Helion

Zbuduj własne urządzenia pomiarowe z platformą Arduino!

Monitorowanie otoczenia z Arduino

Emily Gertz, Patrick Di Justo

MAKERMEDIA

Tytuł oryginału: Service-Oriented Architecture : Concepts, Technology, and Design

Tłumaczenie: Andrzej Grażyński

ISBN: 978-83-246-7493-0

Authorized translation from the English language edition, entitled: SERVICE-ORIENTED ARCHITECTURE (SOA): CONCEPTS, TECHNOLOGY, AND DESIGN; ISBN 0131858580; published by Pearson Education, Inc, publishing as Prentice Hall.

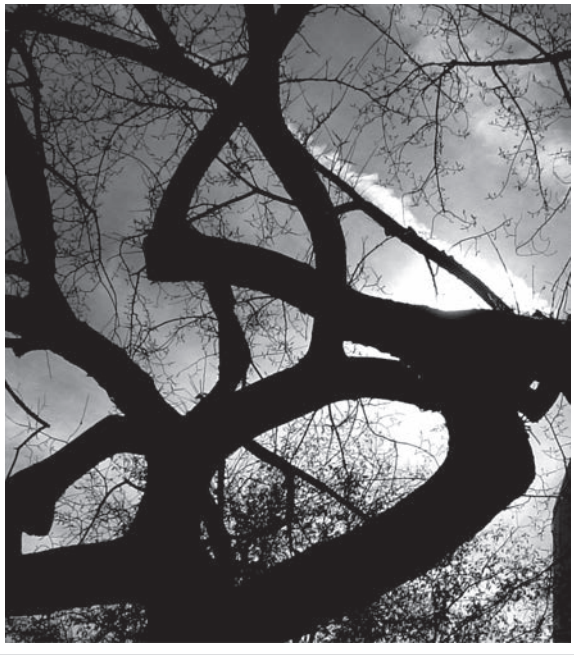
Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: http://helion.pl (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/soakon
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/soakon.zip

Printed in Poland.

Spis treści



Rozdział 3

Wprowadzenie do SOA

Zanim zajmiemy się szczegółami rozwiązań zorientowanych na usługi i praktycznym tworzeniem tych rozwiązań, zdefiniujmy wpierw kilka podstawowych koncepcji i przedstawmy podstawowe problemy związane z nimi.

3.1. Fundamenty SOA

Termin „orientacja na usługi" funkcjonuje już od dłuższego czasu i to w różnych kontekstach, i dla różnych celów. Niezmiennym elementem każdego z tych znaczeń jest jednak aspekt dekompozycyjny — czyli osobne traktowanie poszczególnych podproblemów rozwiązywanego problemu. Oznacza to, że logika rozwiązywania skomplikowanego problemu może być jaśniej skonstruowana, mniej złożona i lepiej kontrolowana, gdy problem ten rozpatrywany będzie jako kolekcja prostszych, powiązanych części; każda z tych części odpowiadać powinna dobrze określonemu aspektowi oryginalnego problemu.

Ta generyczna reguła dekompozycji przenika wszem i wobec każde praktyczne podejście do rozwiązywania dowolnych problemów, także rozwiązań automatyzacji biznesu i przeznaczonych do tego technologii — SOA nie jest w tym względzie wyjątkiem. Tym jednak, co dla SOA charakterystyczne, jest sposób, w jaki wykonywana jest separacja podproblemów.

3.1.1. Orientacja na usługi — analogia

Wyobraźmy sobie typowe, tętniące życiem duże miasto. Jego mieszkańcy nieustannie korzystają — mniej lub bardziej świadomie — z bogatego zestawu usług. Ten globalnie postrzegany „sektor usługowy" można potraktować jak pojedynczy „biznes" zarabiający na udostępnianiu określonych dóbr ogółowi swych konsumentów; zdecydowanie bardziej realne będzie jednak potraktowanie odrębnie każdego z usługodawców, dbającego o jak najlepszą kondycję własnego przedsięwzięcia. Kompromisem godzącym oba te podejścia jest postrzeganie naszej metropolii jako swoistej architektury wiążącej konsumentów z usługodawcami działającymi w rozproszeniu.

„Architektura" to określenie wywołujące nieodmiennie skojarzenia technologiczne. Faktycznie, „architektura zorientowana usługowo" to model, w ramach którego logika automatyzacji podzielona („dekomponowana") jest na odrębne jednostki logiczne. Owe jednostki, z założenia rozseparowane, podporządkowane są jednak swemu pierwowzorowi, jakim jest oryginalny problem.

Wspominaliśmy już o tym, że dekompozycja złożonego problemu na podproblemy jest regułą generalną i jej przejaw na gruncie „orientacji na usługi" ma charakter wyjątkowy. *W jakim sensie wyjątkowy?* Lwia część tej książki poświęcona jest sformułowaniu odpowiedzi na to właśnie pytanie, więc na początek przyjrzyjmy się w zarysie najważniejszym cechom owej wyjątkowości.

Jeśli w środowisku rozproszonego biznesu narzucimy na uczestników tego biznesu krępujące zależności, możemy w znaczącym stopniu ograniczyć możliwości wykorzystania potencjału każdego uczestnika. A zatem, mimo iż jak najbardziej pożądana jest możliwość wzajemnego komunikowania się usługodawców ze sobą i wzajemnego wymieniana między sobą rozmaitych dóbr, to nie mogą one odbywać się za cenę zbyt silnego uzależnienia od siebie uczestników biznesu. Konieczne jest pozostawienie każdemu swobody rozwoju i wzrostu, (względnie) niezależnie od innych.

Mimo postulowanej niezależności poszczególnych „biznesów", każdy z nich musi jednak stosować się do pewnych uzgodnionych konwencji — na przykład sprzedawania i kupowania dóbr oraz usług za tę samą walutę, przestrzegania wymaganych standardów technicznych i społecznych, czy nawet wymogu, by pracownicy kontaktowali się z klientami w ich ojczystym języku.

Stwarza to — oczywiście — pewne ograniczenia, jednakże pomyślane głównie dla wygody konsumentów i niestwarzające znaczących barier na drodze samorozwoju każdego z usługodawców.

Zgodnie z powyższą analogią, architektura zorientowana na usługi (SOA) eksponuje istnienie każdej z usług niezależnie od pozostałych, lecz nie w zupełnej izolacji od nich. Poszczególne jednostki logiczne wciąż muszą pozostawać w zgodności z pewnymi regułami, wprowadzającymi określony stopień zgodności i standaryzacji, bez jednakże krępowania niezależnego rozwoju. Na gruncie SOA wspomniane jednostki logiczne nazywane są **usługami** (*services*).

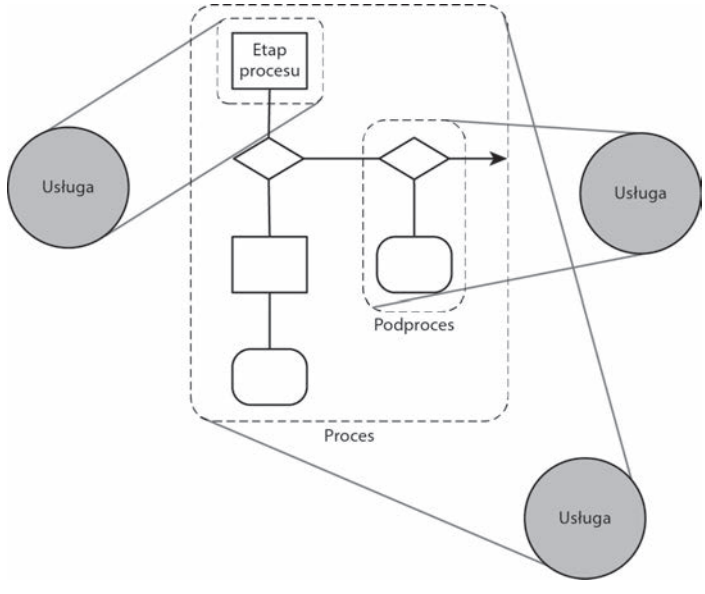
3.1.2. Jak usługi enkapsulują logikę?

Poszczególne usługi, aby zachować własną niezależność, muszą enkapsulować swą logikę w odrębnym kontekście. Kontekst ten może być specyficzny dla danego zadania biznesowego, podmiotu gospodarczego czy też innej, swoistej reguły grupowania.

Zróżnicowany może być także zakres logiki realizowanej przez usługę, zależnie od specyfiki problemu, którego rozwiązaniu usługa ta jest dedykowana. Co więcej, ponieważ dana usługa może być „dostarczycielem logiki" dla innych usług, można połączyć kilka usług w kolekcję, postrzeganą jako pojedynczą, złożoną usługę.

I tak na przykład konkretne rozwiązanie z dziedziny automatyzacji biznesu dedykowane jest zazwyczaj konkretnemu problemowi biznesowemu; logika tego procesu przekłada się na określoną sekwencję kroków realizowanych przez wspomniane rozwiązanie — sekwencję wynikającą z reguł biznesu i właściwości środowiska wykonawczego.

Jak pokazano na rysunku 3.1, budując rozwiązanie składające się z usług, powierzamy każdej usłudze zadanie wynikające z konkretnego kroku (etapu) lub ciągu kroków składających się na podproces; pojedyncza usługa może być także dedykowana kompletnemu procesowi. W dwóch ostatnich przypadkach logika reprezentowana przez złożoną usługę może obejmować logikę „importowaną" z innych usług.

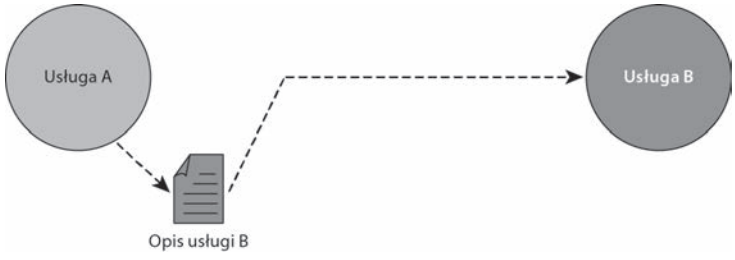


Rysunek 3.1. Usługi mogą reprezentować zróżnicowany zakres logiki

Aby można było wykorzystywać logikę enkapsulowaną przez usługę, usługa ta musi partycypować w realizacji aktywności biznesowych. Żeby było to możliwe, konieczne jest ustanowienie wyraźnych relacji z jej użytkownikami.

3.1.3. Jak powiązane są usługi?

W ramach SOA usługi mogą być wykorzystywane przez inne usługi lub programy. Muszą być w tym celu świadome nawzajem swego istnienia; świadomość ta zapewniana jest poprzez **opisy usług** (*service descriptions*).



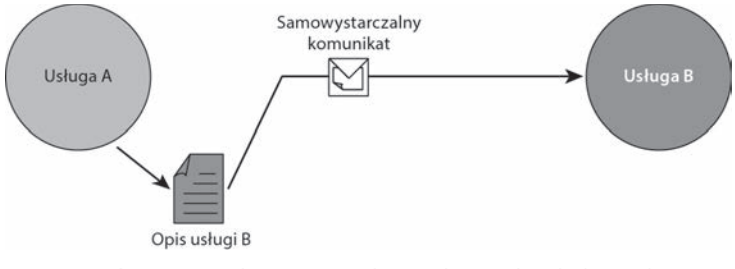
Rysunek 3.2. Usługa A może komunikować się z usługą B, gdyż posiada niezbędną do tego informację w postaci opisu usługi B

Opis usługi, w najprostszej postaci, zawiera nazwę usługi oraz specyfikację danych oczekiwanych i zwracanych przez tę usługę. Powiązania między usługami, realizowane za pośrednictwem opisów, nazywane są **luźnymi powiązaniami** (*loose couplings*). W przykładzie pokazanym na rysunku 3.2 usługa A jest świadoma istnienia usługi B, ponieważ znajduje się w posiadaniu jej opisu.

Świadomość istnienia usługi to dopiero początek. Aby usługi mogły współdziałać i generalnie przejawiać jakąś użyteczność, muszą wymieniać między sobą informacje. Konieczny jest do tego framework komunikacyjny, pozwalający na zachowanie luźnego powiązania; takim frameworkiem jest **messaging** („komunikowanie").

3.1.4. Jak komunikują się usługi?

Gdy usługa wysyła komunikat, traci kontrolę nad tym, co dzieje się z tym komunikatem. Z tego właśnie względu komunikaty nazywane są „niezależnymi jednostkami komunikacji"; oznacza to, że komunikaty, podobnie jak usługi, są bytami autonomicznymi. Każdy komunikat musi więc być wyposażony w zasób inteligencji wystarczający do realizacji powierzonej mu roli w logice przetwarzania (patrz rysunek 3.3).

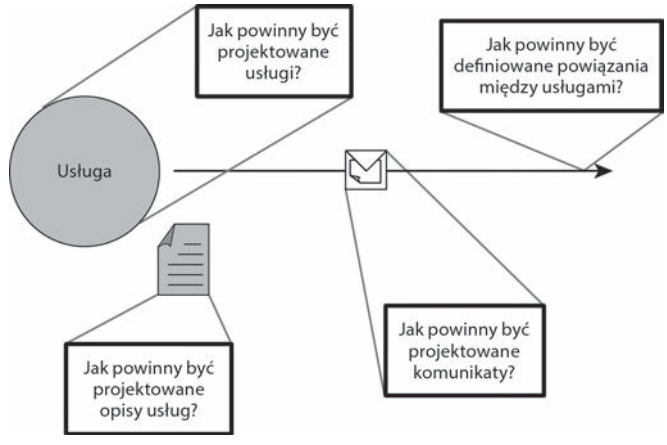


Rysunek 3.3. Komunikat istniejący jako niezależna jednostka komunikacji

Usługi wyposażone w swe opisy i zdolne wymieniać informacje za pomocą komunikatów tworzą podstawową architekturę. Na pierwszy rzut oka wydaje się ona podobna do starszych architektur przetwarzania rozproszonego, opartych na wymianie komunikatów oraz odseparowaniu interfejsu od logiki przetwarzania. Czynnikiem, który wyróżnia ją w tym kontekście, jest sposób powiązania jej trzech podstawowych komponentów — usług, opisów i komunikatów: właśnie to powiązanie jest charakterystyczne dla **orientacji na usługi** (*service-orientation*).

