MECHANIZMY INTEGRACJI DANYCH I LOGIKI BIZNESOWEJ W KOOPERACYJNYCH SYSTEMACH STEROWANIA PROCESAMI PRZEDSIĘBIORSTWA

Paweł BUCHWALD

Streszczenie: Artykuł dotyczy wybranych problemów implementacji szkieletowego systemu EPC, będącego uniwersalnym modelem referencyjnym zintegrowanych systemów zarządzania i sterowania procesami biznesowymi. Przedstawiono w nim metody integracji danych i logiki biznesowej używane w rozproszonych systemach informatycznych, które mogą być pomocne przy implementacji rozbudowanych systemów zarządzania zawierających szkieletowy system EPC. Pokazano, jak wyodrębniać oraz identyfikować zewnętrzne usługi i dane w celu ich powiązania z elementami szkieletowego systemu EPC. Odpowiednie mechanizmy inżynierii oprogramowania uwzględniają wielowarstwową architekturę i zastosowanie relacyjnych baz danych we współczesnych systemach zarządzania.

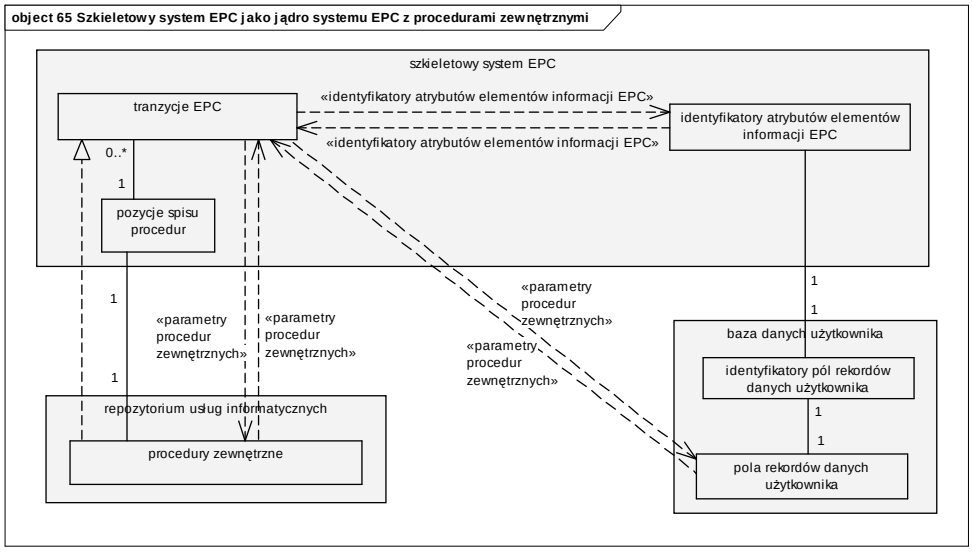
Słowa kluczowe: zintegrowane systemy zarządzania, bazy danych, integracja danych biznesowych, ETL, SOA, Web Services, Service Broker, SQL Server, Oracle.

1. Kooperacyjne systemy EPC

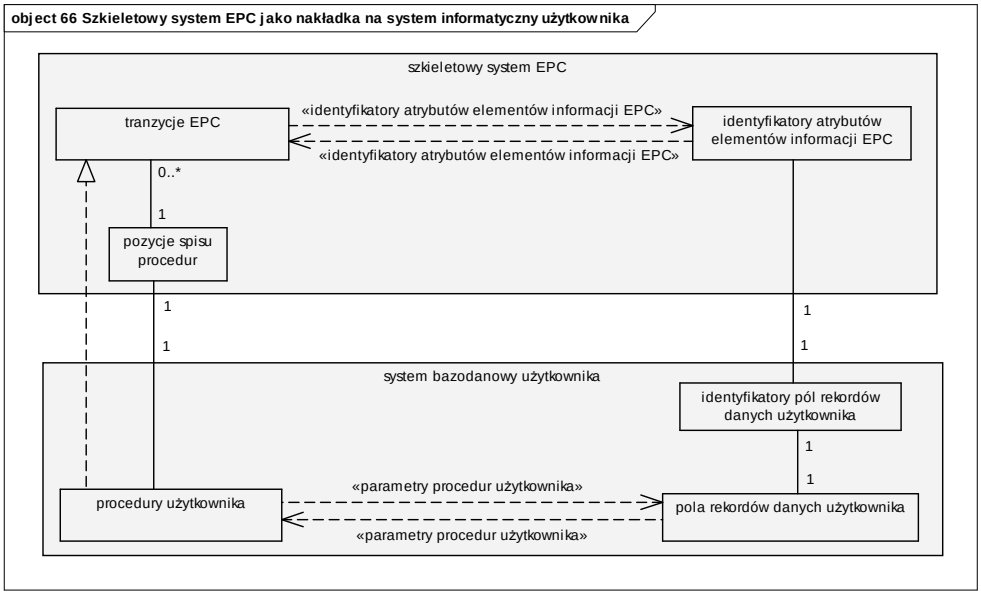
Rozwój informatycznych systemów zarządzania w ostatnich 30 latach opierał się w dużym stopniu na ciągle doskonalonych modelach referencyjnych [6], które coraz lepiej tłumaczą strukturę procesów biznesowych będących obiektami zarządzania [3, 8]. W ostatnich latach podjęto prace nad szkieletowym systemem sterowania procesami przedsiębiorstwa, w skrócie EPC (Enterprise Process Control) [10, 11, 12, 13]. Jest to uniwersalny model referencyjny systemów zarządzania i sterowania procesami biznesowymi, który opisuje nie tylko procesy, zasoby i organizację przedsiębiorstw, lecz także typową strukturę zintegrowanego systemu zarządzania i sterowania procesami od poziomu zarządzania nadzorczego do poziomu sterowania procesami wewnątrz stanowisk roboczych. Szkieletowy system EPC zawiera ponadto szkielet systemu informatycznego o strukturze zgodnej ze strukturą modelu referencyjnego przedsiębiorstwa. Z punktu widzenia inżynierii oprogramowania jest to pewna sieć tranzycji i rozdzielających je miejsc informacji, w czym przypomina ona sieci Petriego [5]. Tranzycje są tu umiejscowieniami procedur przetwarzania danych, a miejsca są rozłącznymi podzbiorami zbioru wszystkich wierszy wszystkich tabel relacyjnej bazy danych (rys. 1, rys. 2).

