Moduł monitorowania usług

Ilona Bluemke[1,2], **Marcin Warda**[1]

Streszczenie: W pracy przedstawiono koncepcję modułu monitorowania usług w architekturze zorientowanej na usługi - SOA (Service Oriented Architecture). Implementację modułu wykonano w Instytucie Informatyki PW (Warda, 2006) dla platformy integracyjnej *webMethods*. Zrealizowany moduł monitorowania może być bardzo przydatny dla osób zarządzających zintegrowanym system. Pozwala na monitorowanie usług i przepływów realizowanych na wielu serwerach. Wskazano także na możliwości poprawienia jakości usług w oparciu o dane dostarczane przez moduł.

Słowa kluczowe: integracja systemów, SOA, jakość usług, monitorowanie usług,

1. Wstęp

Integrowanie systemów i aplikacji informatycznych jest koniecznością w dobie cywilizacji silnie wykorzystującej komputerowe systemy informacyjne. Współdziałanie systemów jest niezbędne do lepszej automatyzacji procesów biznesowych, do minimalizowania kosztów ponoszonych na rozwiązania informatyczne i do oferowania klientowi wszelkiego rodzaju usług realizowanych w wielu technologiach. Integrowanie systemów nie jest rzeczą nową, towarzyszy informatyce od początku pojawiania się pierwszych systemów wspomagających korporacje. Pomysły na integrowanie ewoluowały i zmieniały się szybko. Problematyce integracji systemów, architekturom i technikom intergacyjnym poświęconych jest wiele prac np. Alonso at al. 2004, McGovern at al. 2006, Hope at al.2003. Dzisiejsza koncepcja architektury zorientowanej usługowo - SOA (Alonso at al. 2004, Buhler at al. 2004, Dustdar at al. 2004, Hope at al. 2003, McGovern at al. 2006) z pewnością zostanie w przyszłości zastąpiona jeszcze lepszymi koncepcjami. Spojrzenie na dowolny system jako na dostawcę pewnej usługi informatycznej, pozwala zbudować wspólny poziom komunikacji między systemami i bez problemu wykorzystać ich funkcjonalności. Bardzo ważnym problemem jest jakość usług w architekturze zorientowanej usługowo, jak i wpływ architektury na jakość usług. Na jakość usług oferowanych przez platformę integracyjną ma wpływ bardzo wiele czynników, jak choćby architektura logiczna platformy, architektura fizyczna, systemy integrowane, implementacja przepływów, czy technologia. Określenie ograniczeń, wąskich gardeł, czy przyczyn niskiej jakości usługi nie jest rzeczą łatwą. Moduł monitorowania usług, zrealizowany w Instytucie Informatyki PW (Warda 2006) jest próbą stworzenia narzędzia pozwalającego

[1] Instytut Informatyki, Politechnika Warszawska, Nowowiejska 15/19, 00-665 Warszawa
e-mail:I.Bluemke@ii.pw.edu.pl

[2] Praca wykonana w ramach grantu Dziekana Wydziału Elektroniki i Technik Informacyjnych PW nr 503/G/1032/4000/000

prowadzić zaawansowane analizy zjawisk zachodzących na platformie integracyjnej. Różni sie on istotnie od innych narzędzi monitorowania. Implementację modułu zrealizowano dla typowego produktu integracyjnego tj. dla platformy integracyjnej *webMethods*. W platformie tej jest dostępny produkt *Optimize for Process* monitorujący procesy biznesowe. Produkt ten nie pozwala jednak na monitorowanie usług na poziomie szyny usług (ang. Enterprise Service Bus -ESB) , toteż zrealizowany moduł monitorowania może być bardzo przydatny dla osób zarządzających platformą integracyjną. Inne narzędzia monitorują usługi wykonywane na pojedynczym serwerze albo w przypadku realizacji rozproszonej dostarczają tylko listę niepowiązanych serwisów. Zrealizowany w Instytucie Informatyki PW moduł monitorowania dostarcza użytkownikowi pełen kontekst przepływu, nawet dla realizacji rozproszonej.