3.1.5. Jak projektuje się usługi?

Podobnie jak „orientacja obiektowa", tak i „orientacja na usługi" wykształciła się jako dobrze zdefiniowane podejście projektowe, oparte na powszechnie przyjętych zasadach ustanawiających zarówno miejsce, jak i projekt komponentów architektonicznych (patrz rysunek 3.4).



Rysunek 3.4. Zasady orientacji na usługi odzwierciedlają problemy projektowe

Organizując logikę przetwarzania wedle reguł orientacji na usługi, otrzymujemy zestandaryzowaną **logikę przetwarzania zorientowaną na usługi**; gdy rozwiązanie biznesowe oparte zostaje na jednostkach tej logiki, otrzymujemy coś, co nazywać będziemy **rozwiązaniem zorientowanym na usługi**.

Wspomniane reguły zorientowania na usługi opisane zostaną wyczerpująco w następnych rozdziałach tej książki; w tym miejscu, niejako tytułem wstępu, wymienimy ich najważniejsze aspekty:

- *luźne powiązanie* — usługi utrzymują między sobą powiązania na zasadzie wzajemnej świadomości swego istnienia, co sprowadza do minimum zależności między nimi;
- *kontraktowość* — usługi stosują się do uzgodnień w kwestii komunikacji, zdefiniowanych w opisie (opisach) i odnośnych dokumentach;
- *autonomia* — każda usługa sprawuje pełną i wyłączną kontrolę nad logiką, którą enkapsuluje;
- *abstrakcyjność* — usługa skrywa przed światem zewnętrznym wszystkie elementy swej logiki, które nie stanowią części kontraktu;

- *wielokrotnego użycia* — podział logiki biznesowej między poszczególne usługi dokonywany jest z intencją jak największej uniwersalności tych usług w znaczeniu ich wielokrotnego wykorzystywania;
- *komponowalność* — działanie usług może być koordynowane w stopniu umożliwiającym łączenie ich w kolekcje, będące de facto usługami złożonymi;
- *bezstanowość* — usługi sprowadzają do minimum informację specyficzną dla swego funkcjonowania;
- *wykrywalność* — usługi projektowane są jako wystarczająco intuicyjne, by za pomocą dostępnych mechanizmów ich wykrywania można je było odnajdywać i oceniać ich przydatność do określonego celu.

Znając komponenty składające się na naszą podstawową architekturę oraz reguły ich projektowania, do kształtowania i standaryzowania brakuje już tylko jednego — platformy implementacyjnej, która pozwoliłaby te zamierzenia urzeczywistnić, czyli nadać im materialny kształt rozwiązania zorientowanego na usługi. Taką platformę oferuje technologia **usług sieciowych**.

3.1.6. Jak zbudowane są usługi?

Jak już wspominaliśmy, pojęcie „zorientowania na usługi" i rozmaite abstrakcyjne modele SOA istniały już przed pojawieniem się usług sieciowych. Żadna jednak z dostępnych technologii nie okazała się tak przydatna i tak udana w manifestowaniu SOA jak właśnie usługi sieciowe.

Wszyscy znaczący dostawcy narzędzi do tworzenia oprogramowania wspierają obecnie tworzenie rozwiązań zorientowanych na usługi i obsługa ta jest najczęściej rozumiana jako środki do tworzenia usług sieciowych. Doceniając zatem w pełni fakt, że osiągnięcie celów SOA niekoniecznie musi prowadzić poprzez usługi sieciowe, w tej książce skupiamy się na takiej właśnie drodze ich osiągania.

3.1.7. Pierwotna SOA

W kilku ostatnich punktach opisaliśmy poszczególne elementy tego, co nazywane jest **pierwotną SOA** (*primitive SOA*). Jak słusznie można wnioskować, jest to pewna „linia odniesienia", czyli zasób technologii oczekiwanych od współczesnych dostawców platform programistycznych.

Wszystkie postaci SOA, jakie opisywać będziemy dalej w tej książce, bazują na tym pierwotnym modelu i stanowią jego rozszerzenie. Niektóre z opisywanych rozszerzeń tej grupy stanowią wynik zastosowania zaawansowanych technik projektowych, inne natomiast opierają się na dostępności predefiniowanych usług sieciowych i wsparciu dla nich ze strony dostawców rozwiązań.

ANALIZA PRZYPADKU

System finansowo-księgowy firmy RailCo jest dwuwarstwową aplikacją typu klient-serwer, przy czym zdecydowana większość logiki biznesowej zlokalizowana jest w programach zainstalowanych i uruchamianych na stacjach klienckich. Szczegóły tej aplikacji opiszemy w następnym rozdziale, tu ograniczymy się do dwóch jej najważniejszych zadań, którymi są:

- wprowadzenie zamówienia klienta,
- utworzenie zamówienia klienckiego.

Wykonanie każdego z tych zadań obejmuje serię kroków realizujących proces biznesowy. Proces ten był modelowany za pomocą logiki standardowego przepływu pracy, a następnie zaimplementowany w postaci pakietu. Mimo iż na poziomie kodu źródłowego poszczególne aspekty procesu zostały odzwierciedlone w postaci odrębnych modułów (podprogramów), całość została ostatecznie skompilowana do postaci pojedynczego modułu wykonywalnego, automatyzującego wspomniany proces według ustalonej a priori aranżacji.

Na gruncie modelu zorientowanego usługowo jest zgoła inaczej: logika kryjąca się za określonym procesem powinna zostać zrealizowana w postaci zespołu usług (w szczególności — pojedynczej usługi), automatyzacja zaś całego procesu powinna zostać wykonana jako kompozycja tychże usług. Każda z usług powinna reprezentować podproces czy nawet pojedynczy jego etap i powinna być wykonywana niezależnie od pozostałych. Przykładowo proces utworzenia zamówienia klienckiego może składać się z następujących podprocesów:

- pobranie danych zamówienia,
- sprawdzenie dostępności towaru,
- utworzenie dokumentu zamówienia,
- zapisanie dokumentu zamówienia w bazie.

Jak wiemy, proces jako całość może być traktowany (i modelowany) jako usługa. Niekiedy kilka procesów można połączyć tak, że reprezentować będą pojedynczą usługę. Przykładowo procesy utworzenia dokumentu zamówienia i wygenerowania faktury dla klienta mogą zostać połączone w pojedynczy proces przetwarzania zamówienia. Wreszcie, możemy zaprojektować wspomniane procesy jako na tyle elastyczne, że zdolne wykorzystywać procesy lub zasoby dostępne gdziekolwiek w przedsiębiorstwie. Przykładowo możemy do procesu przetwarzania zamówienia dodać podproces automatycznie wyszukujący adres e-mailowy klienta; ten podproces może także istnieć jako część większego procesu raportowania dla klientów.

Aby zaimplementować ten model, potrzebujemy architektury technicznej zdolnej do:

- podziału logiki automatyzacji biznesu na jednostki właściwie reprezentujące poszczególne usługi,
- zapewnienia względnej niezależności wspomnianych jednostek logicznych, dzięki czemu spełniać będą one postulat zdolności do wielokrotnego wykorzystywania, w różnych kompozycjach,
- umożliwienia wspomnianym jednostkom logicznym komunikowania się w sposób zachowujący ich niezależność.

Fundamentalna charakterystyka enkapsulacji logiki w ramach usług, luźnego powiązania i komunikowania, zrealizowana zgodnie z zasadami zorientowania na usługi i zaimplementowana na bazie usług sieciowych, jest właśnie spełnieniem powyższych wymagań w ramach implementacji pierwotnej SOA.

PODSUMOWANIE

- SOA i zorientowanie na usługi to oderwane od implementacji paradygmaty projektowe, przeznaczone do realizacji na dowolnej, przydatnej do tego platformie.
- Model „pierwotnej SOA" reprezentuje zestaw odmian SOA bazujących wyłącznie na usługach sieciowych i powszechnie przyjętych zasad zorientowania na usługi.
- Ilekroć dalej w tej książce użyjemy terminu SOA, będzie on odnosił się do modelu pierwotnej SOA.

3.2. Ogólnie o współczesnej SOA

Pierwotny model SOA to dopiero początek. Ostatnie trendy przemysłowe i konkretne produkty nadały SOA nowe oblicze. Podstawowe zasady modelu pierwotnego zostały zachowane, jednak wiele z nich poszerzono głównie ze względu na spodziewane korzyści i — oczywiście — techniczne możliwości urzeczywistnienia abstrakcji.

Znaczący dostawcy oprogramowania wciąż wykazują niespożytą inwencję w formułowaniu nowych specyfikacji usług sieciowych i tworzeniu coraz większego wsparcia dla tychże usług oraz standardu XML na współczesnych platformach technologicznych. I właśnie to ukształtowało nową, rozszerzoną odmianę architektury zorientowanej na usługi — odmianę tę nazywać będziemy **współczesną SOA** (*contemporary SOA*).

Współczesna SOA wyrosła więc na gruncie jej podstawowego modelu, jako efekt spożytkowania nowych możliwości technologicznych i trendów przemysłowych — na drodze do ideału, o którym pisaliśmy w rozdziale 1. I choć zmieniają się (i będą się zmieniać) technologie implementacyjne, pod względem koncepcyjnym współczesną SOA można uznać za ustabilizowaną — w tym sensie, że:

- jest najważniejsza spośród platform zorientowanych na usługi,
- przyczynia się do podwyższenia jakości usług,
- jest zasadniczo autonomiczna,
- bazuje na otwartych standardach,
- wspiera zróżnicowanie dostawców,
- wspiera naturalne współdziałanie,
- propaguje wykrywalność,
- propaguje federacyjność,
- propaguje komponowalność architektoniczną,
- sprzyja wrodzonej wielokrotnej używalności,
- eksponuje rozszerzalność,
- wspiera paradygmat zorientowanego usługowo modelowania biznesu,
- implementuje warstwy abstrakcji,
- promuje luźne powiązania komponentów infrastruktury IT,
- propaguje zwinność organizacyjną,
- jest budulcem,

- oznacza ewolucję,
- jest modelem wciąż dojrzewającym,
- jest dążeniem do ideału.

Zwróćmy uwagę na fakt, iż na powyższej liście brakuje tradycyjnych wartości w rodzaju „bezpieczna", „transakcyjna", „niezawodna" itp. Ten „brak" jest jednak tylko pozorny, przymioty te są elementem „podwyższenia jakości usług", wymienionego na drugiej pozycji. W rozdziałach 6. i 7. opiszemy szczegółowo, jak ewoluujące specyfikacje usług sieciowych ujmują typowe wymagania jakości usług (QoS — *Quality of Service*), a teraz w kolejnych punktach wyjaśnimy dokładniej znaczenie każdej z wymienionych cech, wieńcząc opis sformułowaniem definicji współczesnej SOA.

3.2.1. Współczesna SOA jest najważniejszą spośród platform zorientowanych na usługi

Zanim rozważać będziemy rzeczywiste znaczenie „współczesnej SOA", zobaczmy najpierw, jak w kręgach przemysłu IT traktowany jest sam akronim „SOA". Otóż w powszechnym odczuciu zdaje się on wykraczać poza obszar wyznaczany przez ostatnią jego literę — architekturę; gdy mowa o „produkcie SOA", „projektowaniu SOA" czy „technologii SOA" można odnieść wrażenie, iż słuchamy symfonii z nowego świata — świata całkiem nowej platformy aplikacyjnej.

To wyraźne odstępstwo od tradycji: dotychczas ilekroć danemu określeniu towarzyszył przedrostek (przyrostek) „architektura" w rozmaitych odmianach, było oczywiste, iż mamy do czynienia ze stricte architektonicznym aspektem tegoż określenia. Tak było na przykład z określeniami „klient-serwer" czy „n-warstwowy" — zależnie od kontekstu mogły one odnosić się do narzędzi, infrastruktury administracyjnej czy (właśnie) architektury aplikacji.

I na tę wieloaspektową modłę można by równie dobrze potraktować akronim SOA, gdyby nie fakt, że organicznie tkwi już w nim konkretny — architektoniczny — aspekt. A mimo to, gdy tylko mowa o platformie opartej na usługach sieciowych i zaprojektowanej z uwzględnieniem zasad orientacji na usługi, natychmiast zaczyna wokół pobrzmiewać wielogłosowa fuga na temat „S-O-A".

Jest więc jak jest, warto jednak zastanowić się, jak być powinno — by było prawdziwie, czyli bez niepotrzebnych nieporozumień. Chyba najlepiej będzie, by (poprzedzający cokolwiek) przedrostek SOA oznaczał, że „to coś" stworzone zostało przy (bezpośrednim lub pośrednim) wsparciu ze strony architektury bazującej na zasadach zorientowania na usługi. Ta konwencja tyczy się także niniejszej książki: mimo iż w tytule widnieje słowo „Architektura", jej treść wykracza jednak poza ramy czysto architektoniczne, z intencją możliwie szerokiego opisania współczesnych platform zorientowanych na usługi.