Współczesne zintegrowane systemy zarządzania klasy ERP (Enterprise Resource Planning) [6] są rozbudowanymi, a przez to kosztownymi systemami, będącymi wynikiem wieloletniej pracy wielu projektantów i programistów. Dlatego można przewidywać, nie wchodząc w dyskusję przydatności, poprawności i ogólności szkieletowego systemu EPC jako modelu systemów zarządzania, że ewentualne pierwsze wdrożenia nie będą systemami zbudowanymi w całości od nowa. Będą to raczej systemy kooperacyjne [13], w których szkieletowy system EPC (bez procedur przetwarzania danych i bez atrybutów niekluczowych w tabelach bazy danych) będzie współpracował z istniejącą bazą danych

przedsiębiorstwa, a procedury będą wywoływane albo z istniejącego systemu bazodanowego (rys. 2), albo z repozytorium usług SOA [4] (rys.1).



Rys. 1. Przepływ informacji między szkieletowym systemem EPC, repozytorium usług informatycznych i bazą danych użytkownika [13]



Rys. 2. Przepływ informacji między szkieletowym systemem EPC i istniejącym systemem informatycznym użytkownika [13]

W obu przypadkach trzeba zapewnić podtrzymywanie związków 1 do 1 między numerami procedur tranzycyjnych w systemie EPC i adresami procedur w systemie użytkownika, albo w repozytorium usług SOA oraz między identyfikatorami atrybutów elementów informacji systemu EPC i identyfikatorami pól rekordów w bazie danych użytkownika. W tym zastosowaniu identyfikatorami atrybutów elementów informacji są trójki (numer tabeli rodzaju informacji, numer elementu informacji danego rodzaju – czyli numer wiersza w tej tabeli, numer atrybutu tego elementu informacji). Trzeba więc pokazać, że współcześnie dostępne technologie informatyczne umożliwiają określenie jednoznacznego adresu każdej procedury i jednoznacznego identyfikatora każdego pola w każdym wierszu każdej tabeli relacyjnej bazy danych. Problem ten, jak również inne problemy integracji danych i logiki biznesowej, które mogą wystąpić w kooperacyjnych systemach EPC, są przedmiotem dalszej części artykułu.

2. Mechanizmy identyfikacji danych i procedur w warstwie bazy danych

Zastosowanie systemów zarządzania relacyjnymi bazami danych jako mechanizmów przechowywania, udostępniania i aktualizacji danych pozwala na stosunkowo łatwą identyfikację i udostępnianie informacji o strukturze danych projektowanej aplikacji. Dotyczy to również rzeczywistych systemów zarządzania, które wykorzystują relacyjne bazy danych. Systemy zarządzania relacyjnymi bazami danych, oprócz udostępnianych schematów danych implementowanych na potrzeby projektowanych rozwiązań, mają szereg wbudowanych obiektów systemowych. Korzystając z tych systemowych obiektów można wyselekcjonować szereg danych umożliwiających identyfikację poszczególnych elementów bazy danych. Przez element bazy danych należy tu rozumieć każdy abstrakcyjny byt podlegający zarządzaniu w relacyjnej bazie danych, a więc tabele, kolumny, widoki, indeksy, procedury wbudowane, ale również pewne elementy, które dotychczas były utożsamiane z warstwą logiki biznesowej, a od jakiegoś czasu mogą być implementowane jako integralne części bazy danych. Przykładem takiego obiektu jest dostępny w SQL Server assembly [2], będący modułem umożliwiającym implementacje funkcji lub procedury w jednym z kilku języków programowania udostępnianych na platformie .NET. Za pomocą jednego z tych języków można zrealizować bardziej elastyczny dostęp do zasobów systemowych, korzystać z wydajnych mechanizmów obliczeniowych udostępnianych przez wspólną przestrzeń typów platformy .NET oraz wspólne środowisko uruchomieniowe, jak również korzystać z paradygmatów projektowania obiektowego i wydajnych narzędzi deweloperskich [9]. Zapewnia to bogatsze możliwości integracji z rozmaitymi systemami zarządzania i przetwarzania danych, również w warstwie szeroko pojętych rozwiązań manipulacji danymi. Wykorzystując te mechanizmy można zaimplementować funkcje wywoływane przez tranzycje systemu EPC w taki sposób, aby pozwalały one wywoływać logikę biznesową z zewnętrznych systemów informatycznych (rys. 2).

