Zygmunt Ryznar

Integrowanie bankowego systemu informatycznego

W praktyce prawie wszystkie banki mają do pokonania wiele problemów integracyjnych wynikających z niejednorodności środowiska informatycznego (różny typ komputerów, różne systemy operacyjne, różne systemy zarządzania bazami danych, różni dostawcy i autorzy oprogramowania aplikacyjnego oraz narzędziowego, itp. Z takich czy innych względów prawie zawsze i wszędzie istnieje potrzeba dopasowywania poszczególnych składników systemu do siebie. W wyniku takiego "nieuporządkowanego" stanu rzeczy istnieje w banku "wiele prawd" (źródeł informacji) na ten sam temat, co wymaga wielu uzgodnień, korekt i filtracji danych przed dokonaniem konsolidacji, np. dla potrzeb analitycznego CRM - Customer Relationship Management (ci sami klienci występują w różnych jednostkach organizacyjnych pod innymi symbolami i posiadają inną charakterystykę). Wg Gartner Group w typowym dużym przedsiębiorstwie 35-40% całego wysiłku programistycznego przeznacza się na oprogramowanie, którego jedynym celem jest transfer danych (obejmujący ekstraktyzację, transformację, replikację, rozładowania i ładowania aktualizacyjne) pomiędzy różnymi bazami danych.

Wymagania integracyjne w bankowości

Banki nie zaczynają od “zera”. Z reguły na różnego rodzaju komputerach pracuje wiele “historycznych” autonomicznych aplikacji bankowych, kupionych dla potrzeb poszczególnych jednostek organizacyjnych lub na "potrzeby chwili". Trzeba je usunąć, jeśli funkcje które realizują znajdują się w bardziej kompleksowym systemie, albo wymienić na lepsze lub dorobić połączenia od nich i do nich. Nie można jednakże tworzyć całego systemu bankowego wyłącznie w drodze "sklejania" pakietów, gdyż "mimo, iż małe jest piękne" to "duże będące ich sumą " może okazać się tworem pokracznym o niskiej wydajności (m.i. ze względu na pośrednictwo wielu mechanizmów interfejsowych) i trudnym do utrzymywania (każdy pakiet korzystać może z innej organizacji plików, posiadać inną organizację menu i komunikację z użytkownikami itp.). Przynajmniej rdzeń (jądro) systemu powinien być produktem jednolitym, narzucającym architekturę systemu, m.i. poprzez utrzymywanie wspólnych podstawowych baz danych (klientów, rachunków).

System bankowy- rozumiany jako zestaw aplikacji używanych w banku- powinien być nie tylko kompleksowy od strony funkcjonalnej, tzn. realizujący funkcje zarówno bankowości korporacyjnej jak i detalicznej, lecz również zintegrowany co najmniej na poziomie danych.

Potrzeby integracyjne wynikają głównie z aplikacji wymagających dostępu globalnego (w skali banku) do potrzeb systemów informowania kierownictwa oraz do uzyskiwania takich zestawów informacyjnych jak pozycja walutowa banku, pozycja klienta, globalne limity (klienta, grupy kapitałowej, branży, kraju), przepływ pieniężny w skali banku, zestaw danych o kliencie pojawiający się na ekranie operatora systemu "call/contact center". Wreszcie argumentem za integracją danych jest przeciwdziałanie dublowaniu takich samych danych (np. nazwisko klienta, adresy itp.) w różnych aplikacjach.

Praktycznym sprawdzianem stopnia integracji informacji o klientach w skali systemu jest możliwość otrzymywania w trybie on-line (czyli np. z terminala) i w czasie rzeczywistym (a wiec w danej chwili) aktualnych *informacji o pozycji klienta* w . skali całego banku . Pozycja klienta obejmuje zwykle takie informacje jak: ogólne saldo rozliczeniowe netto, salda (w tym średnie) i obroty na poszczególnych rachunkach i grupach produktów, odsetki naliczone, pobrane i wpłacone, opłaty, raty kredytowe przeterminowane, gwarancje, limit globalny i limity przedmiotowe, kategoria ryzyka kredytowego itp.