Ponieważ umiejscowiliśmy „współczesną SOA" jako koncepcję wyrosłą na gruncie pierwotnego modelu SOA i stanowiącą jego rozszerzenie, możemy pokusić się o zaczątek naszej definicji:

> *Współczesna SOA reprezentuje architekturę propagującą zorientowanie na usługi z wykorzystywaniem usług sieciowych.*

3.2.2. Współczesna SOA przyczynia się do podwyższenia jakości usług

Naturalnym wymaganiem stawianym SOA staje się zapewnienie realizacji jej elementów funkcjonalnych na (co najmniej) takim poziomie bezpieczeństwa i niezawodności, jakiego gwarancję dają obecne, uznane instancje architektur rozproszonych. Wymaganie to skonkretyzować można w kategoriach typowych dla „jakości usług" (QoS — *Quality of Service*), czyli między innymi:

- zdolności do realizowania zadań w sposób bezpieczny, a zatem z zapewnieniem zarówno ochrony treści komunikatów, jak i kontroli dostępu do poszczególnych zasobów;
- niezawodnej komunikacji między zadaniami, a zatem gwarancji, że każdy wysłany komunikat albo zostanie poprawnie dostarczony, albo też przypadek jego niedostarczenia skwitowany będzie stosownym powiadomieniem;
- wystarczającej wydajności gwarantującej, że narzut ze strony protokołu SOAP i przetwarzania treści XML nie spowoduje pogorszenia efektywności realizowanych zadań;
- transakcyjności chroniącej integralność określonych zadań biznesowych oraz gwarantującej wykonanie „awaryjnego" kodu w przypadku niepowodzenia w wykonywaniu zadania.

Od współczesnej SOA oczekuje się więc wypełnienia luki w zakresie QoS, charakterystycznej dla pierwotnego modelu SOA. Wiele koncepcji i specyfikacji omawianych w części II obejmuje elementy, które dają się odnieść bezpośrednio do wymienionych powyżej wymagań. Z braku bardziej zwięzłego przymiotnika implementacje SOA czyniące zadość tym wymaganiom określać będziemy jako „zdolne do zapewnienia jakości usług" lub krócej „korzystne jakościowo" (w anglojęzycznej literaturze przedmiotu zaletę tę określa się jako *QoS-capable*).

3.2.3. Współczesna SOA jest zasadniczo autonomiczna

Na mocy jednej z prezentowanych reguł zorientowania na usługi poszczególne usługi powinny być od siebie niezależne i samowystarczalne tak dalece, jak tylko to możliwe — tylko wówczas każda z usług będzie mogła sprawować wyłączną kontrolę nad enkapsulowaną przez siebie logiką. Postulat niezależności sięga jeszcze dalej: autonomii i samowystarczalności wymaga się również od *komunikatów* wymienianych między usługami, dzięki czemu każdy komunikat wyposażyć można w zasób inteligencji niezbędny do kontroli tego, jak komunikat ów zostanie przetworzony przez usługę-adresata.

SOA wykorzystuje i rozszerza tę zasadę, propagując koncepcję autonomii na całość rozwiązań automatyzacji w przedsiębiorstwie. Przykładowo aplikacje zbudowane z autonomicznych usług same mogą być traktowane jak złożone, samowystarczalne usługi, realizujące swą samowystarczalność w zintegrowanym środowisku zorientowanym na usługi.

Nieco później wyjaśnimy, jak poprzez kreowanie warstw abstrakcji usług uzyskać można wyłączną kontrolę nad określonymi domenami tematycznymi — i w rezultacie osiągnąć poziom autonomii przekraczający granice poszczególnych rozwiązań.

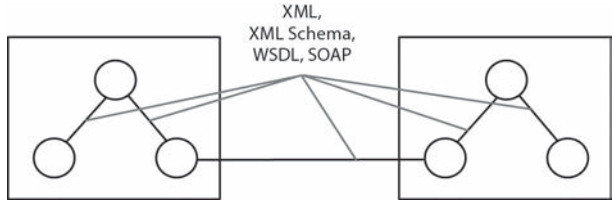
3.2.4. Współczesna SOA bazuje na otwartych standardach

Bodaj jednak najbardziej znaczącym atrybutem usług sieciowych jest fakt, że wymiana danych między usługami zorganizowana jest na bazie otwartych standardów. Komunikat wysłany przez jedną usługę do innej obsługiwany jest na bazie protokołów globalnie zestandaryzowanych i powszechnie zaakceptowanych.

Zestandaryzowany jest także każdy z komunikatów zarówno pod względem formatu, jak i sposobu reprezentowania ładunku użytecznego. Użycie SOAP, WSDL, XML i XML Schema pozwala na zachowanie samodzielności komunikatów i uzgodnienie wspólnych zasad komunikacji, dzięki czemu komunikujące się usługi nie muszą wiedzieć o sobie nic ponad to, co wynika z ich opisów.

Wykorzystywanie otwartego, standardowego modelu komunikacyjnego eliminuje konieczność współdzielenia systemu typów[1] przez logikę usług i generalnie wspiera paradygmat luźnego powiązania.

Współczesna SOA w pełni spożytkowuje ten framework komunikacyjny, niezależny od konkretnych dostawców (patrz rysunek 3.5). SOA redukuje rolę specyficznych, własnościowych technologii jedynie do implementacji i hostowania logiki aplikacyjnej enkapsulowanej przez usługę. Powoduje to, że komunikacja między usługami zawsze jest dostępną opcją.



Rysunek 3.5. Standardowe, otwarte technologie wykorzystywane są zarówno na gruncie poszczególnych rozwiązań, jak i w relacji pomiędzy nimi

3.2.5. Współczesna SOA wspiera zróżnicowanie dostawców

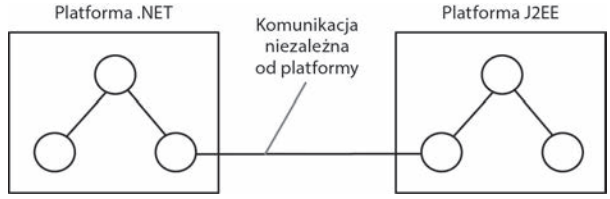
Opisywany w poprzednim punkcie otwarty framework komunikacyjny nie tylko stwarza pomost do komunikowania się heterogenicznych środowisk, w ramach przedsiębiorstwa i między przedsiębiorstwami, lecz także ułatwia wybór środowiska najlepiej dostosowanego do potrzeb konkretnej aplikacji.

Przykładowo, niezależnie od tego, w jakim konkretnym środowisku programistycznym tworzona jest dana aplikacja, jeżeli tylko środowisko to zapewnia wsparcie dla standardowych usług sieciowych, wspomnianą aplikację można wyposażyć w warstwę standardowego interfejsu, otwierającego drogę do współdziałania z innymi aplikacjami tego rodzaju (patrz rysunek 3.6). Przy okazji nadaje to nowe oblicze architekturze integracyjnej, która teraz, za pośrednictwem odpowiednich „usługowych" adapterów, enkapsulować może także starszą („spadkową") logikę oraz spożytkowywać zalety rozmaitego oprogramowania typu middleware[2].

---

[1] Pod pojęciem „systemu typów" autor rozumie tu zapewne pewne zdefiniowane a priori zasady operowania typami danych, powszechne chociażby w językach trzeciej generacji (Fortranie, Algolu czy wczesnych wersjach Pascala): jeśli weźmiemy pod uwagę na przykład przekazywanie parametrów do procedur i funkcji (podprogramów), zauważymy, że w wygenerowanym przez kompilator kodzie binarnym nie ma żadnej informacji o typie przekazywanej wartości, jest tylko przyjęte założenie o jej typie, odzwierciedlone przez kompilator w treści przekładu. W opisywanym przypadku luźnego powiązania między usługami jest diametralnie inaczej — w komunikacie wymienianym między usługami znajduje się kompletna informacja na temat przekazywanej wartości (czyli również informacja o jej typie) i nie ma odniesienia do jakiegokolwiek predefiniowanego „zewnętrznego" systemu typów (gdyby odniesienie takie istniało, powiązanie między usługami nie byłoby już tak luźne, bo skrępowane koniecznością ich stosowania się do domyślnych „zewnętrznych" konwencji — *przyp. tłum.*

[2] Określenie *middleware* (skrót od ang. [*in*] *middle software*, dosł. *oprogramowanie pośrednie*) używane jest w dwóch znaczeniach: jako oprogramowanie o charakterze *pośrednim* między wysokopoziomowymi mechanizmami aplikacyjnymi a niskopoziomowymi funkcjami systemu operacyjnego lub jako oprogramowanie *pośredniczące* między dwoma (lub więcej) obiektami, zapewniające ich współdziałanie. W tej książce występować będzie ono głównie w tym drugim znaczeniu — *przyp. tłum.*



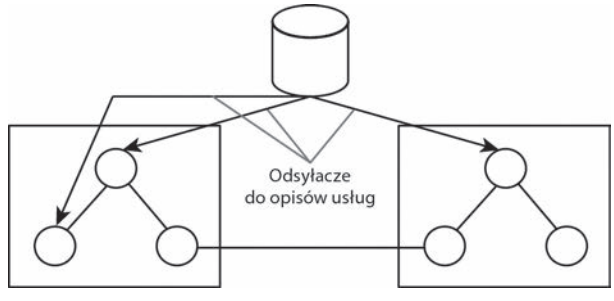
Rysunek 3.6. Różnica technologiczna między platformami zorientowanymi na usługi nie stanowi przeszkody we współdziałaniu tych platform

Przedsiębiorstwa mają więc możliwość korzystania w dalszym ciągu z istniejących rozwiązań, narzędzi programistycznych i zaplecza serwerowego — oczywiście, ma to sens wówczas, gdy przyczynia się do spożytkowania zalet tkwiących w firmowych zasobach. Równocześnie warto jednak przyglądać się ofertom nowych dostawców. Taka elastyczność jest konsekwencją przyjęcia otwartych technologii, nieodłącznie związanych z frameworkiem usług sieciowych; dzięki standaryzacji i zasadom wprowadzanym przez SOA można tę elastyczność osiągnąć łatwiej niż poprzednio.

3.2.6. Współczesna SOA propaguje wykrywalność

Mimo iż standard UDDI pojawił się już w pierwszej generacji usług sieciowych, niewiele ówczesnych środowisk wykorzystywało rejestrowanie usług jako swój integralny element. Może dlatego, że tylko nieliczne z budowanych usług uwzględniały możliwość rejestrowania; bardziej prawdopodobną przyczyną był jednak fakt, że sama koncepcja rejestrowania usług tradycyjnie nie istniała w ówczesnych architekturach rozproszonych — usługi sieciowe były raczej wykorzystywane na potrzeby rozwiązań typu punkt-punkt i koncepcja wykrywania usługi nie wydawała się specjalnie interesująca.

SOA diametralnie zmienia ten stan rzeczy, propagując ogłaszanie i wykrywanie usług, w skali całego przedsiębiorstwa i poza jego granicami. Każda niebanalna implementacja SOA wykorzystuje prawdopodobnie jakąś formę rejestru czy katalogu jako narzędzia do zarządzania opisami usług (patrz rysunek 3.7).

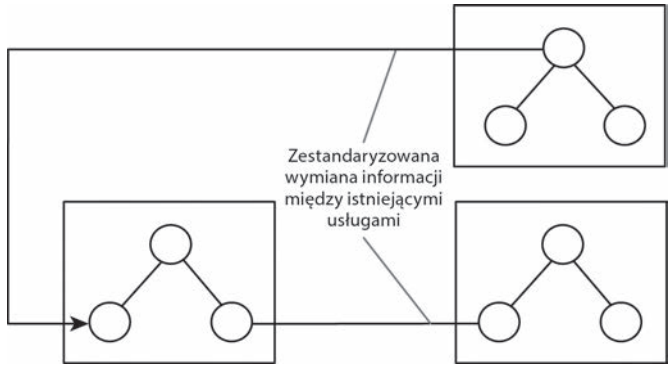


Rysunek 3.7. Rejestr jako mechanizm umożliwiający wykrywanie usług

3.2.7. Współczesna SOA wspiera naturalne współdziałanie

Na bazie opisywanego wcześniej wymogu używania otwartych standardów, uniezależniania środowiska od konkretnych dostawców i zapewnienia wzajemnej wykrywalności usług rodzi się nowa koncepcja — naturalne współdziałanie. Niezależnie od tego, czy konkretna aplikacja charakteryzuje się własnymi wymogami pod względem integracji z innymi mechanizmami, zasady projektowania usług przemawiają za ich wyposażeniem w cechy naturalnie propagujące współdziałanie.

W aplikacji SOA tworzonej „od zera" usługi posiadające zdolność naturalnego współdziałania stają się potencjalnymi węzłami integracji (patrz rysunek 3.8). Przy właściwej standaryzacji prowadzi to do architektury integracyjnej opartej na usługach, w ramach której gotowe rozwiązania same z siebie uzyskują wrodzoną zdolność współdziałania z innymi rozwiązaniami. Przekłada się to w prostej linii na zmniejszenie kosztów i wysiłku w przyszłych zabiegach integracyjnych wynikających z wymogów nowych aplikacji.

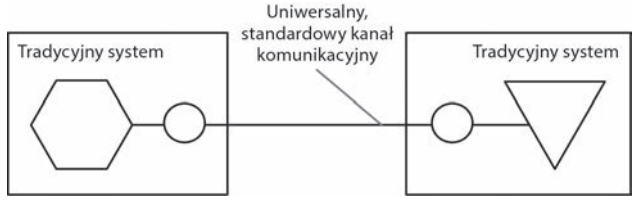


Rysunek 3.8. Usługi posiadające wrodzoną zdolność współdziałania stwarzają nieocenione możliwości w zakresie integracji

3.2.8. Współczesna SOA propaguje federacyjność

Budowanie SOA wewnątrz przedsiębiorstwa nie musi oznaczać rezygnacji ze starszych, z powodzeniem wykorzystywanych rozwiązań. Jedną z najbardziej atrakcyjnych cech SOA jest możliwość jednoczenia środowisk, które dotąd funkcjonowały odrębnie. Mimo iż taką *federację* umożliwiały już usługi sieciowe, SOA podwyższa rangę tej możliwości poprzez mechanizm enkapsulacji logiki starszych, tradycyjnych aplikacji i eksponowanie tejże logiki za pośrednictwem powszechnie używanego, otwartego i standardowego frameworku komunikacyjnego (tworzenie takich frameworków możliwe jest dzięki bogatej ofercie rozmaitych adapterów).