Wspomniane moduły assembly, które w bazach danych są stosunkowo nowymi rozwiązaniami, mogą być wykorzystane do identyfikacji procedur użytkownika (rys. 2). Do identyfikacji danych użytkownika (rys. 1, rys. 2) można wykorzystać klasyczne obiekty baz danych, jakimi są tzw. tabele systemowe, które we współczesnych systemach rozproszonych służą do realizacji pewnych powiązań danych dla celów integracyjnych. Zidentyfikowane za pomocą tabel systemowych pola określonych rekordów bazy danych mogą być bezpośrednio związane z identyfikatorami atrybutów elementów informacji systemu EPC. Konieczna dla realizacji takich powiązań jest identyfikacja wszystkich

obiektów mogących mieć znaczenie podczas realizacji rozwiązań integracyjnych. Identyfikatory takie są przechowywane w swoistym repozytorium. Sposób dostępu do tego repozytorium jest zależny od konkretnego systemu zarządzania relacyjną bazą danych. W SQL Server repozytorium to jest dostępne za pomocą widoku sys.objects (rys. 3). Widok ten jest elementem obiektów systemowych tworzonych zawsze podczas powstawania nowej bazy danych. Odpowiednikiem tego widoku w bazie danych Oracle jest bazowa tabela sys.obj$.

name	object_id	principal_id	schema_id	parent_object_id	type	type_desc	create_date	modify_date
wystawca	37575...	NULL	1	0	U	USER_TABLE	2010-06-21 23:06:53.663	2010-06-21 23:06:53.667
formaplatnosci	53575...	NULL	1	0	U	USER_TABLE	2010-06-21 23:06:53.680	2010-06-21 23:06:53.683
PK_formaplatno...	69575...	NULL	1	53575229	PK	PRIMARY_KEY_CONSTRAI...	2010-06-21 23:06:53.680	2010-06-21 23:06:53.680
odbiorca	85575...	NULL	1	0	U	USER_TABLE	2010-06-21 23:06:53.680	2010-06-21 23:06:53.683
PK_odbiorca	10157...	NULL	1	85575343	PK	PRIMARY_KEY_CONSTRAI...	2010-06-21 23:06:53.680	2010-06-21 23:06:53.680
MyImages	11757...	NULL	1	0	U	USER_TABLE	2010-06-24 20:56:44.717	2010-06-24 20:56:44.780
PK_MyImages	13357...	NULL	1	117575457	PK	PRIMARY_KEY_CONSTRAI...	2010-06-24 20:56:44.777	2010-06-24 20:56:44.777
QueryNotificatio...	19770...	NULL	1	0	SQ	SERVICE_QUEUE	2005-10-14 01:36:25.360	2005-10-14 01:36:25.360

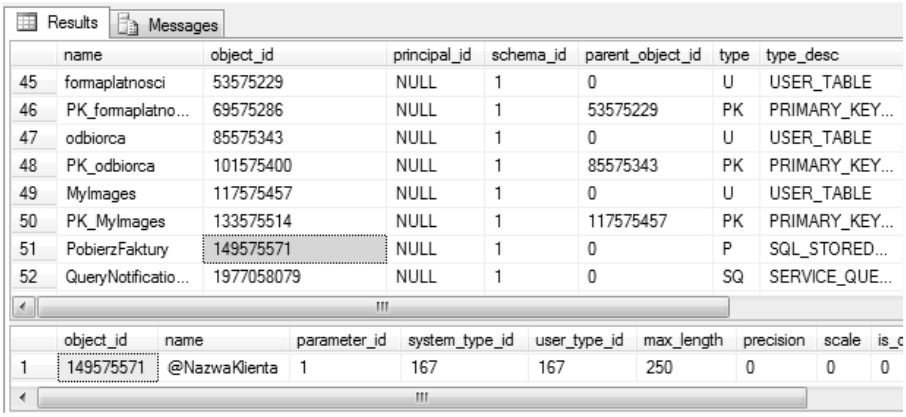
Rys. 3. Zrzut ekranowy wybranych rekordów tabeli sys.object SQL Server 2008 (opracowanie własne na podstawie narzędzia SQL Server Management Studio)

Jak widać na rysunku 3, oprócz identyfikatorów i nazw obiektów udostępnianych przez warstwę bazy danych, można odczytać typ każdego z obiektów, jego datę utworzenia, ostatniej modyfikacji, oraz identyfikator obiektu bezpośrednio nadrzędnego – tzw. rodzicielskiego. Wszystkie te dane mogą być użyte do bezpośredniego zidentyfikowania obiektu. Mając odpowiednie uprawnienia do korzystania z obiektu można użyć go w celu pozyskania danych z istniejącej bazy danych. Podobnym mechanizmom identyfikacji w bazie danych podlegają obiekty, które umożliwiają manipulacje danymi, w tym funkcje i procedury wbudowane w relacyjną bazę danych.

W przypadku wywołania procedur czy funkcji wbudowanych w bazę danych oprócz wiedzy o nazwie procedury konieczne jest podanie wymaganych parametrów wejściowych i ewentualnie wyjściowych. Pozwala to na komunikację z istniejącymi systemami bazodanowymi i wykorzystanie wcześniej zaimplementowanej logiki biznesowej. W wielu implementacjach praktycznych procedury wbudowane i wykorzystanie ich wraz minimalizacją uprawnień do innych obiektów bazodanowych stanowi metodę podwyższenia bezpieczeństwa integralności danych i realizację wydajnej komunikacji bazy danych i aplikacji. Takie podejście zwalnia projektantów od modyfikacji warstwy aplikacyjnej nawet podczas zmiany struktury samych tabel bazy danych. Konieczne jest jedynie przebudowanie procedur i funkcji wbudowanych w bazę wraz z zachowaniem ich prototypów umożliwiających komunikowanie się ze światem zewnętrznym w określony sposób.