Szczególnie wiele zadań w zakresie integracji aplikacji poprzez danych może występować w zakresie powiązań pomiędzy systemem bankowości komercyjnej (hurtowej) i systemem bankowości detalicznej - oczywiście w przypadku jeśli bank nie dokonał zakupu systemu kompleksowego. W obszarze powiązań pomiędzy tymi dwoma sferami występują takie pliki danych jak plan kont, księga główna, baza klientów, tablica walut, tablica kursów wymiany walut, pozycja klienta w banku, kalendarze dni roboczych (lub nieroboczych) dla walut i krajów, transakcje na rachunkach nostro

w bankach zagranicznych itd. W kompleksowych nowoczesnych systemach pliki te, wraz z rachunkami klientów, tworzą wspólną bazę danych utrzymywaną centralnie, dostępną bezpośrednio (a nie poprzez mechanizmy eksportowo-importowe) dla każdej aplikacji.

Niezbędnym minimum integracji modułu obsługi operacji zagranicznych z pozostałymi częściami systemu bankowego jest wspólna baza danych obejmująca rachunki nostro (plus krajowe rachunki dewizowe), waluty i kursy walut. Obsługa operacji dilerskich wymaga bowiem zapewnienia informacji o bieżącym stanie rachunków nostro i walut, co oznacza, że powinny być one od razu aktualizowane przez wszystkie transakcje mające z nimi związek.

Pożądane jest aby wspólna baza danych obejmowała również klientów i banki. Moduł informacji o klientach powinien być wykorzystywany przez wszystkie moduły funkcjonalne systemu. Operacyjna baza informacji o klientach pełni zwykle ważną rolę integrującą w systemie, gromadząc m.i. wszystkie informacje niezbędne do określenia globalnej pozycji pozycji klienta w banku. Informacje o klientach znajdują się również w hurtowni marketingowo-klientowskiej, przeznaczonej zwykle do automatycznej segmentacji klientów.

Integracja komunikacyjna ze środowiskiem,

Integracja systemu ze środowiskiem dotyczy powiązań komunikacyjnych (w tym doboru protokołów) z innymi aplikacjami funkcjonującymi w banku oraz na zewnątrz, powiązań w ramach architektury klient/-serwer oraz łączności z klientami (home/corporate banking) itp.

Ponadto w ramach samego systemu podstawowego (core system) występują potrzeby integracyjne komputera centralnego z komputerami lokalnymi w oddziałach, baz centralnych i lokalnych (łącznie z problemami uzgodnień pomiędzybazowych po awarii łączy telekomunikacyjnych) itp.

Podstawową metodą integracji komunikacyjnej jest stosowanie oprogramowania pośredniczącego middleware. Zadania związane z “uniwersalną translacją danych” (universal data translation) pomiędzy aplikacjami eksploatowanymi na różnych platformach sprzętowych wchodzą również w zakres tzw. middleware. Middleware integrujący aplikacje na Zachodzie określany jest skrótem EAI (Enterprise Application Integration). Do czołowych firm działających w tym zakresie należą m.i. Neon, Verastream, Vitria. Do ebiznesu stosowane jest ostatnio pojęcie eAI, charakteryzujące się intensywnym wykorzystaniem Javy i XML oraz integracyjnych serwerów (serwerów aplikacyjnych dla middleware). W kategorii eAI do czołowych firm - w szczególności dla e-biznesu - zaliczają się firmy . stosujące technologię specjalizowanych serwerów aplikacyjnych BEA (WebLogic), IBM (WebSphere), Ariba (B2B), IONA (iPortal Integrator), TopTier (wchodzi w skład SAP), Sybase (EP-Enterprise Portal). Klasycznym zadaniem integracyjnym jest w middleware konwersja danych na wspólny format danych za pomocą aplikacyjnych serwerów integracyjnych.

Za pomocą oprogramowania pośredniczącego wykonuje się nie tylko prace integracyjne lecz też buduje aplikacje wymagające dostępu do wielu systemów (np. systemy call/contact center, CRM).