Podstawowym celem monitorowania usługi jest zapewnienie jej wysokiej jakości z punktu widzenia klienta oraz niskiego kosztu wykonania zarówno dla klienta, jak i dla dostawcy usługi. Na jakość usługi, rozpatrywaną w kategoriach architektury zorientowanej usługowo składają się dwa parametry: wynik operacji i czas wykonania operacji. Wynik - świadczy o skuteczności operacji. Wynik może być zwracany razem z danymi wyjściowymi usługi, jako kod błędu. Usługa może zwracać wiele kodów błędów świadczących o niepowodzeniu operacji, z których można wyróżnić grupę błędów wynikających z niepoprawnego działania usługi oraz grupę błędów powodowanych przez klienta usługi (np. wysłanie niepoprawnych danych, dla których nie można wykonać żądania). O ile błędy wynikające z niepoprawnego działania usługi wpływają ujemnie na jakość usługi, o tyle druga kategoria błędów jest z punktu widzenia dostawcy usługi poprawnym zachowaniem. Wyniki nieudanych operacji można dalej podzielić na błędy funkcjonalne oraz błędy techniczne, które ogólnie wynikają z awarii platformy (infrastruktury integracyjnej, czy systemów biorących udział w przepływie), na której wykonywana jest dana usługa. Odpowiedzialność za błędy techniczne spoczywa na administratorach platformy usługowej. Należy dodatkowo zaznaczyć, że awaria elementu infrastruktury może objawić się dwojako z punktu widzenia klienta usługi. Zazwyczaj awaria powoduje podniesienie wyjątku w danej warstwie oprogramowania. Jeśli więc wyjątek zostanie przechwycony wewnątrz usługi, klient dostanie standardową odpowiedź, zawierającą opis błędu. Natomiast, jeśli np. cała platforma usługowa będzie nieczynna (np. z powodu braku zasilania), wtedy wyjątek zostanie przechwycony przez klienta usługi. Z punktu widzenia klienta usługi, oba scenariusze świadczą o niedostępności usługi. Natomiast należy brać pod uwagę fakt, że druga sytuacja nie zostanie wykryta przez monitorowanie usług. W przypadku, gdy żadna usługa nie jest wykonywana, to nie ma wyników świadczących o problemie. W takiej sytuacji, konieczne są informacje z monitoringu technicznego platformy.

Drugim, obok wyniku operacji parametrem, od którego zależy jakość usługi jest czas wykonania. Czas wykonania przepływu, jest to w najprostszym scenariuszu czas mierzony od otrzymania zlecenia od klienta do momentu wysłania odpowiedzi do klienta. Taki pomiar może być przeprowadzony wyłącznie dla usług synchronicznych, gdy klient oczekuje na odpowiedź o wykonaniu operacji. W przypadku zleceń asynchronicznych czas operacji nie jest zwykle krytyczny z punktu widzenia

klienta, ale jego monitorowanie umożliwia śledzenie zajętości zasobów platformy usług. Czas wykonania w przypadku platformy usługowej jest ściśle związany z kosztem wykonania usługi. Wykonywanie operacji na platformie integracyjnej wymaga procesora i pamięci wykorzystywanych przez wątek serwera wykonujący usługę. Można przyjąć, że jedna usługa jest wykonywana w jednym wątku serwera. Dodatkowo w przypadku użycia w przepływie systemu kolejkującego, zużywane są zasoby tego systemu. Jeszcze jeden zasób, zajmowany w przepływie, to zasób integrowanego systemu, biorący udział w przepływie, a więc także zajęte połączenie do tego systemu w puli utrzymywanej na platformie integracyjnej. Mając informację o czasie trwania danego przepływu w rozbiciu na wewnętrzne serwisy adapterowe można oszacować ile zasobów dany przepływ zużywa. Dołączenie do tej informacji danych o ilości wywołań przepływów i ich rozkładu w czasie, daje dane niezbędne do skalowania środowiska.