Oczywiście, prowadzi to do powstawania rozmaitych rozwiązań hybrydowych. Najważniejsze jest jednak to, że kanały komunikacji ustanowione w drodze takiej integracji zorientowanej na usługi są w pełni standardowe i ujednolicone (patrz rysunek 3.9).



Rysunek 3.9. Usługi umożliwiają standardowe federację tradycyjnych, różniących się od siebie systemów

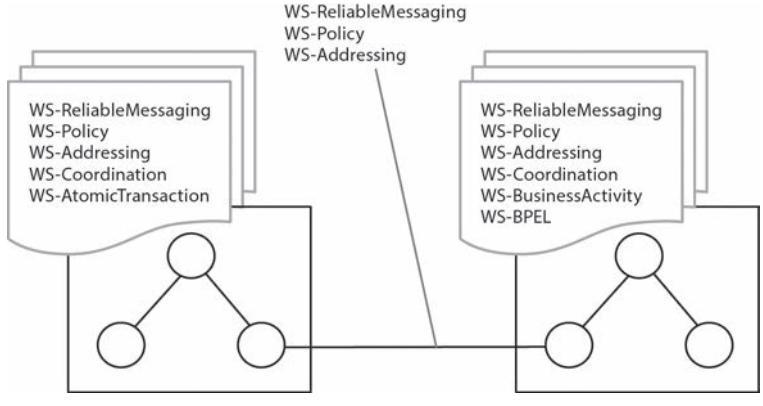
3.2.9. Współczesna SOA propaguje komponowalność architektoniczną

Komponowalność to głęboko zakorzeniona cecha SOA, która realizować się może na różnych poziomach. Przykładowo dzięki możliwości tworzenia i rozwijania usług komponowalnych SOA wspomaga automatyzację elastycznych, adaptowalnych procesów biznesowych. Jak wielokrotnie

podkreślaliśmy, poszczególne usługi egzystują jako niezależne jednostki logiczne; proces biznesowy może więc zostać rozbity na szereg usług, z których każda reprezentować będzie pewien logiczny aspekt tego procesu.

W sposób bardziej spektakularny przejawia się komponowalność na gruncie frameworku usług sieciowych drugiej generacji, który to framework wyewoluował z biegiem czasu do obecnej postaci licznych specyfikacji grupy WS-*. Modularna natura tych specyfikacji pozwala na tworzenie rozwiązań SOA poprzez selektywny dobór tylko tych komponentów, które faktycznie są w danym przypadku potrzebne.

Tym, co dostarcza owej elastyczności, jest fakt, że wspomniane specyfikacje usług sieciowych drugiej generacji zaprojektowane zostały z myślą o wykorzystaniu zalet modelu komunikacyjnego SOAP. Poszczególne specyfikacje składają się z modularnych rozszerzeń oferujących specyficzne mechanizmy. Ponieważ oferta wsparcia dla wspomnianych rozszerzeń ze strony dostawców systematycznie się zwiększa, coraz większa staje się elastyczność w doborze niezbędnych komponentów (patrz rysunek 3.10). Innymi słowy, platformy obsługujące WS-* umożliwiają tworzenie eleganckich i zoptymalizowanych architektur, aplikacji, usług, a nawet komunikatów.

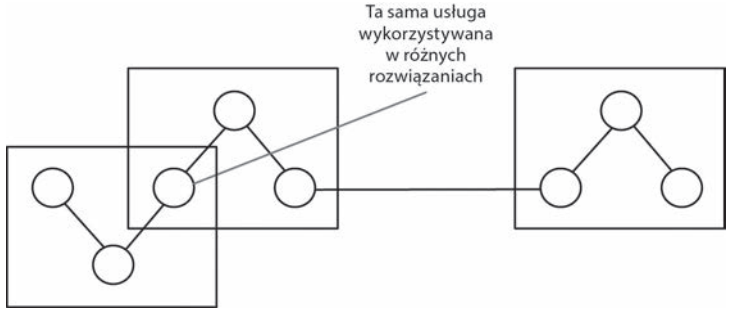


Rysunek 3.10. Różne rozwiązania mogą być budowane jako odmienne kompozycje rozszerzeń i mogą ze sobą współdziałać, o ile zawierają wspólne komponenty niezbędne do tego współdziałania

Zgodnie z naszą definicją współczesnej SOA, odzwierciedlimy opisaną cechę poprzez opisanie całej architektury jako „komponowanej". Oznacza to zarówno usługi komponowalne, jak i rozszerzenia składające się na rozszerzenia konkretnych implementacji SOA.

3.2.10. Współczesna SOA sprzyja wrodzonej wielokrotnej używalności

SOA ustanawia środowisko propagujące na wielu poziomach wielokrotne wykorzystywanie usług. Po pierwsze, usługi tworzone zgodnie z zasadami zorientowania na usługi są z natury wielokrotnie używalne, nawet jeżeli nie jest to absolutnie konieczne, z punktu widzenia ich zastosowań. Po drugie, kolekcje usług stanowiące *de facto* usługi złożone, same mogą być elementami składowymi większych (super)kolekcji. Po trzecie, ta sama usługa wchodzić może w skład części wspólnej dwóch różnych kompozycji (tak jak na rysunku 3.11).

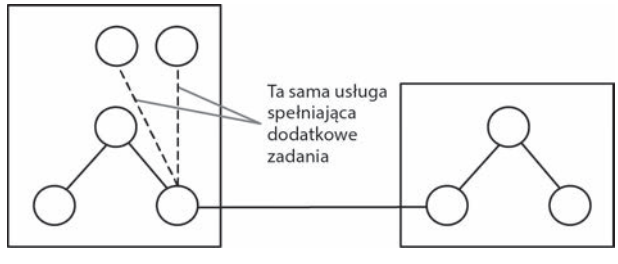


Rysunek 3.11. Przyrodzona wielоużywalność usług stwarza nieocenione możliwości w zakresie ich wielokrotnego wykorzystywania

Eksponowana przez SOA niezależność tworzonych usług od procesów biznesowych i rozwiązań automatyzacji, na potrzeby których usługi te mają być wykorzystywane, prowadzi do powstawania środowisk, w których wielоużywalność dostarczanych usług jest korzystnym efektem ubocznym. Budowanie rozwiązań zorientowanych na usługi jest więc naturalną promocją wbudowanej wielоużywalności.

3.2.11. Współczesna SOA eksponuje rozszerzalność

Organizując wyrażanie funkcjonalności usług poprzez ich opisy, SOA skłania do postrzegania ich w sposób wykraczający poza tradycyjny schemat komunikacji punkt-punkt. Przy właściwym podziale logiki problemu między usługi i umiejętnym zaprojektowaniu ich interfejsów, na odpowiednim stopniu granulacji, zakres funkcjonalności oferowanej przez daną usługę można często rozszerzyć bez łamania istniejących interfejsów (patrz rysunek 3.12).



Rysunek 3.12. Rozszerzalne usługi umożliwiają wzbogacanie oferowanej funkcjonalności przy minimalnym wpływie na inne komponenty

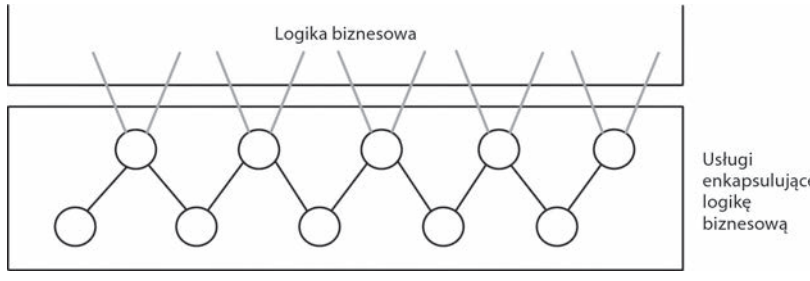
Na gruncie SOA rozszerzalność obejmuje nie tylko pojedyncze usługi, ale wręcz przenika całą architekturę. Istniejące rozwiązania mogą być wzbogacane funkcjonalnie przez dodawanie nowych usług lub przez łączenie z innymi aplikacjami zorientowanymi na usługi (co ostatecznie skutkuje tym samym — dodawaniem usług). Ponieważ luźne powiązanie między usługami minimalizuje ich wzajemne zależności, ich wzbogacanie nie powoduje znaczącego zakłócenia istniejącego układu.

Czas zatem powrócić do naszej początkowej definicji „współczesnej SOA” i rozszerzyć ją o kilka istotnych przymiotników:

Współczesna SOA reprezentuje ***otwartą, rozszerzalną, federacyjną*** *i* ***komponowalną*** *architekturę propagującą zorientowanie na usługi,* ***niezależną od konkretnych dostawców, złożoną z usług autonomicznych, współdziałających, wykrywalnych*** *i* ***potencjalnie wielokrotnie używalnych, zdolną do polepszania jakości tych usług,*** *zaimplementowanych w technologii usług sieciowych.*

3.2.12. Współczesna SOA wspiera paradygmat zorientowanego usługowo modelowania biznesu

Opisując model pierwotnej SOA, wyjaśniliśmy pokrótce, jak procesy biznesowe mogą być reprezentowane i wyrażane przez usługi. Partycjonowanie logiki biznesowej pomiędzy usługi, z których następnie komponować można kompletne rozwiązania, ma niebagatelne konsekwencje pod względem sposobu modelowania procesów biznesowych (patrz rysunek 3.13). Oczywiście, analitycy wykorzystują tę okazję — implementując SOA, niejako z automatu wprowadzają wspomniane partycjonowanie do projektowania procesów biznesowych.



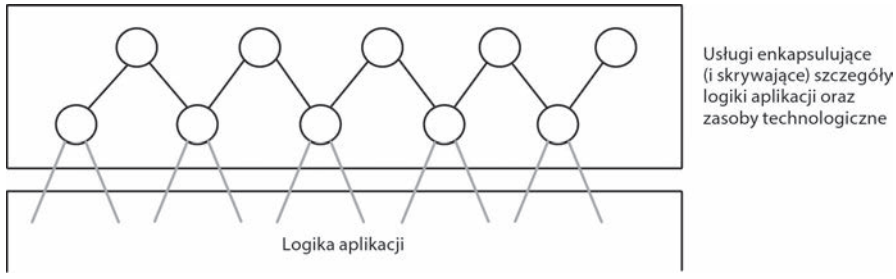
Rysunek 3.13. Kolekcja (warstwa) usług enkapsulująca logikę procesu biznesowego

Innymi słowy, wszelkie modele biznesowe — zarządzania procesami (BPM — *Business Process Management*), podmiotów gospodarczych czy innych form inteligencji biznesowej — mogą być precyzyjnie reprezentowane w formie kompozycji skoordynowanych usług. Niestety, ten paradygmat projektowania nie jest obecnie ani powszechnie przyjęty, ani też należycie zrozumiany; wyjaśnieniu jego istoty poświęcona jest więc znacząca część tej książki.

3.2.13. Współczesna SOA implementuje warstwy abstrakcji

Wśród zasad zorientowanego na usługi projektowania aplikacji jedna z nich ma wyraźnie ewolucyjny charakter — mowa o zasadzie abstrakcyjności. Typowa instancja SOA realizuje warstwy abstrakcji, czyniąc usługi jedynymi punktami dostępu do rozmaitych zasobów oraz logiki przetwarzania.

Abstrakcjonizm ten może być ukierunkowany na logikę biznesową i logikę aplikacji. Przykładowo, tworząc warstwę „punktów końcowych" reprezentujących kompletne rozwiązania i konkretne platformy technologiczne, powodujemy całkowite ukrycie ich szczegółów (patrz rysunek 3.14). Widoczne pozostają tylko elementy funkcjonalne oferowane za pośrednictwem interfejsów usług.



Rysunek 3.14. Szczegóły technologiczne realizujące logikę aplikacji mogą zostać ukryte przez dedykowaną warstwę usług

Abstrakcjonizm, w ramach którego proces biznesowy i logika aplikacji nawzajem skrywają przed sobą swe szczegóły, jest jedną z podstawowych cech paradygmatu projektowania zorientowanego na usługi, omawianego w poprzednim punkcie. Abstrakcjonizm jest także jednym z fundamentów luźnego powiązania między usługami.

3.2.14. Współczesna SOA promuje luźne powiązania komponentów infrastruktury IT

Wyjaśnialiśmy już, że główną korzyścią wynikającą z luźnego powiązania między usługami jest zachowanie niezależności pomiędzy logiką każdej z nich: każdą usługę można tworzyć i rozwijać niezależnie od pozostałych, jedynym czynnikiem wiążącym usługi jest wzajemna świadomość ich istnienia.

Jeśli zastosujemy tę zasadę w skali całego przedsiębiorstwa zarówno do modelowania biznesowego, jak i projektu technicznego, korzyści z luźnego powiązania stają się jeszcze bardziej spektakularne.

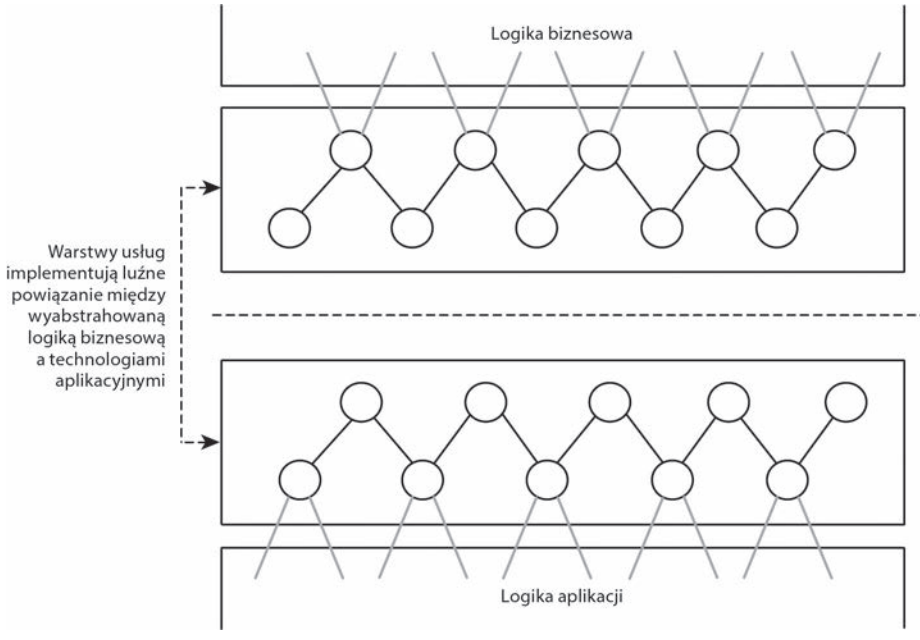
Jeżeli zaimplementujemy zestandaryzowaną warstwę abstrakcji usług, efekt luźnego powiązania wystąpić może również między dwiema domenami przedsiębiorstwa — domeną procesów biznesowych i domeną technologii aplikacyjnych (patrz rysunek 3.15). Obie domeny skrywają nawzajem swe szczegóły i współegzystują wyłącznie na zasadzie świadomości istnienia partnera. Dzięki temu mogą rozwijać się względnie niezależnie od siebie — a to umożliwia wystarczająco szybkie reagowanie na nowe potrzeby biznesu i na pojawiające się wciąż nowe technologie. Formalnie zdolność tę nazywa się zwinnością organizacyjną (*organizational agility*).