W komunikacji pomiędzy aplikacjami stosowane są rozmaite schematy połączeniowe. Połączenie synchroniczne "point-to-point" czyli "punkt-punkt” (wiadomość przesłana musi być bezwłocznie przyjęta) ma miejsce w przypadku zastosowania wywołań RPC (*Remote Procedure Call)* zaś asynchroniczne MOM (*Messaging Oriented Middleware*) gdy występuje kolejkowanie komunikatów np. przy uzyciu IBMowskiego oprogramowania MQSeries, MSMQ lub ActiveExchange firmy Tibco albo MessageQ firmy BEA. Schemat "many-to-many" czyli "wiele do wielu” występuje przy użyciu brokerskiego systemu wymiany komunikatów. Z kolejkowaniem komunikatów może być skojarzona "translacja" danych (przeformatowanie) a odbywać się to może np. poprzez oprogramowanie firm WebMethods, SeeBeyond (przedtem STC) i Vitria.

Integracja pomiędzy bazami danych

Do komunikacji z bazami danych stosowane są łącza bazodanowe: natywne, ODBC (Open Data Base Connectivity), JDBC(Java DataBase Connection) lub RDD (Replaceable Data Driver).

Przykładem pakietów do komunikacji międzybazowej są DataMove i ChangeDataMove firmy BMC Software przeznaczone do propagacji danych z IBMowskich rozwiązań IMS, DB/2, CICS, VSAM do Oracle, Sybase i MS SQL Server. Przykładem integracji danych w warunkach sieciowych jest

PowerChannel firmy Informatica, dokonujący transformacji danych pomiędzy węzłami sieci. Oprogramowanie Integration Suite do integracji i transformacji danych oferowane jest przez Data Junction.

Bazodanowe połączenia integracyjne odbywają się w czasie rzeczywistym, pseudo-rzeczywistym (w zadanych interwałach czasowych) lub okresowo (np. na koniec dnia) . Dane nie spełniające wymagań standaryzacyjnych (np. XML) zwykle ulegają przeformatowaniom (zgodnie z wymaganiami odbierającej je aplikacji) wg słowników danych, w tym -w najbardziej zaawansowanej postaci - repozytorium metadanych).

Dane w standardach dokumentów elektronicznych stosowane powinny być szczególnie w aplikacjach e-biznesowych opartych na internetowym dostępie, gdyż można je oprzeć na "naturalnych" standardach tam stosowanych a więc językach HTML (Hyper Text Markup Language), a szczególnie XML (eXtensible Markup Language). Problemy integracji poprzez dane rozwiązywane mogą być wówczas łatwiej rozwiązywane w porównaniu z tradycyjnymi metodami (mapowanie i konwersja danych, ekstraktów w postaci plików tekstowych). Szczególnie przydatny tutaj może być standard XML, polegający na umieszczaniu w dokumencie zarówno danych jak i metadanych (opisu danych), co umożliwia zachowanie w poszczególnych aplikacjach swoich własnych struktur danych i automatyczne ich rozpoznanie przez przeglądarki internetowe lub inne narzędzia przed udostępnieniem innym użytkownikom czy aplikacjom. Głównym odbiorcą dokumentów elektronicznych są na razie użytkownicy końcowi, należy jednakże spodziewać się w przyszłości opracowania automatycznych interfejsów pomiędzy XMLem a relacyjnymi bazami danych i repozytoriami metadanych.

Integrację informacji dla potrzeb hurtowni danych przeprowadza się pod kontrolą administratora hurtowni danych po przez specjalistyczne oprogramowanie ekstraktyzacji, transformacji i kontroli jakości danych, ładowania i aktualizacji hurtowni danych. Oprogramowanie to korzysta czasem z centralnego repozytorium metadanych, za pośrednictwem którego dokonywana jest konwersja danych i kontrola ich jakości. Źródeł zasilania hurtowni danych bywa wiele. Na przykład Bank of America swoją hurtownię o 9 milionach klientach zasilał danymi aż z 49 aplikacji operacyjnych.