Czas oraz wynik wykonania usługi są parametrami, którymi zainteresowany jest sam klient usługi. Dodatkowo zbieranie informacji o każdym przepływie realizowanym na platformie integracyjnej jest źródłem danych niezbędnym dostawcy usługi do prowadzenia analiz środowiska np. analizy pojemności i obciążenia platformy, wykrywanie wąskich gardeł architektury, błędnie zaprojektowanych przepływów, problemów w oprogramowaniu i infrastrukturze platformy integracyjnej.
W niniejszej pracy rozpatrywano monitorowanie usługi z punktu widzenia klienta, który patrzy na usługę jak na *czarną skrzynkę* realizującą daną funkcjonalność. W rzeczywistości taka usługa realizowana jest zwykle jako bardziej skomplikowany przepływ, który odpytuje różne zasoby, wykonuje transformacje danych, kolejkuje wykonywane operacje, itd. Rzeczywiste architektury integracyjne są zwykle dość skomplikowane i rozbudowane, a w szczególności, praktycznie zawsze są architekturami rozproszonymi. Rozproszenie oznacza, że różne usługi realizowane są na fizycznie różnych serwerach, komunikujących się między sobą. Usługa z punktu widzenia klienta, odwzorowuje się w rzeczywistości na przepływ składający się z wywołań wielu serwisów na różnych fizycznych serwerach i zasobów systemów zewnętrznych. W związku z tym, ważnym wymaganiem dla modułu monitorowania usług na platformie integracyjnej jest gromadzenie danych pozwalających na analizę ruchu między logicznymi i fizycznymi elementami platformy. Podobnie, kolekcjonowanie informacji o fizycznym serwerze, na którym wykonywana jest dana usługa pozwala na porównywanie czasów wykonania dla tej samej usługi na różnych serwerach. Dzięki temu możliwe jest wykrywanie serwerów, które są wolniejsze od innych i które mogą wpływać niekorzystnie na jakość usługi z punktu widzenia klienta.

W kolejnej sekcji pracy przedstawiono, w oparciu o przykład przepływów na platformie integracyjnej, zagadnienie jakości usług oraz podstawy monitorowania usług. Następna sekcja zawiera wybrane wymagania funkcjonalne i niefunkcjonalne modułu monitorowania usług oraz pokazuje czynności optymalizacyjne w oparciu o dane dostarczane przez ten moduł. Następnie pokazano zarys implementacji modułu monitorowania.

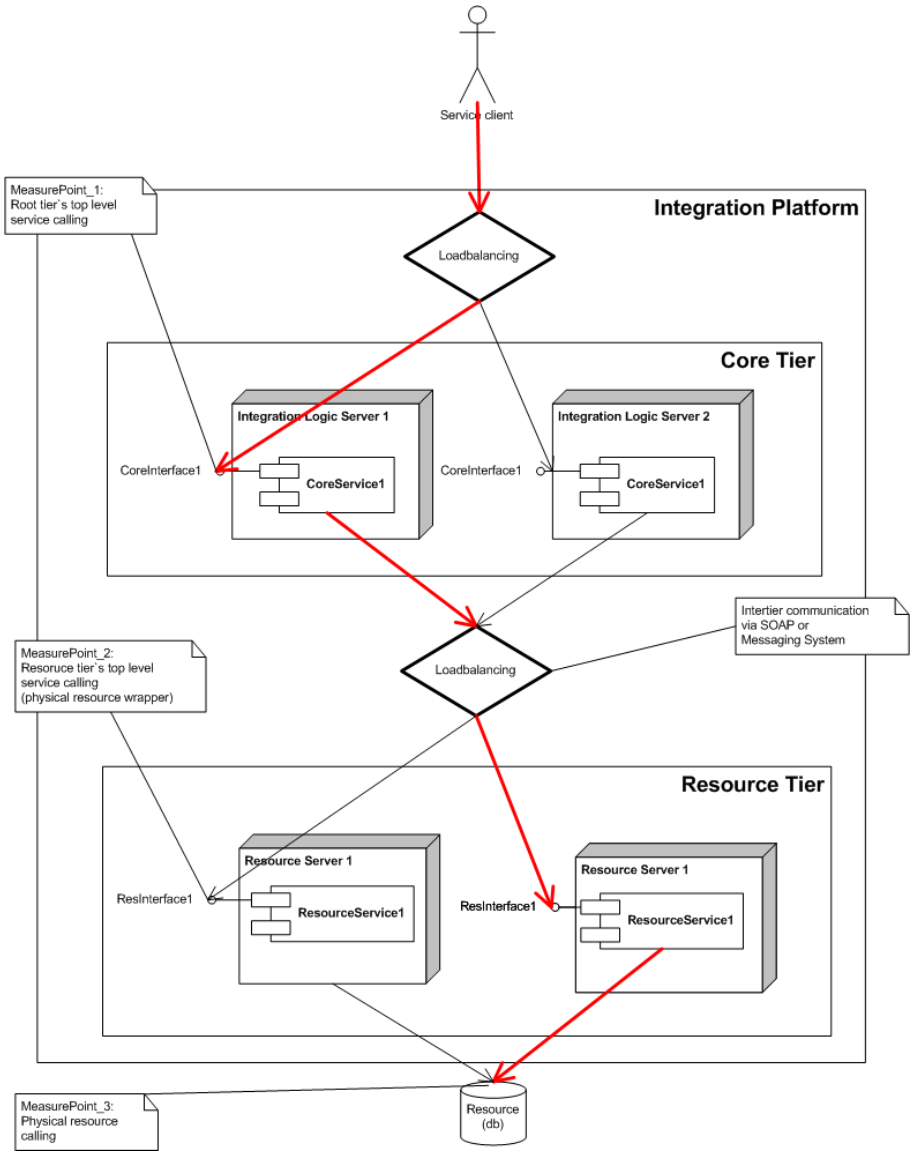
2. Analiza przepływu na platformie integracyjnej

