza tę tendencję rozwojową, gdyż nie udostępniają one praktycznie interfejsu XML WS.

6. PODSUMOWANIE

Ponowne użycie oprogramowania stanowi od wielu lat ważny element optymalizacji procesu wytwarzania nowych aplikacji. Wraz ze zmianami technologii zmieniają się też metody i techniki ponownego użycia oprogramowania.

Informacje i porównania przedstawione w pracy mogą być wsparciem w procesie wytwarzania oprogramowania przez usystematyzowanie procesu wyboru integrowanych modułów oraz określenie głównych cech stosowanych technologii. W przypadku systemów przeznaczonych na ściśle określone platform wykonania wskazane będzie użycie metod mniej przenośnych, takich jak interfejs API. W przypadku, gdy wysoka przenośność jest wymagana, wskazane jest użycie bardziej uniwersalnych rozwiązań jak RS WS.

7. BIBLIOGRAFIA

1. Singh M.P., Huhns M.N.: Service–Oriented Computing, Semantics, Processes, Agents, John Wiley & Sons, 2005, ISBN: 0-470-09148-7
2. Shepherd P.: Oracle SCA – The Power of the Composite, Oracle, 2009
3. Booz D., Karmarkar A.: Service Component Architecture Spring Component Implementation Specification, OASIS Open, 2011
4. Sprott D., Wilkes L.: Understanding Service–Oriented Architecture, Microsoft Corp., 2004
5. Egyedi T. M.: Standard–compliant, but incompatible?!, Computer Standards & Interfaces, ISSN 0920-5489, 2007
6. Rodriguez A.: RESTful Web services: The basics, IBM developerWorks, 2008
7. Google Map Product Documentation https://developers.google.com/maps/
8. Facebook Product Documentation http://developers.facebook.com/docs
9. Tanenbaum A. S., van Steen M.: Distributed Systems Principles and Paradigms, ISBN: 0132392275, Prentice Hall, 2002
10. Kress J., Maier B., Normann H., et.al., Enterprise Service Bus, Oracle, 2013

SELECTED METHODS OF EFFECTIVE COMPONENT INTEGRATION IN DISTRIBUTED SYSTEMS

Key-words: Service Oriented Architecture, XML Web services, Restful Web services

The paper presents integration oriented development methods of effective distributed applications. Existing software architectures were described including: SOA, ESB and SCA. Methods of distributed communication and integration were analyzed and compared. Described methods cover XML Web services and RESTful Web services. Additionally, the approach based on Application Programming Interface (API) was described as a method of integration that allows to use different low level communication mechanisms. It was noticed that RESTful Web services become more popular in use compared to XML Web services because of better performance and easier configuration