Integracja programistyczno-operacyjna

Integracja programistyczno-operacyjna, umożliwia "naturalny" tryb przetwarzania (on-line i real-time) wykorzystując połączenia synchroniczne i asynchroniczne, łącza programistyczne tradycyjne (API) lub obiektowe (wg standardów języka IDL organizacji OMG i języka Java nadzorowanego przez organizację JavaSoft). W integracji programistycznej stosowane są interfejsy obiektowe i standardy API (*Application Programs Interface*), które m.i. definiują standardowe formaty danych, mechanizmy eksportu i importu danych, umożliwiając tym samym łączność z dowolnymi aplikacjami.

Nie zaleca się „inwazyjnego" trybu integrowania danych polegającego na modyfikacji oprogramowania źródłowego aplikacji. Stosowana jest wtedy gdy znane są kody źródłowe integrowanych systemów i istnieją następujące warunki wyjściowe:

1. znane są mechanizmów wywoływania funkcji i procedur (oraz ich wzajemnych powiązań) w systemach integrowanych
2. istnieje wspólna płaszczyzna komunikacyjna (np. zdalne wywoływanie aplikacji - RPC poprzez porty protokołu TCP/IP) oraz mechanizmy API lub Javy
3. możliwe jest ustalenie połączenia konwersacyjnego (sprzężenia zwrotnego - Chodzi o dwukierunkową komunikację: Request/Reply flow oraz mechanizmu "store & forward".) tj. sposobu postępowania gdy nie powiedzie się wywołanie z innego systemu (np. wykonanie rewersyjnej transakcji w systemie wywołującym , wywoływanie iteracyjne dopóki nie zniknie tzw. flaga błędu itp.)
4. znane są struktury danych w obu systemach (chociażby po to aby sprawdzić poprawność danych po uruchomieniu dodatkowych procedur - w prostym przypadku stosowane może być łącze ODBC, zaś w bardziej złożonym "database gateway" dokonujący odpowiedniej konwersji danych pomiędzy rożniącymi się bazami danych) a następnie wykonać konwersje (tzw.mapowanie) danych z jednej aplikacji do drugiej

5. znane są “drzewa “ wywołań procedur i funkcji, aby zapobiec ew. skutkom “łańcuchowym” po modyfikacji oprogramowania
6. kod aplikacji jest izolowany od systemu operacyjnego (izolacja kodu źródłowego od systemu operacyjnego realizowana ma być np. na Microsoftowej platformie .NET (poprzez warstwę CLS - Common Language Specification).

Przykładem integracji programistyczno-operacyjnej jest sytuacja gdy z procedury w jednym systemie wywoływana jest procedura drugiego systemu, np. po wprowadzeniu numeru klienta do pola na ekranie uruchamiana jest tzw. postprocesorowa funkcja ściągające nazwę i dane adresowe z innego systemu. Nie ma tutaj więc przesyłania całych transakcji, lecz tylko parametrów wywołania i wartości zwracanych przez wywoływaną funkcję, zaś dostęp do baz danych odbywa się w trybie normalnym dla systemu macierzystego (natywne/rodzime sterowniki –w tzw. native mode), poprzez łącze ODBC (Open Data Base Connectivity), łącz javowe JDBC (Java Database Connectivity) lub sterowniki RDD (Replaceable Data Driver) opartych na API komunikających się aplikacji.