Informację o strukturze parametrów, ich typie czy nazwie można również wyselekcjonować używając predefiniowanych tabel systemu zarządzania relacyjną bazą danych. Dla przykładu SQL Server 2008 udostępnia widok systemowy sys.parameters umożliwiający pozyskanie informacji o udostępnianych parametrach. Przykładowy zrzut ekranowy danych udostępnianych przez widok sys.parameters pokazuje rysunek 4. Na rysunku uwidoczniono również powiązanie pomiędzy unikatowym numerem procedury (reprezentowanym przez pole obiect_id), która jest opisana rekordem danych w tabeli sys.object, a jej parametrami. Powstaje przy tym problem zgodności typów danych

pochodzących z różnych źródeł, tożsamy z problemem rzutowania typów podczas budowy aplikacji. Teoria EPC zakłada istnienie ogólnych typów prostych do reprezentacji danych opisujących atrybuty elementów informacji systemu EPC, ale przy wprowadzeniu danych z różnych rzeczywistych systemów informatycznych należy zastosować taki typ, który jak najlepiej pozwoliłby odwzorować rzeczywiste informacje. Jak widać na rysunku 4 rekord opisujący parametr procedury wbudowanej zawiera nazwę parametru i identyfikator typu parametru. W przypadku parametrów złożonych zgodnych z formatem XML w rekordzie jest przechowywany identyfikator kolekcji XML.


Results Messages

	name	object_id	principal_id	schema_id	parent_object_id	type	type_desc
45	formaplatnosci	53575229	NULL	1	0	U	USER_TABLE
46	PK_formaplatno...	69575286	NULL	1	53575229	PK	PRIMARY_KEY...
47	odbiorca	85575343	NULL	1	0	U	USER_TABLE
48	PK_odbiorca	101575400	NULL	1	85575343	PK	PRIMARY_KEY...
49	MyImages	117575457	NULL	1	0	U	USER_TABLE
50	PK_MyImages	133575514	NULL	1	117575457	PK	PRIMARY_KEY...
51	PobierzFaktury	149575571	NULL	1	0	P	SQL_STORED...
52	QueryNotificatio...	1977058079	NULL	1	0	SQ	SERVICE_QUE...

	object_id	name	parameter_id	system_type_id	user_type_id	max_length	precision	scale	is_c
1	149575571	@NazwaKlienta	1	167	167	250	0	0	0

Rys. 4. Powiązanie pomiędzy procedurą a parametrami w tabelach systemowych bazy danych SQL Server (opracowanie własne na podstawie SQL Server 2008)

Ważnym elementem identyfikacji danych umieszczonych w tabelach powiązanych ze sobą relacjami jest możliwość jednoznacznej identyfikacji poszczególnych wierszy. Pozwala to na rozróżnienie takich danych, które mają to samo znaczenie, a są przechowywane w innych strukturach relacyjnej bazy danych. Mechanizmy pozwalające na dokonanie takiej identyfikacji są również udostępniane domyślnie przez oprogramowanie służące do zarządzanie relacyjną bazą danych, choć ich szczegółowa budowa i wykorzystanie zależy od konkretnego narzędzia bazy danych. W bazie danych Oracle istnieje zestaw pewnych wirtualnych kolumn umożliwiających dokonanie identyfikacji rekordu. W tym celu można wykorzystać dodatkowe kolumny ROWID oraz NUMROW. Są one automatycznie dodawane przez system zarządzania relacyjna bazą danych podczas tworzenia każdej tabeli. Należy jednak pamiętać, iż wartość umieszczona w kolumnie ROWID dla każdego rekordu odzwierciedla jego fizyczne położenie w strukturach bazy danych, dlatego podczas pewnych czynności administracyjnych modyfikujących fizyczne położenie rekordu wartość ta może ulec zmianie. Wykorzystanie w celach identyfikacyjnych wartości NUMROW zdaje się więc pewniejszym rozwiązaniem. Wraz z numerem obiektu przypisanym do każdej tabeli bazy danych wartość kolumny NUMROW stanowi unikatowy identyfikator rekordu [7].

Baza danych SQL Server, choć nie posiada bezpośredniego odpowiednika mechanizmu wirtualnych kolumn w tabeli, posiada szereg mechanizmów pozwalających na budowę zastępczej architektury pełniącej podobną rolę i umożliwiającą generowanie i udostępnianie unikatowego identyfikatora wiersza. Generowany przez system unikatowy numer wiersza

może być zaimplementowany podczas tworzenia samej tabeli w relacyjnej bazie danych. Kolumna w tabeli o typie mającym charakter wyliczeniowy posiada właściwość IDENTITY, która pozwala na generowanie unikatowych wartości klucza. Ponadto wraz z właściwością SEED IDENTITY można skonfigurować automatyczne numerowanie poszczególnych kolumn wstawianych do bazy danych. Wygenerowane numery wraz z identyfikatorem pozyskanym z tabeli sys.objects mogą spełniać rolę identyfikatora unikatowego w całej bazie danych.