3.2.15. Współczesna SOA propaguje zwinność organizacyjną

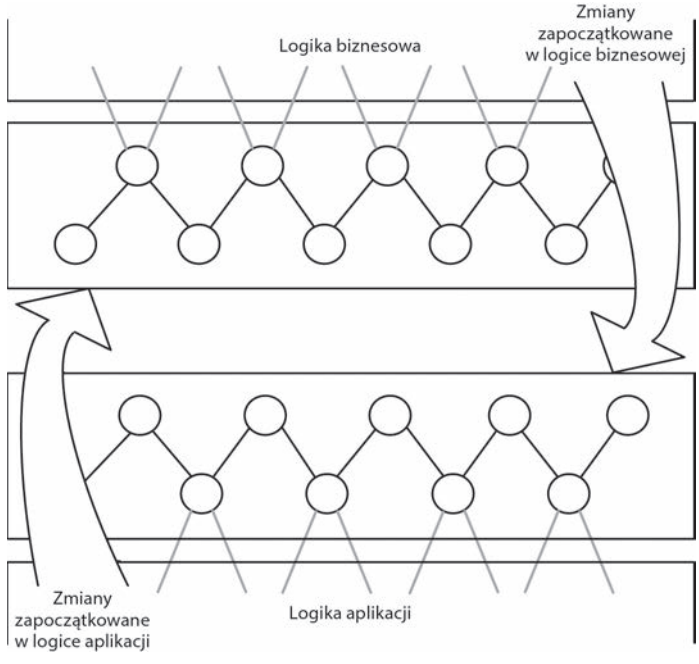
Wewnętrzna reorganizacja, fuzja, zmiana profilu biznesowego, wymiana użytkowanych technologii — zdolność do szybkiego, „zwinnego" reagowania na te i inne (nieprzewidywalne) żywioły świata korporacji może być dla tych korporacji sprawą na miarę „być albo nie być".

Zmiany w logice biznesowej przedsiębiorstwa mają pewien wpływ na technologię aplikacji, które tę logikę automatyzują. I vice versa: zmiana infrastruktury technologicznej ma pewien wpływ na logikę biznesową, której automatyzacji infrastruktura ta służy. Kluczowe znaczenie ma tutaj wymiar słowa „pewien" — im większe wspomniane uzależnienie, tym większy problem z wprowadzaniem zmian i tym większy koszt tego procesu.

Wykorzystując reprezentowanie logiki biznesowej w formie usług, budując warstwę abstrakcji usług i realizując luźne powiązanie między domeną biznesu a domeną aplikacyjną, SOA sprowadza opisaną zależność do minimum, bo wydatnie zwiększa zwinność organizacyjną firmy (patrz rysunek 3.16).



Rysunek 3.15. Dzięki implementacji warstwy usług, separującej logikę biznesową od logiki aplikacji, paradygmat luźnego powiązania zrealizowany zostaje na poziomie całego przedsiębiorstwa



Rysunek 3.16. Relacja luźnego powiązania między biznesem a technologią aplikacyjną pozwala każdej z tych domen efektywniej dostosowywać się do zmian w obrębie drugiej domeny

Istotną rolę w „zwinnym" przeprowadzaniu zmian odgrywają także dwie inne cechy SOA — otwarty framework komunikacji między architekturami integrującymi oraz wbudowana w usługi składowe zdolność do współdziałania z innymi usługami, otwierająca możliwość współdziałania zróżnicowanych platform. Ponadto luźne powiązanie między usługami stanowiącymi „końce" kanału komunikacyjnego powoduje, że jedna z komunikujących się domen pozostaje nieświadoma zmian, jakie dokonują się wewnętrznie w ramach drugiej.

Ogólnie rzecz biorąc, zwinność organizacyjna firmy to bodaj największa z korzyści, jakie daje współczesna SOA.

3.2.16. Współczesna SOA jest budulcem

Migracja firmy w stronę SOA jest procesem długotrwałym i architektura aplikacji zorientowanych na usługi jest prawdopodobnie tylko jedną z kilku (wielu) platform architektonicznych w tej firmie. Jedną z kilku, ale ekspansywną: firma, sukcesywnie transformując swą infrastrukturę w kierunku SOA, zmierza ku ideałowi, jakim jest **przedsiębiorstwo zorientowane na usługi** (SOE — *Service Oriented Enterprise*), w którym *wszystkie procesy biznesowe* realizowane są na bazie usług zarówno pod względem koncepcyjnym, jak i pod względem fizycznej implementacji.

Z perspektywy SOE granice funkcjonalne SOA wyznaczają już fragment tego przyszłego ideału — fragment stanowiący bądź to wydzielony sektor biznesu, bądź też usługę enkapsulującą część lub całość logiki automatyzacji. W konsekwencji zmian na poziomie modelu biznesowego obszar SOA może albo zostać poszerzony, albo też stać się częścią architektury integracyjnej warunkującej współdziałanie wielu aplikacji.

Konkretna aplikacja zorientowana na usługi — jako całość —może więc być reprezentowana przez pojedynczą usługę i modelowana jako taka. Jak wcześniej wyjaśnialiśmy, rozmiar logiki enkapsulowanej przez usługę jest praktycznie nieograniczony: nieograniczony jest więc także poziom zagnieżdżania usług — SOA może składać się z usług, które są kolekcjami usług będących kolekcjami usług, itd. W tym ujęciu dane rozwiązanie oparte na SOA może być traktowane jako pojedyncza usługa stanowiąca część SOE.

Opisana właściwość wymaga rozszerzenia naszej definicji współczesnej SOA — dołączmy do niej następujące stwierdzenie:

> *SOA zdolna jest do stworzenia abstrakcji logiki biznesowej i architektury technologicznej, przez co między tymi dwiema domenami powstaje relacja luźnego powiązania. Dzięki temu zmiany w obrębie jednej z tych domen wywołują niewielki wpływ na konieczne zmiany w drugiej. Sprzyja to płynnemu wdrażaniu zorientowania na usługi na ewolucyjnej drodze do przedsiębiorstwa zorientowanego na usługi (SOE).*

3.2.17. Współczesna SOA oznacza ewolucję

Architektura definiowana przez SOA kultywuje, co prawda, tradycje swych poprzedników, lecz jest od nich zasadniczo różna: od klasycznej architektury klient-serwer i obliczeń rozproszonych różni się przede wszystkim zasadami projektowymi oraz ukierunkowaniem na usługi sieciowe. Można zaryzykować stwierdzenie, że SOA, zachowując zalety platform-poprzedników, buduje na ich fundamencie nową platformę, opartą na nowatorskich wzorcach projektowych i implementowaną przy użyciu nowoczesnych technologii.

Wśród wspomnianych zalet wymienić należy przede wszystkim wielokrotne używanie oraz partycjonowanie logiki aplikacji na dobrze zdefiniowane komponenty. Znalazły one swe wyraźne odzwierciedlenie wśród zasad zarówno pierwotnego, jak i współczesnego modelu SOA.

3.2.18. Współczesna SOA jest modelem wciąż dojrzewającym

Choć opisane dotychczas właściwości mają znaczenie fundamentalne dla współczesnej SOA, trzeba przyznać, iż w rzeczywistości stanowią raczej subiektywną konstatację stanu, w jakim SOA znajduje się obecnie. Jeśli nawet SOA ma już ustabilizowaną reputację jako bezsprzecznie przyszłościowa platforma dla aplikacji komputerowych, jej gotowość do objęcia tej zaszczytnej roli jest jeszcze problematyczna. Mimo iż usługi sieciowe z powodzeniem wykorzystywane są do implementowania funkcjonalności aplikacji, to wiele z mechanizmów niezbędnych do „poważnych" obliczeń w przedsiębiorstwach wciąż nie jest w pełni dostępnych.

Multum specyfikacji opracowanych przez organizacje standaryzacyjne, w ścisłej współpracy z dostawcami oprogramowania, formalnie reguluje kwestie wielu potrzebnych rozszerzeń; nowoczesne narzędzia programistyczne i serwery aplikacji prezentują się obiecująco pod względem wsparcia technicznego dla nowych koncepcji. Gdy już platformy SOA i towarzyszące im narzędzia osiągną adekwatny poziom dojrzałości, wykorzystanie usług sieciowych w tworzeniu rozwiązań SOA stanie się powszechne i osiągnięcie ideału SOE stanie się perspektywą bardziej realną.

Przymierzając się zatem do sformułowania definicji odzwierciedlającej prawdziwy obraz „współczesnej SOA", nie sposób nie odnieść się do stanu rozwoju niezbędnych technologii. Zważywszy jednak na tempo, w jakim ogólnie rozumiany przemysł IT przejmuje platformy SOA i dokonuje ich doskonalenia, można mieć nadzieję, że już niedługo formułowanie tego typu zastrzeżeń stanie się niepotrzebne.

3.2.19. Współczesna SOA jest dążeniem do ideału

Pełne zaadaptowanie SOA w skali całego przedsiębiorstwa to ideał, który wiele firm chciałoby osiągnąć jak najszybciej. Realia są jednak takie, iż transformacja taka wymaga nie lada wysiłku, dyscypliny i — zależnie od rozmiaru firmy — odpowiednio długiego czasu. Każde środowisko techniczne doznaje rozmaitych zmian w trakcie tego typu migracji, a różne części SOA w odmiennym tempie przechodzą przez kolejne fazy rozwoju. Najpewniej prowadzi to do powstawania niezliczonych architektur hybrydowych, czyli stanowiących mieszankę aplikacji tradycyjnych i aplikacji zorientowanych na usługi. To wszystko zbiega się w czasie z rozwojem środków technologicznych, jakie powstają w celu wspomagania realizacji całej transformacji; od ewoluującej firmy wymaga to sukcesywnej modernizacji infrastruktury, stosownie do zmian wynikających z dojrzewania specyfikacji, standardów i produktów związanych z SOA.

Na szczęście, znakomita większość opisanych wcześniej cech SOA jest osiągalna już dziś. W tej książce zamieściliśmy serię tutoriali oraz opisów krok po kroku wyjaśniających, jak cechy te manifestują się w praktycznych zastosowaniach.

3.2.20. Zdefiniowanie SOA

Zakończyliśmy prezentację najważniejszych cech współczesnej SOA, pora więc sformułować ostateczną jej definicję. Przypomnijmy:

SOA reprezentuje otwartą, rozszerzalną, federacyjną i komponowalną architekturę propagującą zorientowanie na usługi, niezależną od konkretnych dostawców, złożoną z usług autonomicznych, współdziałających ze sobą, wykrywalnych i potencjalnie wielokrotnie używalnych, zdolną do polepszania jakości tych usług, zaimplementowanych w technologii usług sieciowych.

SOA zdolna jest do stworzenia abstrakcji logiki biznesowej i architektury technologicznej, przez co między tymi dwiema domenami powstaje relacja luźnego powiązania. Dzięki temu zmiany w obrębie jednej z tych domen wywołują niewielki wpływ na konieczne zmiany w drugiej. Sprzyja to płynnemu wdrażaniu zorientowania na usługi, na ewolucyjnej drodze do przedsiębiorstwa zorientowanego na usługi (SOE).

I uzupełnijmy:

SOA stanowi wynik ewolucji starszych platform, zachowując korzystne cechy tradycyjnych architektur i łącząc je z nowatorskimi zasadami orientacji na usługi, wyznaczającymi drogę do stworzenia przedsiębiorstwa w pełni zorientowanego na usługi.

SOA jest dążeniem do ideału przedsiębiorstwa zorientowanego na usługi (SOE), a osiągnięcie tego ideału wymaga zaplanowanego procesu transformacji oraz wsparcia ze strony wciąż doskonalonych technologii.

Powyższa definicja współczesnej SOA, choć trafna, jest jednak wysoce szczegółowa. Dla celów praktycznych warto sformułować poniższą, bardziej ogólną wersję, stosującą się zarówno do pierwotnej, jak i współczesnej SOA:

SOA jest formą architektury technologicznej, osnutą wokół zasad zorientowania na usługi. Realizowana w technologii usług sieciowych stwarza potencjał dla propagowania i praktycznej realizacji wspomnianych zasad w odniesieniu zarówno do procesów biznesowych, jak i domeny automatyzacji tych procesów.

3.2.21. Podział charakterystyki

Opisane charakterystyki współczesnej SOA podzielimy teraz na dwie grupy: do pierwszej zaliczymy cechy „konkretne", czyli realizowalne jako rzeczywiste rozszerzenia SOA, w drugiej grupie umieścimy natomiast te, które są de facto komentarzami lub wynikami obserwacji. Jedne i drugie okazały się użyteczne w sformułowaniu formalnej definicji, jednakże odtąd interesować nas będą przede wszystkim konkrety. Usuniemy zatem z oryginalnej listy nieco górnolotne stwierdzenia, iż:

- jest najważniejszą spośród platform zorientowanych na usługi,
- jest budulcem,
- stanowi ewolucję,
- jest modelem wciąż dojrzewającym,
- jest dążeniem do ideału.