W przypadku systemów z góry zaprojektowanych do wymiany danych w architekturze klient-serwer (CORBA ORB) możliwe jest zastosowanie “brokerskiego” systemu wymiany komunikatów pomiędzy aplikacjami. polegającego na stosowaniu zarówno “data broker” (do łączności pomiędzy bazami danych) jak i “front-end message broker (do łączności pomiędzy funkcjami/programami). Message broker kopiuje i rozsyła wiadomości do jednego lub wielu punktów przeznaczenia (dokonując ewentualne formatowanie danych dostosowane do wymagań każdego punktu). Istnieją firmy specjalizujące się w middleware do “brokerskiego” przenoszenia danych (np. Hublink, Constellar, Neon, CrossRoads). “Message broker” reprezentuje aktywny typ komunikacji pomiędzy programami, pozwalając zarówno na czytanie i pisanie, podczas gdy “database gateway” realizuje tylko operacje odczytu. Przykładem użycia oprogramowania typu middleware do integracji oprogramowania bankowego jest rozwiązanie w Abbey National Treasury Services , gdzie oprogramowanie Neonet firmy Sybase/Neon (New Era of Networks) użyto do integracji takich systemów jak Reuters Effix, Kondor Plus (operacje dilerskie) i Summit. Wykorzystano takie moduły jak obsługa komunikatów (messaging, queuing), dynamiczne formatowanie danych i ewaluacja (rules-based message evaluation). Oprogramowanie middleware różnych firm moze współpracować ze sobą np. w Disclosure Inc. do integracji różnych systemów operacyjnych i aplikacji użyto – MQSeries (IBM) w połączeniu z formaterem i ewaluatorem firmy Neon.
Oprogramowanie brokerskie (wg standardu CORBA- Common Request Broker Architecture) dla środowiska heterogenicznego oferowane jest również przez firmę BEA pod nazwą eLink i DIO (Data Integration Option). Do oprogramowania pośredniczącego zalicza się również IBMowski DB2 Connect , który pozwala m.i. na udostępnianie tej samej bazy danych DB2 aplikacjom pracującym pod różnymi systemami operacyjnymi (np. Linux i OS/390). Większość firm (w tym IBM, BEA) ewoluuje od standardu CORBA w stronę rozwiązań javowych.

Zestawienie niektórych wytwórców oprogramowania integracyjnego

Podsumowanie

Przedsięwzięcia integracyjne są szczególnie złożone w przypadku pakietów programistycznych pochodzących od różnych autorów. całkowicie autonomicznych, tzn. posiadających komplet swoich własnych danych (klientów, banków, walut, kont księgowych) i nie wyposażonych w odpowiednie łącza do innych aplikacji (np. mechanizmy importu i eksportu danych z parametryzowaną konwersją formatów danych). Integracja ich jest jeszcze bardziej utrudniona, jeśli przetwarzane są na różnych komputerach lub pod różnymi systemami operacyjnymi lub też używają różnych systemów zarządzania bazami danych.

Zapewnienie komunikacji pomiędzy wieloma autonomicznymi aplikacjami wiąże się z dużymi nakładami pracy. Ocenia się, iż wówczas wielkość kodu źródłowego dedykowanego temu zadaniu może przekroczyć wielkość kodu źródłowego samych aplikacji. Szczególnie dotyczy to sytuacji, gdy nie stosuje się oprogramowania pośredniczącego (middleware) lub standardów dokumentów elektronicznych (np. opartych na XML i EDI).

Przewiduje się, że kłopoty integracyjne zaczną zanikać w aplikacjach następnych zaawansowanych generacji, wyposażanych od początku w międzynarodowe standardowe łącza integracyjne, pozwalające na dołączanie i odłączanie dowolnych aplikacji bez ingerencji w kody źródłowe i szybki przepływ danych nawet w środowisku najbardziej niejednorodnym.

Integracja aplikacji nie oznacza samoistnego usprawnienia systemu informacyjnego, mimo iż daje możliwości dostępu do wielu różnorodnych informacji i lepszego modelowania biznesu oraz polepsza ich jakość. Integracji musi towarzyszyć organizacja służb i procesów wykorzystująca szansę zorganizowania "naturalnego" przepływu informacji w skali całego banku, zgodnego z potrzebami chwili i profilem działania każdego podmiotu. Możliwość uzyskania "globalnych informacji" i zapewnienia źródła "jednej prawdy" daje podmiotom szansę na oderwanie się od partykularnych interesów komórek organizacyjnych, które reprezentują, na rzecz rozwoju biznesu całej firmy.