Alternatywnym rozwiązaniem jest użycie funkcji systemowej SQL Server o nazwie NEWID. Zwraca ona wartość typu uniqueidentifier. Unikalność zwracanej przez funkcję wartości jest zapewniania przez samą bazę danych. Identyfikator wiersza wygenerowany w ten sposób można uznać za unikatowy na skalę całej bazy a nie tylko jednej tabeli. Ten mechanizm numerowania wierszy jest często używany w przypadku braku możliwości fizycznego przechowywania danych w jednej tabeli, gdy zachodzi potrzeba użycia rekordów z wielu tabel relacyjnej bazy z jednoczesną koniecznością rozróżnienia każdego rekordu. Mechanizm ten może być stosowany z powodzeniem w celach integracji danych w bazach rozproszonych. Wraz z wbudowanymi w nowoczesne środowiska bazodanowe rozwiązaniami ETL niesie ze sobą bogatą funkcjonalność zapewniającą pozyskiwanie danych w rozproszonej architekturze i może stanowić podstawę do zaprojektowania modułu integrującego elementy warstwy bazy danych wraz z obiektami przetwarzania tych danych w rozproszonej architekturze. Wykorzystanie mechanizmów pozwalających na wygenerowanie identyfikatorów numerycznych poszczególnych rekordów wraz z odpowiednim numerem kolumny pozwalają na jednoznaczne skojarzenie danych każdego systemu relacyjnej bazy danych z identyfikatorami danych w szkieletowym systemem EPC.

3. Mechanizmy udostępniania i identyfikacji danych oraz obiektów w warstwie logiki biznesowej na przykładzie rozwiązań Web Services

W wielu systemach informatycznych dane nie są dostępne wprost poprzez połączenia do relacyjnej bazy danych, ale za pomocą specjalnie przygotowanych do tego usług działających w warstwie logiki biznesowej. W przypadku integracji takich usług ze szkieletowym systemem EPC potrzebne są mechanizmy umożliwiające identyfikację procedur i danych udostępnianych przez rozpatrywane usługi.

Do mechanizmów mogących służyć do komunikacji międzyprocesowej należą mechanizmy usług Web Services. Umożliwiają one różnym systemom wymianę danych, oraz przetwarzanie danych rozproszonych w obrębie różnych lokalizacji udostępnianych sieciowo. Ta metoda komunikacji pomiędzy systemami przetwarzania danych jest niezależna od zastosowanej technologii, systemu operacyjnego czy architektury sprzętowej. Niezależność tę osiągnięto poprzez zastosowanie ujednoliconych standardów opartych na składni XML. Do wywoływania usług Web Services stosuje się najczęściej zestandaryzowany zestaw protokołów, takich jak SOAP (Simple Object Access Protocol). Popularnym rozwiązaniem jest również zastosowanie obiektów JSON do wymiany danych w ramach komunikacji pomiędzy Web Services. Wyszukiwanie i integracja Web Serwisów odbywa się przy pomocy mechanizmów UDDI (Uniwersal Description Discovery and Integration). Nie są one własnością żadnych konsorcjów i nie objęte są żadnymi prawami patentowymi. Dzięki temu jest możliwe ich zastosowanie na szeroką skalę w rozmaitych systemach wymiany danych i użycie jako sposobu integrowania logiki biznesowej oraz danych. Nie bez znaczenia jest również fakt, iż organizacja World Wide Web Consortium (W3C) uznała mechanizmy wykorzystywane przez Web Services za standard. Mimo że

usługi Web Services działają niezależnie od siebie, możliwe jest ich wzajemne łączenie celem rozwiązania problemów integracji rozproszonych systemów informatycznych [1].

Web Services jest swoistą biblioteką funkcji, które mogą być wywołane zdalnie na potrzeby klienta. Zarówno parametry wejściowe funkcji, jak i uzyskane wyniki, są przekazywane poprzez sieć najczęściej z wykorzystaniem protokołu HTTP. Aby klient mógł wykorzystać funkcję zdalnie udostępnianą przez usługę, musi znać adres URL serwisu webowego. Adres ten może być swoistym identyfikatorem usługi. Są tu możliwe dwa podejścia. Wywołanie serwisu bezpośrednio poprzez jego adres URL, albo wyszukanie adresu serwisu w centralnym repozytorium tzw. Service Brokera. Budując rozproszoną logikę biznesową aplikacji najczęściej mamy do czynienia z pierwszym scenariuszem postępowania. Chcąc jednak skorzystać ze zdalnych serwisów webowych udostępniających swoje funkcje szerokiemu gronu klientów poprzez sieć Internet wygodnie jest skorzystać z centralnego repozytorium. Centralna baza danych, w której są zamieszczone adresy serwisów i sposoby przekazywania danych wejściowych do funkcji zdalnych udostępnianych przez serwisy, jest opisywana przez UDDI (Universal Description, Discovery and Integration). Prace nad wersją 1.0 języka UDDI zostały sfinalizowane przez organizację W3C w 2000 roku. Pierwsza wersja tego standardu przewidywała jedynie możliwość prostego utrzymania opisu serwisów webowych udostępnianych przez innych usługodawców. W roku 2003 wprowadzono wersję 2.0 specyfikacji UDDI, która została rozszerzona o elementy taksonomii dla opisywanych serwisów. Już rok później pojawiła się wersja 3.0, w której uwzględniono aspekty związane z autoryzacją klientów i bezpieczeństwem dostępu do usług. UDDI specyfikuje repozytorium serwisów webowych na trzech płaszczyznach:

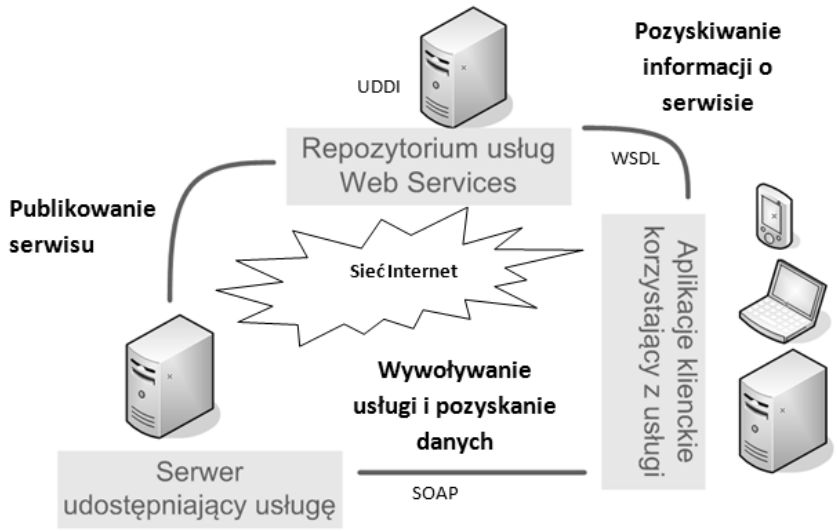
- White Pages – udostępnia dane o adresie usługodawcy,
- Yellow Pages – udostępnia kategorię taksonomiczną usługi w klasyfikacji przemysłowej,
- Green Pages – udostępnia opis wywołania usługi; referencja do pliku WSDL.

Opis wywołania usługi udostępniany poprzez repozytorium dostarcza informacji o procedurach, jakie mogą być wywołane na rzecz klientów zewnętrznych, o parametrach wejściowych oraz o parametrach zwracanych wyników. Pozwala to na zbudowanie interfejsów komunikacyjnych i zdalnych wywołań takich procedur z poziomu procedur inicjowanych poprzez tranzycje szkieletowego systemu EPC.

W trakcie prac nad ujednoliceniem specyfikacji UDDI wiele ze znanych firm branży informatycznej (w tym SAP, IBM, Microsoft) uruchomiło swoje repozytoria Web Services, do których mogły być rejestrowane publiczne usługi. W chwili obecnej centralne repozytoria przestały być utrzymywane na rzecz powstałych repozytoriów lokalnych, które można wygenerować za pomocą wielu dostępnych na rynku komercyjnych produktów. Pełna specyfikacja sposobu komunikacji ze zdalną usługą Web Service jest udostępniana za pomocą języka WSDL (Web Service Description Language). W rozproszonych systemach informatycznych zorientowanych usługowo do opisu komunikacji z usługą używa się pojęcia kontraktu. Kontrakt można zdefiniować jako zbiór komunikatów jakie mogą być wysyłane i odbierane poprzez usługi przesyłające dane pomiędzy sobą. Aby komunikacja pomiędzy serwisem i korzystającym z niego klientem przebiegła pomyślnie należy również opisać typy danych jakimi posługują się obie strony konwersacji. Język WSDL umożliwia pełną specyfikację protokołu używanego do komunikacji poprzez wprowadzenie w swoich strukturach węzłów takich jak:

- Types – węzeł ten opisuje typy abstrakcyjne używane przy wymianie danych z funkcją serwisu. Do opisu typów zaleca się zastosowanie schematu pliku XML.
- Message – węzeł definiuje abstrakcyjny komunikat, który może być zbudowany z wielu części. Każda z części komunikatu może być innego typu, przy czym używane typy muszą być opisane odpowiednimi węzłami types.
- PortType – węzeł definiuje zbiór obsługiwanych operacji i stanowi tzw. Opis interfejsu. Opis ten jest definiowany poprzez zestawienie komuniatów wejściowych i wyjściowych.
- Binding – węzeł opisuje komunikaty i typy dla konkretnego zbioru operacji wyspecyfiokowanego poprzez PortType
- Service – węzeł jest odpowiedzialny za opisanie zbioru punktów końcowych powiązanych ze sobą, opisuje zbiór elementów port i endpoint, które udostępniają poszczególne wiązania w sieci.

Plik WSDL ma postać tekstową opartą o XML i jest obsługiwany w łatwy sposób przez szereg narzędzi służących do projektowania oprogramowania. Narzędzia te wspierają projektantów w tworzeniu klas pośredniczących generowanych na podstawie informacji pozyskanych z plików WSDL i pozwalających na wywołanie funkcjonalności udostępnianych przez Web Services za pomocą takich samych konstrukcji składniowych języków programowania jak wywołanie funkcji czy procedur z bibliotek lokalnych.



Rys. 5. Architektura wymiany danych z użyciem Web Services
(opracowanie własne na podstawie dokumentacji Web Services)

Architektura komunikacji z usługami Web Services w swej klasycznej postaci przedstawionej w tym rozdziale nie rozwiązuje problemów bezpieczeństwa informacji. Z punktu widzenia bezpieczeństwa informatycznych systemów zarządzania należy przedsięwziąć odpowiednie środki zapewniające poufność, integralność, niezaprzeczalność,

uwierzytelnianie oraz autoryzację. Rozwiązanie tych problemów jest możliwe dzięki zastosowaniu dodatkowych standardów, np. opracowanych przez organizację OASIS (ang. Organization for the Advancement of Structured Information Standards). Organizacja ta zajmuje się opracowaniem dodatkowych niezależnych mechanizmów pozwalających na budowę bezpiecznych systemów e-comerce oraz informatycznych rozwiązań dla potrzeb biznesu. Ideą prac organizacji jest opracowanie i rozpropagowanie niezależnych standardów, które mogłyby zdobyć taką popularność jak usługi Web Services w swej klasycznej postaci. Istnieje również wiele rozwiązań komercyjnych podwyższających bezpieczeństwo Web Services, do których należą dodatkowe biblioteki opracowane przez firmę Microsoft o nazwie WSE (ang. Web Service Enhancement) dostępne obecnie w wersji 3.0.