Ze skonkretyzowanej w ten sposób listy cech wyczytać możemy, że współczesna SOA:

- bazuje na otwartych standardach,
- jest komponowalna architektonicznie,
- jest zdolna sprostać powszechnym wymaganiom jakości usług.

A ponadto wspiera, propaguje lub zapewnia:

- zróżnicowanie dostawców,
- wrodzone współdziałanie,
- wykrywalność,
- federacyjność,
- naturalną wielokrotność,
- rozszerzalność,
- zorientowane na usługi modelowanie procesów biznesowych,
- warstwy abstrakcji,
- luźne powiązanie w skali globalnej,
- zwinność organizacyjną.

Tym właśnie konkretom — gdy są właściwie rozumiane i realizowane — zawdzięczamy rzeczywiste, mierzalne korzyści.

UWAGA

Mimo iż zakwalifikowaliśmy wymienione cechy jako jedynie „konkretną" część definicji, dalej w książce właśnie ten zestaw będziemy utożsamiać z tym, co nazywamy „charakterystyką współczesnej SOA".

PODSUMOWANIE

- Sformułowaliśmy definicję współczesnej SOA w kategoriach zestawu cech charakterystycznych, bazujących na zasadach i cechach pierwotnej SOA i rozszerzających je.
- Praktyczna realizacja charakterystyki współczesnej SOA jest treścią następnych rozdziałów tej książki.

3.3. Najczęstsze nieporozumienia dotyczące SOA

Wspominaliśmy już o tym i w tej książce powtarzać będziemy wielokrotnie, że transformacja (ta prawdziwa) w kierunku SOA to przede wszystkim zmiana sposobu myślenia. A jeśli tak, to również okazja do wielu mitów, nieporozumień, przesądów — te wybrane, które tu krótko opiszemy (i postaramy się zobiektywizować), to jedynie wierzchołek góry lodowej.

3.3.1. „Aplikacja korzystająca z usług sieciowych jest aplikacją zorientowaną na usługi"

Ten mit ma swe źródło w nie do końca precyzyjnej definicji SOA. Bo przecież SOA jako abstrakcyjny model to coś innego niż SOA bazująca na usługach sieciowych i zorientowaniu na usługi. Zależnie od tego, jaki model abstrakcyjny przyjmie się za punkt odniesienia, każdą niemal postać architektury rozproszonej można zaklasyfikować jako „zorientowaną na usługi".

Jeżeli jednak transformacja w stronę SOA ma stać się źródłem tych wszystkich korzyści, które opisywaliśmy w poprzednich punktach, musi opierać się na standaryzacji i umiejscowieniu usług sieciowych zgodnie z zasadami orientacji na usługi.

Naprawdę więc to, czy tytułowe zdanie jest tylko mitem, czy może zawiera jednak ziarno prawdy, zależy przede wszystkim od oczekiwań: możemy traktować tradycyjną architekturę rozproszoną jako „zorientowaną na usługi", jeżeli nie spodziewamy się po niej korzyści, które dać może pierwotna i współczesna SOA.

3.3.2. „SOA to nowe hasło marketingowe dla starych usług sieciowych"

Termin „SOA", jako bezsprzecznie chwytliwy, używany jest często (i nadużywany) do celów marketingowych — zawsze wtedy, ilekroć pojawia się kontekst usług sieciowych. Fakt, że współczesna SOA implementowana jest właśnie przy użyciu usług sieciowych, jest powodem pewnego sceptycyzmu dotyczącego rangi samego akronimu SOA: dla wielu hasło „wsparcie dla SOA" to tylko nowa etykietka tego, co naprawdę jest „wsparciem dla usług sieciowych".

SOA nie jest jednak wynalazkiem medialnej reklamy czy speców od marketingu. Jest to pełnoprawny technicznie termin, o niekwestionowanej reputacji, odnoszący się do specyficznej architektury, opartej na dobrze określonych, fundamentalnych założeniach; to także wykorzystywanie ściśle określonych technologii do realizacji tychże założeń. Technologią z wyboru w tej grupie są właśnie usługi sieciowe, stąd całe nieporozumienie.

3.3.3. „SOA to nowa etykietka dla przetwarzania rozproszonego opartego na usługach sieciowych"

Powszechna wiara w ten mit to nic innego jak syndrom „fałszywej SOA" opisywany w rozdziale 1. I trudno się dziwić: wiele wrzawy wokół SOA przyćmiło jej rzeczywiste znaczenie, a wielu dostawców opatruje etykietką „wsparcie dla SOA" zwyczajną mieszankę tradycyjnych obliczeń rozproszonych z usługami sieciowymi. I już łatwo o nieporozumienie, a sama reputacja terminu „SOA" zostaje poważnie nadszarpnięta.

SOA jest jednak tworem samym w sobie. Owszem, jej zasady pozostają w wyraźnej relacji do niegdysiejszych platform obliczeń rozproszonych, lecz jednocześnie definiują wyraźny kontrast w porównaniu z tymi platformami. Szczegółami tego porównania zajmiemy się w następnym rozdziale.

3.3.4. „SOA upraszcza przetwarzanie rozproszone"

Same zasady tworzące fundament SOA są z natury nieskomplikowane. Często jednak to, o czym łatwo się mówi, znacznie trudniej zrealizować w praktyce: wcielenie wspomnianych zasad w życie może być naprawdę trudnym zadaniem. Owszem, spodziewane korzyści to gra warta świeczki; wygrana w tej grze uwarunkowana jest jednak przeprowadzeniem dogłębnej analizy i dochowaniem wierności zasadom projektowania zorientowanego na usługi.

Typowa implementacja SOA wymaga na ogół bardziej szczegółowych badań i poszukiwań, w porównaniu ze starszymi paradygmatami platform obliczeniowych. W dużej części to konsekwencja bogactwa technologii usług sieciowych, stanowiących podstawę implementacyjną współczesnej SOA.

Tak, oczywiście, w końcowym rozrachunku SOA spowoduje uproszczenie przetwarzania rozproszonego; stanie się to wówczas, gdy standaryzacja zorientowana na usługi na dobre zagości w środowisku IT, czyli gdy zaistnieje wystarczający zestaw komponowalnych usług, a zasady zorientowania na usługi zostaną w pełni zintegrowane ze wspomnianym środowiskiem.

3.3.5. „Aplikacja z usługami sieciowymi, wykorzystująca rozszerzenia WS-*, jest aplikacją zorientowaną na usługi"

Podczas gdy druga generacja specyfikacji usług sieciowych predestynuje SOA do roli pierwszego skrzypka w orkiestrze infrastruktury IT, to odwrotna relacja niekoniecznie jest prawdziwa — proste uczynienie tych specyfikacji częścią architektury nie sprawi, że ta architektura stanie się zorientowana na usługi. Niezależnie od bogactwa samych usług sieciowych, tym, co czyni te usługi częścią architektury zorientowanej na usługi, jest sposób zaprojektowania tejże architektury samej z siebie.

Z drugiej strony, można jednak żywić nadzieję, że większość rozwiązań poważnie podchodzących do wykorzystania rozszerzeń WS-* to rozwiązania faktycznie zorientowane na usługi. Przesłanką tej nadziei jest prawdopodobieństwo, iż tempo upowszechniania SOA jest z grubsza zbieżne z coraz większym wsparciem dla rozszerzeń WS-* w środowiskach programistycznych i produktach middleware.

3.3.6. „Jeśli znasz usługi sieciowe, bez problemu zbudujesz SOA"

Znajomość tematyki usług sieciowych pod kątem koncepcyjnym i technicznym jest niewątpliwie nieodzowna dla projektantów, jednakże — jak wspominaliśmy na początku tego rozdziału — fundamentalne zasady zorientowania na usługi pozostają bez związku z konkretną technologią. Zorientowanie na usługi koncentruje się raczej na odpowiednim postrzeganiu i partycjonowaniu logiki biznesowej i logiki aplikacji, a w stosunku do usług sieciowych — na wymaganiu, by były projektowane i wykorzystywane zgodnie z fundamentalnymi zasadami.

Usługi sieciowe mogą być z łatwością włączane do istniejących tradycyjnych architektur rozproszonych, w których bądź to zajmą centralne miejsce i obarczone zostaną odpowiedzialnością za krytyczną część przetwarzania, bądź też staną się punktami dostępu do peryferyjnych aplikacji.

SOA wykorzystuje usługi sieciowe w znacząco inny sposób. Nacisk położony na enkapsulację logiki biznesowej i tworzenie warstwy abstrakcji usług wymaga zwykle doświadczenia zarówno w zakresie technologii, jak i analityki biznesowej. Można więc śmiało przyjąć, że realizacja współczesnej SOA wymaga umiejętności wykraczających daleko poza li tylko technologię usług sieciowych.

3.3.7. „Gdy przejdziesz na SOA, wszystko zacznie współdziałać"

Do upowszechnienia tego mitu walnie przyczynił się szum medialny i agresywna reklama. Wielu mniema, że z racji samego zbudowania rozwiązań zorientowanych na usługi infrastruktura techniczna magicznym sposobem sama przekształci się w ujednoliconą, sfederowaną architekturę.

Owszem, może się tak stać, lecz tylko za sprawą inwestycji, analiz, no i przede wszystkim wysokiego stopnia standaryzacji. Wykorzystanie frameworku komunikacyjnego usług sieciowych i architektury zorientowanej na usługi (i różnych architektur integracyjnych zorientowanych na usługi) pozwala na abstrahowanie i skrywanie szczegółów konkretnego rozwiązania, jego platformy i zastosowanej technologii.

Tak właśnie tworzy się przewidywalne medium komunikacyjne dla wszystkich aplikacji wyeksponowanych za pośrednictwem usług sieciowych. Niestety, nie oznacza to automatycznie standaryzacji *informacji* wymienianej za pomocą tego medium. Gdy więc SOA stanie się bardziej powszechna, będzie można spotkać jej znakomite implementacje oraz implementacje zdecydowanie mniej udane. Warunkiem prawdziwie udanego współdziałania, federacyjności, wielokrotnego użycia i innych jeszcze korzyści jest poddanie każdej z usług składowych wspólnym standardom projektowym.

PODSUMOWANIE

- Większość nieporozumień dotyczących znaczenia SOA wynika z polityki marketingowej i nieobiektywnej reklamy medialnej.
- Często usługi sieciowe działające w rozproszonej architekturze internetowej mylone są z prawdziwą, współczesną SOA.
- Najbardziej zdradliwym mitem jest postrzeganie rozwiązań zorientowanych na usługi jako prostych z natury, łatwych w tworzeniu i automatycznie zapewniających współdziałanie.

3.4. Najważniejsze wymierne pożytki z SOA

Dotychczas zajmowaliśmy się komponentami tworzącymi SOA i tematykę tę będziemy szczegółowo rozwijać dalej w tej książce, zajmując się wewnętrznymi szczegółami funkcjonowania rozwiązań zorientowanych na usługi. Nie można jednak nie zadać sobie oczywistego pytania — po co to wszystko? Z jakich to powodów gremia informatyków zadają sobie trud tak znaczącej metamorfozy zarówno filozofii, jak i technologii, by zaadaptować SOA na gruncie dotychczasowej infrastruktury, jako docelowy następnik tejże?

Skupimy się w tym miejscu na uchwytnych, wymiernych zyskach ze wspomnianej inwestycji, związanych głównie z:

- usprawnieniami, jakie SOA zdolna jest wprowadzać do konstruowania rozwiązań automatyzujących biznes,
- korzyściami, jakie dla globalnie postrzeganego przedsiębiorstwa niesie proliferacja orientacji na usługi.

UWAGA

Korzyści, jakie odnosić może firma dzięki SOA, mogą być wielorakie i rozmaite, zależnie od wytyczonych celów oraz sposobu, w jaki wdrażane są komponenty SOA, wraz z całą armią produktów powiązanych. Przedstawiana w tym miejscu lista ma charakter raczej ogólny i jako taka daleka jest — oczywiście — od kompletności, stanowi jedynie pewien wyznacznik potencjału, jaki kryją w sobie platformy architektoniczne SOA.

3.4.1. Usprawniona integracja (i naturalne współdziałanie)

SOA umożliwia tworzenie rozwiązań składających się z usług naturalnie współdziałających ze sobą. Wykorzystywanie rozwiązań bazujących na współdziałających usługach jest częścią **integrowania zorientowanego na usługi** (SOI — *Service-Oriented Integration*), prowadzącego do stworzenia **architektury integracyjnej zorientowanej na usługi**.

Dzięki niezależności frameworku komunikacyjnego od konkretnych dostawców — niezależności wynikającej z zasad SOA implementowanej przez usługi sieciowe — możliwe jest implementowanie wysoce zestandaryzowanych opisów usług i struktur komunikatów. W rezultacie otrzymujemy naturalne współdziałanie platform, a wtedy projektanci mogą skupić się raczej na modelowaniu procesów biznesowych niż konstruowaniu specyficznych pomostów komunikacyjnych między aplikacjami.

Zatem koszt i wysiłek integracji międzyaplikacyjnej stają się znacząco mniejsze, gdy integracja ta zgodna jest z zasadami SOA.

3.4.2. Wbudowana wielokrotna używalność

Orientacja na usługi propaguje projektowanie usług w sposób zapewniający ich wielokrotną używalność. Projektowanie w ten sposób otwiera prostą drogę do lepszego wykorzystywania istniejących implementacji logiki automatyzacyjnej.

Tworzenie rozwiązań składających się z usług, które, oprócz wymagań na szczeblu aplikacji, mogą być również wykorzystywane przez potencjalnych przyszłych wnioskodawców, zwiększa efektywność projektowania i implementowania, bo raz zainwestowany wysiłek może być wykorzystywany wielokrotnie.