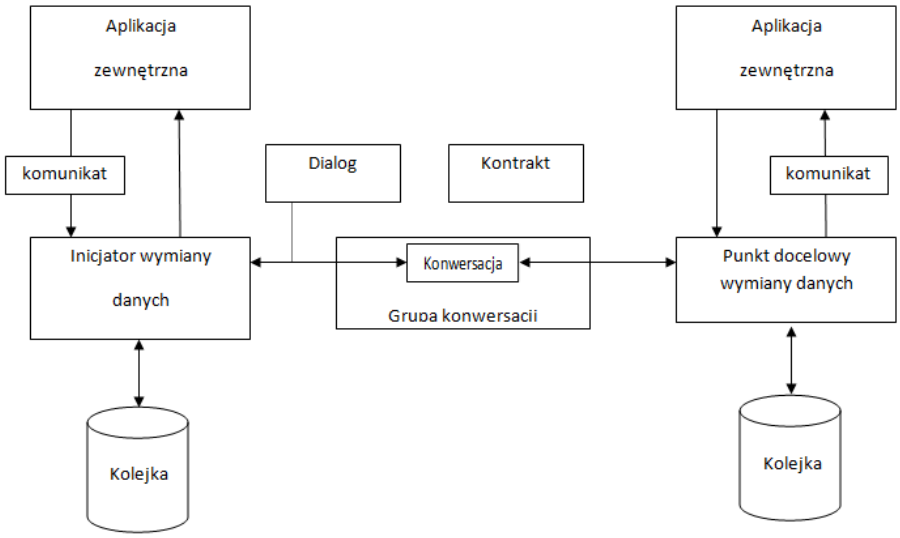
4. Integracja mechanizmów dostępnych w architekturze zorientowanej usługowo z warstwą bazy danych przy pomocy usług Service Broker

W wielu implementacjach dane zewnętrzne mogą być pozyskiwane nie tylko poprzez połączenie się z bazą danych, ale poprzez skorzystanie ze specjalnie przygotowanych do tego celu usług. Dzięki temu systemy, które nie korzystają z relacyjnych baz danych bezpośrednio, ale wykorzystują do komunikacji formaty wymiany danych takie jak XML, mogą odzwierciedlać wysyłane przez siebie dane w tabelach relacyjnych baz danych. Wpisanie danych w bazę relacyjną umożliwia identyfikację danych w celu ich skojarzenia ze szkieletowym systemem EPC za pomocą mechanizmów analogicznych do przedstawionych w rozdziale 2. Od wersji bazy danych SQL Server 2005 udostępniono możliwość wykorzystania usług zgodnych z architekturą SOA poprzez rozszerzenia języka T-SQL i użycie komponentu Service Broker. Komponent ten jest ściśle zintegrowany z silnikiem bazy danych, przez co stanowi dodatkową warstwę pośredniczącą, umożliwiającą zaprojektowanie architektury rozproszonej i funkcjonalności integracji danych [2]. Za pomocą SQL Server Service Broker można zrealizować dwukierunkową implementację przesyłania komunikatów pomiędzy procesami zewnętrznymi i wewnętrznymi. Przesyłanie komunikatów jest realizowane w sposób transakcyjny, a do ich przechowywania w elementach takich jak kolejki są wykorzystywane tabele relacyjnej bazy danych. Ponadto mechanizm Service Broker integruje ze sobą zarówno dane jak i logikę przetwarzania danych zapewniając jednocześnie potwierdzenia otrzymania komunikatów, jak i odpowiednią kolejność ich przetwarzania. Implementacja mechanizmu przesyłania komunikatów za pomocą Service Broker zapewnia również asynchroniczną komunikację. Klientami usług Service Broker mogą być zarówno wewnętrzne procedury bazy danych jak i zewnętrzne procesy usługowe, zainicjowane pod wpływem pewnych impulsów zewnętrznych, takich jak zdarzenia wynikające z usług harmonogramowania zadań. Impulsem do zainicjowania żądania dla usługi Service Broker może być więc dowolny program umożliwiający komunikowanie się z bazą danych. W celu komunikacji z usługą Service Broker należy zdefiniować następujące elementy:

- Typ komunikatu – typ ten jest określony poprzez schemat pliku XML. Można również określić go jako dowolny plik, poprawny w sensie składni języka XML.
- Kontrakt – określa sposób wysyłania odpowiednich komunikatów, ich kolejność oraz to, które z komunikatów są wysyłane przez stronę odbierającą, a które przez stronę wysyłającą.

- Dialog – wiąże ze sobą kontrakt, punkty komunikacji oraz określa typ wiadomości inicjujących dialog. Wiele elementów typu dialog może być powiązanych z tzw. konwersacją.
- Konwersacja – jest długotrwałą wymianą wiadomości mogącą mieć różny wymiar czasu, która ma charakter asynchroniczny.
- Grupa konwersacji – jest zespołem dialogów w obrębie wielu konwersacji, które maja na celu implementowanie pewnej funkcji biznesowej.