Zatem, mimo umiarkowanie większego wysiłku koniecznego do zapewnienia wielokrotnej używalności, zyskujemy znacząco większy stopień spożytkowania owoców tego wysiłku.

3.4.3. Eleganckie architektury i rozwiązania

Jedną z fundamentalnych koncepcji SOA jest komponowalność. Wbrew pozorom, koncepcja ta wykracza daleko poza proste agregowanie usług w ramach kolekcji. Cała platforma WS-* zbudowana jest całkowicie na bazie komponowalności.

Jak pisaliśmy już w punkcie 3.2.9, ten aspekt architektury zorientowanej na usługi może prowadzić do powstawania wysoce zoptymalizowanych środowisk automatyzacyjnych, zawierających wyłącznie komponenty faktycznie niezbędne.

Zatem osiągnięcie opisanej optymalizacji wymaga zgodności ze standardami projektowymi, rządzącymi dozwolonymi rozszerzeniami w każdym środowisku aplikacyjnym. Ograniczenie wykorzystywanych rozszerzeń do niezbędnego zestawu przyczynia się do zredukowania zarówno wysiłku projektantów, jak i zakresu ich niezbędnych kwalifikacji (bo te ograniczają się do wiedzy związanej z konkretną aplikacją, usługą lub rozszerzeniem), przyczynia się również do poprawy ogólnej wydajności.

UWAGA

Wspomniana powyżej poprawa wydajności odnosi się wyłącznie do eliminacji realizowania zbędnych rozszerzeń (nieobecnych w zoptymalizowanej implementacji), bo wymiana danych oparta na komunikatach SOA charakteryzuje się większym obciążeniem przetwarzania w porównaniu ze zdalnym wywoływaniem procedur (RPC — *Remote Procedure Call*) stosowanym w tradycyjnych architekturach rozproszonych. (O zagadnieniu wydajności na platformach SOA piszemy w punkcie 3.5.5).

3.4.4. Spożytkowanie dawnych inwestycji

Upowszechnienie się technologii usług sieciowych w przemyśle informatycznym stało się motorem napędowym na rynku różnorodnych adapterów, umożliwiających dostosowywanie wielu starszych aplikacji do uczestnictwa w architekturach integracyjnych zorientowanych na usługi. Tradycyjne środowiska, działające dotychczas w izolacji od siebie, mogą stawać się dzięki temu komponentami wspólnej architektury, zapewniającej w jednolity sposób współdziałanie i uwalniającej od konieczności opracowywania kosztownych i często niepewnych kanałów integracyjnych typu punkt-punkt. I nawet mimo faktu, że wspomniane adaptery nie rozwiązują wielu istotnych problemów — jak chociażby problem sprostania przez tradycyjną aplikację znacznie większemu

strumieniowi danych w porównaniu z tym, dla którego została oryginalnie zaprojektowana — to i tak czerpanie wciąż korzyści ze starszych rozwiązań w nowych warunkach orientacji na usługi jest samo z siebie perspektywą na wskroś atrakcyjną.

Zatem koszt i wysiłek integracji starszych rozwiązań z nowymi stają się znacznie mniejsze, a konieczność rezygnacji ze starszych aplikacji — znacznie mniej prawdopodobna.

3.4.5. Jednolite reprezentowanie danych w standardzie XML

Język XML to jeden z fundamentów SOA, więc transformacja w stronę SOA stanowi doskonałą okazję do wykorzystania wszelkich zalet reprezentacji danych w tym standardzie. Zestandaryzowany format reprezentacji danych — gdy zostanie opracowany i pomyślnie wdrożony — oznacza na ogół zmniejszenie złożoności całego środowiska aplikacyjnego. Oto szczegóły.

- Dokumenty w formie języka XML (i towarzyszące im schematy) upakowane w komunikatach SOAP, wymienianych między aplikacjami lub ich komponentami, powodują ujednolicenie formatu wymienianych danych, co umożliwia budowanie przewidywalnych, a w konsekwencji łatwo rozszerzalnych i adaptowalnych sieci komunikacyjnych.
- Samoopisująca natura języka XML ułatwia interpretowanie zakodowanych w tym języku danych przez architektów, analityków i deweloperów. Dzięki temu dane przenoszone przez komunikaty stają się prostsze koncepcyjnie, bardziej zrozumiałe i łatwiejsze w śledzeniu.
- Standaryzacja formy wymienianych danych stwarza grunt dla wewnętrznego współdziałania. Przykładowo wykorzystywanie wspólnych słowników powoduje redukcję konieczności niwelowania różnic w interpretowaniu korporacyjnych modeli danych przez różne aplikacje.

Adaptacja starszych platform do standardu XML wykonywana była z reguły wybiórczo, stosownie do bieżących potrzeb. Jej korzyści dla organizacji były więc mocno ograniczone w stosunku do tych, jakie potencjalnie przynieść mogłoby systematyczne przejście na XML. Ze względu na fundamentalne znaczenie XML dla współczesnej SOA, taka systematyczna adaptacja jest nie opcją, lecz koniecznością — i jednocześnie okazją do standaryzacji w znacznie szerszym zakresie.

Zatem proliferacja standardowej reprezentacji danych w języku XML przyczynia się do redukcji kosztów i wysiłków związanych z tworzeniem aplikacji.

UWAGA

Korzyści wspomniane w dwóch ostatnich akapitach (integrowanie starszych aplikacji i reprezentowanie danych w języku XML) opisywane są obszerniej w książce *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services.*

3.4.6. Skoncentrowane inwestowanie w infrastrukturę komunikacyjną

Ponieważ usługi sieciowe ustanawiają wspólny framework komunikacyjny, komunikacje wewnątrzaplikacyjna i międzyaplikacyjna stają się centralnym elementem standardowej infrastruktury IT. Ujednolicenie tej komunikacji oznacza, że przeznaczone na nią inwestycje można skoncentrować na pojedynczej technologii.

Zatem ograniczenie się do pojedynczej technologii realizującej komunikację w ramach sfederowanej części infrastruktury IT przedsiębiorstwa oznacza redukcję kosztów skalowania tej komunikacji.

3.4.7. Wybór najlepszego wariantu

Częstą przyczyną ataków, jakich obiektem staje się dział IT firmy, są ograniczenia określonej platformy technologicznej w zakresie możliwości realizacji wymagań dotyczących automatyzacji biznesu w tejże firmie. Ograniczenia te mogą wynikać z niedopuszczalnie wysokich kosztów przedsięwzięcia, lecz równie dobrze mogą być pochodną wrodzonych ograniczeń samej technologii. W efekcie dział IT zmuszony zostaje do moderowania swych planów w zakresie poszerzania lub wymiany istniejących rozwiązań, a często nawet do rezygnacji z tych planów.

Oczywiście, SOA nie stanowi panaceum na problemy tego rodzaju, ale może je wydatnie łagodzić za sprawą jednej ze swych fundamentalnych cech — niezależności od konkretnej technologii. Jedynym żądaniem pod adresem określonego „kawałka" automatyzacji jest wyeksponowanie adekwatnego interfejsu usługi; poszukując technologii do realizacji tejże usługi, jesteśmy uniezależnieni od konkretnej platformy czy oprogramowania middleware i mamy swobodę wyboru tej, która najlepiej odpowiada określonym potrzebom i możliwościom.

Zatem niezależność technologiczna zwiększa możliwości w zakresie automatyzacji biznesu oraz generalnie oznacza lepszą jakość tworzonych rozwiązań.

3.4.8. Zwinność organizacyjna

„Zwinność" to cecha, którą odnosić można niemal do wszystkiego: zwykły algorytm, komponent oprogramowania, rozwiązanie, platforma, proces — miarą zwinności każdego z tych elementów jest sposób jego skonstruowania, umiejscowienia i użyteczności. To, jak całość tych elementów wpisuje się w ogół finansowych i kulturowych uwarunkowań firmy i jak dzięki tym elementom usprawniona zostaje realizacja procesów biznesowych, stanowi miarę zwinności firmy jako całości.

Znakomita większość rozwiązań zorientowanych na usługi konstruowana jest ze świadomością faktu, iż to, co zostało dzisiaj zbudowane, z biegiem czasu będzie musiało być ewolucyjnie dostosowywane do nieuchronnych zmian w biznesie. Jednym z pożytków, jakie daje dobrze zaprojektowana SOA, jest właśnie złagodzenie związanych z tym uciążliwości. Jeśli zdolność reagowania na zmiany staje się normą w projektowaniu rozwiązania rozproszonego, zalety, takie jak wielokrotna używalność i współdziałanie, stają się czymś naturalnym. Przewidywalność, jaką niosą ze sobą te cechy, zapewnia należyty poziom zwinności organizacyjnej. Wszystko to można jednak osiągnąć tylko pod warunkiem właściwego projektowania i standaryzacji.

W mało elastycznych środowiskach IT zmiany są prawdziwą plagą: mogą być destrukcyjne, kosztowne i generalnie niebezpieczne. Zdolność antycypowania zmian przez tworzone rozwiązania automatyzacyjne (wraz ze wspomagającą infrastrukturą) jest więc zaletą trudną do przecenienia. Zestandaryzowane środowisko składające się z luźno powiązanych, komponowalnych, współdziałających i potencjalnie wielokrotnie używalnych usług niezaprzeczalnie zdolność taką posiada.

Co więcej, poprzez tworzenie specjalizowanych warstw usług, abstrahujących logikę biznesową i technologię implementacyjną, SOA ustanawia luźne powiązanie między tymi dwiema domenami. Każda z nich może więc rozwijać się względnie niezależnie i elastycznie reagować na zmiany wymuszane przez drugą domenę. A to w znaczącym stopniu przyczynia się do zwinności organizacyjnej, niezależnie od tego, jak intensywnych zmian wymagać będą nowe uwarunkowania.

Zatem SOA przyczynia się do redukcji kosztu i wysiłku związanego z adaptacją infrastruktury IT do zmian zachodzących w procesach biznesowych lub technologiach implementacyjnych.

PODSUMOWANIE

- Inwestowanie w SOA może przynosić wymierne zyski wyrażające się w konkretnych kategoriach.
- Wiele długofalowych korzyści obiecywanych przez SOA uwidacznia się dopiero wówczas, gdy zasady orientowania na usługi zostaną wdrożone w skali całego przedsiębiorstwa.
- Na szczęście, SOA umożliwia także osiągnięcie wielu korzyści o charakterze doraźnym.

3.5. Najważniejsze pułapki w adaptacji SOA

Skoro omówiliśmy potencjalne korzyści płynące ze współczesnej SOA, czyli to, co „ewentualnie może być", pomyślmy teraz kategoriami „jak jest naprawdę", czyli przyjrzyjmy się realiom.

Generalnie każdy projekt, architektura, aplikacja itd. może być produktem różnej jakości i SOA nie jest w tym względzie wyjątkiem. Zważywszy na niebanalny rozmiar działań (również ich wymiar czasowy) koniecznych do pełnego zaadaptowania SOA w przedsiębiorstwie, a przede wszystkim na fakt, że transformacja taka wymaga zmiany sposobu myślenia na temat automatyzacji biznesu, można żywić obawy, iż transformacja ta nie zawsze przebiegać będzie we właściwym kierunku. I rzeczywiście; poniżej przedstawiany krótkie omówienie kilku wybranych, najczęściej popełnianych błędów tego rodzaju.

3.5.1. Budowanie architektury zorientowanej na usługi na wzór architektury tradycyjnej

Prawdopodobnie błędem numer jeden w dążeniu do celów spodziewanych po SOA jest budowanie lub rozbudowywanie tradycyjnej architektury rozproszonej w przekonaniu, że oto buduje się współczesną SOA. To złudne przekonanie jest najczęściej wynikiem przyjęcia któregoś z błędnych założeń, o których dopiero co pisaliśmy.

Niby nic wielkiego, wszyscy popełniamy błędy; niebezpieczeństwo kryjące się za *tym błędem* jest jednak na tyle poważne, że firma może zabrnąć dość daleko w ślepą uliczkę (ze wszystkimi, nie tylko finansowymi, tego konsekwencjami), zanim zorientuje się, że podąża w ewidentnie złym kierunku i trzeba zawracać. Oto kilka typowych praktyk, które mogą zwiastować opisane nieszczęście.

- Konstruowanie opisów usług w stylu typowym dla RPC, z konsekwencją w postaci zbyt intensywnej wymiany „drobnoziarnistych" komunikatów.
- Unikanie wykorzystywania cech oferowanych przez specyfikacje WS-*.
- Niewłaściwe partycjonowanie elementów funkcjonalnych pomiędzy usługi.
- Tworzenie usług niekomponowalnych (lub komponowalnych połowicznie).
- Uporczywe trzymanie się wzorców komunikacji synchronicznej.
- Tworzenie usług hybrydowych lub niestandardowych.

Kluczem do unikania opisanych sytuacji jest zrozumienie fundamentalnych różnic między SOA a wcześniejszymi architekturami (tematyce tej poświęcona jest część rozdziału 4.).

3.5.2. Brak standaryzacji

W większych firmach, w których realizuje się równolegle kilka projektów, zdefiniowanie odpowiednich standardów jest zadaniem pierwszorzędnym. Integracja (w przyszłości) aplikacji utworzonych w ramach dwóch różnych projektów, rządzących się odmiennymi standardami, będzie prawdopodobnie bardzo trudna, kosztowna, a jej rezultat wątpliwy. Lekcję tę przerabiało już wiele działów IT, stających przed zadaniem integrowania „spadkowych" aplikacji.

SOA umożliwia federowanie zróżnicowanych środowisk — to jedna z jej fundamentalnych zasad — lecz nie dokona się ono „z automatu", za sprawą zakupienia jedynie najnowszych aktualizacji narzędzi deweloperskich czy oprogramowania serwerowego. SOA, podobnie zresztą jak wiele innych architektur, ujawnia swój potencjał dopiero w połączeniu z opracowaniem i wdrożeniem odpowiedniej standaryzacji (pisaliśmy już o tym w punkcie 3.3.7).