Odpowiednia konfiguracja powyższych elementów umożliwia opisanie zasad wymiany danych z zewnętrznymi systemami. Poglądowy schemat komunikacji, zgodny z wymogami architektury zorientowanej usługowo implementowanej za pomocą mechanizmu Service Broker, został przedstawiony na rysunku 6.



Rys. 6. Schemat wymiany danych y użyciem komponentów Service Broker SQL Server (opracowanie własne na podstawie dokumentacji technicznej Microsoft)

5. Wnioski

Rozproszonym systemom wymiany i przetwarzania danych, do których można zaliczyć również systemy zarządzania, oprócz wymagań funkcjonalnych stawiane są również coraz częściej wymagania integracyjne. Są one podyktowane koniecznością powiązania ze sobą szeregu rozmaitych systemów przetwarzania danych zrealizowanych w oparciu o różne rozwiązania techniczne zarówno pod względem sprzętowym jak i wykorzystywanego oprogramowania. Mechanizmy identyfikacji oraz integracji danych, udostępniane przez nowoczesne systemy zarządzania relacyjnymi bazami danych oraz przez usługowe serwisy webowe mogą być zastosowane także do odzwierciedlania danych zewnętrznych oraz do

wykorzystywania zewnętrznych elementów logiki biznesowej w szkieletowym systemie EPC.

Jednoznaczna identyfikacja danych oraz elementów logiki biznesowej, dostarczanych przez komponenty informatycznych systemów zarządzania za pomocą mechanizmów przedstawionych w artykule, pozwala na wymuszenie związków 1 do 1 pomiędzy tymi danymi oraz elementami i ich identyfikatorami w szkieletowym systemie EPC. Dzięki temu jest możliwa praktyczna implementacja kooperacyjnych systemów EPC z wykorzystaniem już istniejących komponentów informatycznych systemów zarządzania. Szkieletowy system EPC może być wtedy uważany za warstwę integracyjną rozproszonych systemów zarządzania.

Przedstawione w artykule mechanizmy integracyjne mogą wykorzystane nie tylko w kooperacyjnych systemach EPC. W taki sam sposób mogą one działać w symulatorze informatycznych systemów zarządzania, zbudowanym na bazie szkieletowego systemu EPC [13], gdzie służą do wprowadzania procedur oraz przetwarzanych przez nie danych z badanych systemów do symulatora. Dzięki temu można dokonywać porównań różnych systemów zarządzania, badanych w odrębnych eksperymentach symulacyjnych przy takich samych warunkach początkowych i takich samych oddziaływaniach zewnętrznych. Symulowane systemy odnoszą się do tego samego przedsiębiorstwa lub jego części, ale w każdym eksperymencie różnią się metodami zarządzania, np. strukturami przepływu informacji w systemach ERP oferowanych przez różnych dostawców, lub procedurami przetwarzania danych, które są przypisane do poszczególnych tranzycji systemu EPC.

Praca finansowana w ramach badań statutowych
Wyższej Szkoły Biznesu w Dąbrowie Górniczej.

Literatura

1. Barry D.K.: Web Services and Service-Oriented Architectures: The Savvy Manager's Guide. Morgan Kaufmann Publishers, ISBN 1-55860-906-7.
2. Brust A., Forte S.: Programowanie Microsoft SQL Server 2005. Promise, Warszawa 2005
3. Davis R., Brabander E.: ARIS Design Platform. Getting Started with BPM. Springer-Verlag, Berlin, 2007.
4. Frączkowski K., Mazur Z.: SOA – architektura zorientowana na usługi. Bazy Danych, nr 7, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław, 2006.
5. Jensen K.: Coloured Petri Nets. Springer-Verlag, Berlin, 1997.
6. Kasprzak T. (red.): Modele referencyjne w zarządzaniu procesami biznesu. Difin, Warszawa, 2005.
7. Loney K.: Oracle, podręcznik administratora baz danych. Oracle Press, Warszawa 2002.
8. Scheer A.-W.: Business Process Engineering. Reference Models for Industrial Enterprises. Springer-Verlag, 1994.
9. Troelsen A.: Język C# 2008 i platforma .NET 3.5. Wydawnictwo Naukowe PWN, Warszawa, 2009.
10. Zaborowski M.: Outline of the Enterprise Resource Control Systems Architecture. Advances in Intelligent and Soft Computing, vol. 64, Springer-Verlag, Berlin, 2009, pp. 131-140.

11. Zaborowski M.: The EPC theory. Basic Notions of Enterprise Process Control. Management and Production Engineering Review, Vol. 1, No 3, September 2010.
12. Zaborowski M.: The EPC theory. Couplings between Transitions in Enterprise Process Control Systems. Management and Production Engineering Review, Vol. 1, No 4, December 2010.
13. Zaborowski M.: Ogólny opis i zakres zastosowań szkieletowego systemu sterowania procesami przedsiębiorstwa. W: Knosala R. (red.) „Komputerowo Zintegrowane Zarządzanie", Oficyna Wyd. Polskiego Towarzystwa Zarządzania Produkcją, Opole, 2011.

Mgr inż. Paweł BUCHWALD
Katedra Informatyki
Wyższa Szkoła Biznesu w Dąbrowie Górniczej
41-300 Dąbrowa Górnicza, ul. Cieplaka 1c
e-mail: pawel@buchwald.com.pl