Realizowanie projektu rozwiązania zorientowanego na usługi w oderwaniu od innych projektów i aplikacji może doprowadzić do sytuacji, w której próba integrowania tegoż rozwiązania z innymi, działającymi w tej samej firmie, przysporzy wiele dotkliwych problemów, z których najbardziej dotkliwe to:

- niekompatybilność w reprezentacji danych — ten sam typ informacji reprezentowany będzie według różnych schematów;
- nieregularna struktura i zróżnicowanie semantyki opisów usług;
- niespójność implementacyjna, czyli wykorzystywanie (do realizacji tego samego zadania) różnych rozszerzeń lub rozszerzeń implementowanych w odmienny sposób.

SOA w naturalny sposób propaguje oddzielenie interfejsu od implementacji, czyli ukrycie szczegółów funkcjonowania zaplecza. Nie zmienia to konieczności standaryzacji projektu i interakcji usług kolektywnie realizujących logikę tego zaplecza. Różne standardy projektowe, takie jak podejście „najpierw WSDL" omawiane w częściach IV i V tej książki, stworzone zostały z myślą o osiągnięciu wielu kluczowych korzyści oferowanych potencjalnie przez SOA.

3.5.3. Brak planu transformacji

Szanse na udaną migrację w stronę SOA stają się znacząco mniejsze, gdy migracja ta nie jest realizowana według jasnego, zrozumiałego planu. Ponieważ sposób umiejscowienia usług w infrastrukturze IT może oznaczać kompletną redefinicję tejże struktury, reperkusje źle przeprowadzonej migracji mogą być dla firmy katastrofalne.

Wspomniany plan umożliwia skoordynowanie poszczególnych faz migracji, przeprowadzanej równolegle na trzech poziomach — technologicznym, architektonicznym i organizacyjnym.

Szczegółowa treść takiego planu jest specyficzna dla konkretnej firmy, choć — ogólnie rzecz biorąc — powinna zawsze zawierać pewne kluczowe elementy, między innymi:

- analizę przewidywanego wpływu przeprowadzanej transformacji na istniejące zasoby, procesy, wewnętrzne standardy i technologie;
- definicję architektury transformacyjnej, w ramach której środowiska przewidziane do transformacji ewoluują poprzez serię zaplanowanych, hybrydowych etapów;

- spekulatywną analizę możliwości wykorzystania przyszłego rozwoju usług sieciowych i technologii wspierających te usługi;
- wykaz szczegółowych zmian w scentralizowanej logice (związanych na przykład z nowym modelem bezpieczeństwa).

Opracowanie (i przestrzeganie) planu transformacji pozwala uniknąć wielu nieprzyjemnych niespodzianek, jakie mogą przytrafić się w konsekwencji doraźnych zabiegów, wykonywanych w ramach transformacji ad hoc. Jak wspominaliśmy, szczegółowa postać planu transformacji jest specyficzna dla konkretnej firmy — dla jej oczekiwań, ograniczeń i celów.

3.5.4. Początki bez XML

W świecie współczesnej SOA wszystko zaczyna się od usług sieciowych — to zdanie, powtarzane niby mantra, jest jednak tylko połowicznie prawdziwe. W świecie współczesnego SOA wszystko faktycznie bierze swój początek od standardu XML. Na bazie tego standardu zdefiniowano wiele standardów pochodnych, składających się na architekturę de facto reprezentowania danych. Właśnie te standardy napędzały tworzenie wielu specyfikacji usług sieciowych — specyfikacji stanowiących motor napędowy współczesnej SOA.

Obecnie tak wiele uwagi poświęca się transferowi danych między usługami, że praktycznie zaniedbywana jest kwestia strukturalizacji i walidacji tych danych przez komunikujące się usługi. Może to odbijać się negatywnie na jakości (wydajności) przetwarzania danych: przykładowo te same dane mogą być wielokrotnie poddawane walidacji według tych samych kryteriów bądź też te same dane mogą być niepotrzebnie przetwarzane dwukrotnie — przed wysłaniem przez jedną usługę i po odebraniu ich przez drugą. I to wszystko niejako przypadkowo, bez wyraźnych intencji projektantów i programistów.

Standaryzacja sposobu, w jaki kluczowe technologie XML wykorzystywane są do reprezentowania, walidacji i przetwarzania danych korporacyjnych przesyłanych w środowisku aplikacji (zarówno wewnątrz usług, jak i między nimi), jest warunkiem niezbędnym do skonstruowania solidnej, zoptymalizowanej i interoperacyjnej SOA.

3.5.5. Ignorowanie wymagań wydajnościowych SOA

W niewielkiej skali łatwo tworzyć rozwiązania zorientowane usługowo, prawidłowo funkcjonujące i dobrze reagujące na komunikaty z otoczenia. Gdy rozrasta się środowisko i dodawana jest nowa funkcjonalność, w konsekwencji rośnie natężenie wymiany komunikatów. W nieprzygotowanych na to aplikacjach obserwuje się wówczas drastyczne spowolnienie przetwarzania danych.

Ponieważ współczesna SOA opiera się na tworzeniu abstrakcji przetwarzania danych, pojawia się problem dodatkowego obciążenia narzucanego przez te warstwy. Uzależnienie współczesnej SOA od usług sieciowych pogłębia jej zależność od reprezentacji danych w języku XML, co w naturalny sposób zwiększa wymagania dotyczące przetwarzania tej reprezentacji. Przykładowo mechanizmy bezpieczeństwa usług sieciowych — między innymi szyfrowanie i podpisy cyfrowe — skutkują pojawieniem się nowych warstw przetwarzania, po stronie zarówno nadawców, jak i odbiorców komunikatów.

Dla pomyślnego budowania rozwiązań opartych na usługach konieczne jest więc zawczasu zarówno zrozumienie i oszacowanie opisanych wymagań, jak i rozpoznanie ograniczeń własnej struktury w tym zakresie. W praktyce oznacza to:

- przetestowanie zdolności wspomnianej infrastruktury do przetwarzania komunikatów, jeszcze przed zainwestowaniem w usługi sieciowe;
- przetestowanie, w warunkach ekstremalnego obciążenia, różnorodnych procesorów (dedykowanych standardom XML, XSLT, SOAP itp.), których zastosowanie bierze się pod uwagę;
- przeprowadzenie rozpoznania w obszarze alternatywnych procesorów, akceleratorów i innych technologii wspomagających; w tej grupie wymienić można między innymi specyfikacje XOP (*XML-binary Optimized Packaging*) i SOAP MTOM (*Message Transmission Optimization Mechanism*), obie autorstwa W3C (szczegółowe informacje na ich temat znaleźć można w witrynie *www.w3c.org*).

Wydajność jest też jednym z kryteriów preferowania „gruboziarnistych" interfejsów usług oraz asynchronicznej komunikacji między usługami. Wraz z innymi jeszcze zabiegami projektowymi cechy te pozwalają na unikanie potencjalnych „wąskich gardeł" przetwarzania.

3.5.6. Niedocenianie bezpieczeństwa usług sieciowych

Zakres rozwoju technologii usług sieciowych w danym środowisku jest zwykle powiązany ze swobodą deweloperów i architektów w ramach danego frameworku technologicznego. W rozrastającej się architekturze wygodnie jest nadal wykorzystywać uproszczony mechanizm wymiany komunikatów, którego zabezpieczenie opiera się na szyfrowaniu dostarczanym przez protokół SSL (*Secure Sockets Layer*).

Mimo iż SSL świetnie się spisuje w roli zabezpieczenia bezpośredniego, nie jest technologią „z wyboru" dla SOA: w miarę jak usługi obejmować będą coraz większy zakres przetwarzania, pojawi się zapotrzebowanie na mechanizmy zabezpieczające na poziomie komunikatów. Framework WS-Security oferuje powszechnie akceptowalny model zabezpieczeń, realizowany przez rodzinę specyfikacji przenikających aplikacje usługowe i architektury korporacyjne na wielu poziomach.

Jednym z ważniejszych zagadnień projektowych związanych z wykorzystywaniem modelu WS-Security jest zapewnienie centralizacji zabezpieczeń. Zgodnie z takim podejściem, duża porcja logiki i zasad bezpieczeństwa wydzielona zostaje w postaci odrębnej, centralnej warstwy abstrakcji, na bazie której zapewnione zostaje bezpieczeństwo dla aplikacji usługowych.

Jeśli nawet wykorzystywana obecnie platforma nie zapewnia wystarczającego wsparcia dla WS-Security, a istniejące zabezpieczenia oparte na SSL spełniają bieżące wymagania w zakresie bezpieczeństwa, warto pomyśleć perspektywicznie, zwracając baczną uwagę na zmiany, jakie nieuchronnie prędzej czy później okażą się konieczne. Rozwijanie infrastruktury bez wykorzystywania WS-Security nieuchronnie prowadzi do sytuacji, w której niezbędne stanie się ponowne projektowanie i implementowanie niektórych rozwiązań. Argument ten staje się ważniejszy, gdy decydujemy się na zaimplementowanie scentralizowanego modelu bezpieczeństwa, który stanie się rozszerzeniem infrastruktury IT.

3.5.7. Niedotrzymywanie kroku nowoczesnym platformom i standardom

Profesjonaliści IT, pracujący w ramach jednej platformy programistycznej, nie mają na ogół w zwyczaju interesować się bieżącymi trendami objawiającymi się w świecie innej platformy — i tak na przykład twórcy aplikacji dla platformy .NET nie są generalnie zbyt podekscytowani tym, co aktualnie dzieje się w kręgach Javy (i vice versa).

Niezależność SOA od konkretnych technologii otwiera przed działami IT szeroki rynek produktów oraz możliwość wyboru platformy do tworzenia i hostowania logiki aplikacji. I chociaż względy produktywności przemawiają za kontynuacją wykorzystywania tego, co zna się najlepiej (bo poznało się i praktykowało przez lata), to jednak opcja sięgnięcia po nowsze narzędzia istnieć będzie zawsze, między innymi również dlatego, że otwarty framework komunikacyjny (jako jeden z fundamentów SOA) gwarantować będzie możliwość komunikacji między aplikacjami zbudowanymi za pomocą różnych technologii — i w konsekwencji współdziałanie tychże aplikacji.

Jednym z istotnych czynników, które należy brać pod uwagę przy wyborze konkretnego dostawcy, jest relacja tego dostawcy z trwającym procesem opracowywania specyfikacji rozszerzeń WS-*, zwłaszcza w sytuacji, gdy określone rozszerzenie ma zostać użyte do implementacji kluczowych elementów logiki aplikacji. Interesująca będzie z pewnością pozycja dostawcy wśród dostawców konkurencyjnych, dostarczających implementacje tych samych rozszerzeń. Równie interesujący może być ogólny związek konkretnego dostawcy ze światem specyfikacji WS-*, na przykład w kwestii tego:

- które specyfikacje znajdują wsparcie w jego produktach,
- jakie jest jego zaangażowanie w sam proces opracowywania specyfikacji,
- z którymi innymi organizacjami współpracuje podczas doskonalenia określonego standardu,
- jak przedstawia się planowane wsparcie, ze strony produktów lub platform, dla specyfikacji i standardów, które opublikowane zostaną w najbliższej przyszłości.

W rozdziale 4. podajemy ogólny opis procesu opracowywania standardów oraz charakterystykę pierwszych organizacji standaryzacyjnych związanych z SOA.

PODSUMOWANIE

- Wiele błędów popełnianych przy transformacji w stronę SOA wynika z niezrozumienia jej istoty oraz nieświadomości wymagań warunkujących pomyślne jej wdrożenie.
- Zrozumiały plan transformacji jest najlepszą bronią przeciwko przeszkodom, które napotykać mogą firmy na swej drodze migracji w kierunku SOA.
- Dotrzymywanie kroku tworzonym produktom komercyjnym oraz opracowywanym standardom stanowi istotny czynnik zapewniający zgodność tworzonych obecnie rozwiązań z przyszłym rozwojem infrastruktury.

Skorowidz

C

D

E

F

G

I

J

K

L

Ł

M

N

O

P

Q

R

S

Ś

T

U

W

X

Z

Ź

Ż

Przekonaj się, jak SOA może zmienić Twój świat!

Czasy, kiedy użytkownicy otrzymywali zamknięte produkty, zapakowane w pudełko z nośnikiem i instrukcją, przemijają. Świat IT zmierza w kierunku SOA (ang. Service-Oriented Architecture). Producenci chcą nam sprzedawać wiele usług, które potrafią komunikować się między sobą i dostarczać użytkownikom wymierne korzyści. To podejście pozwala lepiej zarządzać procesem wytwarzania usługi, łatwiej wprowadzać zmiany i aktualizacje oraz elastyczniej rozliczać się z klientami. Od tej drogi nie ma już odwrotu — SOA to przyszłość branży IT!

Ta książka to świetne źródło informacji na ten temat. W trakcie lektury poznasz podstawy i założenia tej architektury oraz dowiesz się, jakie narzędzia wykorzystać, żeby wytworzyć produkt spełniający zasady SOA. Książka ta jest bogata w analizy przypadków oraz przykłady z życia, które pozwolą Ci się przekonać, jak bardzo architektura ukierunkowana na usługi jest przydatna przy rozwiązywaniu codziennych problemów. SOA jest tworem żywym, który cały czas ewoluuje — wymaga to kontroli organizacji oraz ustalenia pewnych standardów, czyli określenia, jak takie usługi mają wyglądać. W trakcie lektury dowiesz się, kto sprawuje nad tym pieczę oraz jakie są kierunki rozwoju SOA. Książka ta będzie biblią każdego projektanta i dewelopera usług sieciowych. Nie możesz minąć jej obojętnie!

Dzięki tej książce dowiesz się:

- jak SOA zmienia świat IT
- dlaczego warto wykorzystać SOA w Twoim projekcie
- jak projektować usługi sieciowe
- jak zapewnić bezpieczeństwo usługom w SOA

ISBN 978-83-246-7493-0

cena: 99,00 zł