Rysunek 1 przedstawia scenariusz bardzo prostego użycia usługi realizowanej na platformie integracyjnej. Platforma integracyjna ma w tym przypadku prostą architekturę. Fizyczne serwery integracyjne zgrupowane są w dwóch logicznych warstwach: *Core Tier* - na których realizowana jest walidacja i transformacja danych, routing do odpowiedniej usługi, realizującej dostęp do zasobów oraz *Resource Tier* - serwery posiadające połączenia z zasobami zewnętrznymi (w tym przypadku jest to pojedyncza baza danych). Komunikacja między warstwami logicznymi platformy może odbywać się poprzez system kolejkujący (ang. messaging system) lub bezpośrednio protokołem SOAP. Każda warstwa posiada po dwa serwery fizyczne, a ruch między warstwami jest równomiernie rozkładany poprzez moduł wyrównujacy obciążenia (ang. loadbalancer). Zaprojektowane rozwiązanie zapewnia dostępność usługi zawsze, gdy chociaż jeden serwer z danej warstwy jest dostępny (oczywiście przy założeniu, że generowany przez klienta ruch nie jest większy od pojemności pojedynczego serwera) oraz równomierny rozkład ruchu pomiędzy fizycznymi serwerami wchodzącymi w skład warstwy. Serwery warstwy zasobów łączą się z zewnętrzną bazą danych, która jest źródłem danych w przepływie. Natomiast klient łączy się wyłącznie z serwerami warstwy *Core*, również za pośrednictwem modułu wyrównywania obciążeń.

Dwie podstawowe informacje świadczące o jakości usługi z punktu widzenia klienta to czas trwania całej operacji oraz wynik zwrócony z serwisu *CoreService1*. Na rysunku 1 pokazano jak wiele jest elementów uczestniczących w przepływie i będących potencjalnym źródłem niskiej wydajności, bądź błędnego działania usługi. Zapisywanie wyłącznie wyniku i czasu trwania serwisu *CoreService1* nie pozwoli na zlokalizowanie dokładnej przyczyny problemów realizacji usługi. Możliwe będzie określenie czy poziom jakości został dotrzymany, ale jeśli nie, to nie będzie wiadomo z jakiego powodu. Moduł monitorowania usług powinien mierzyć parametry przepływów w takich punktach, aby możliwa była dekompozycja czasu i wyniku wykonania usługi widzianej przez klienta na czasy wykonania poszczególnych serwisów na poszczególnych elementach platformy uczestniczących w przepływie. Może się zdarzyć również tak, że czasy wykonania danego serwisu, różnią się znacząco dla poszczególnych wywołań, mimo że wykonują takie same operacje (również wołają inne serwisy które trwają tyle samo czasu). Dysponowanie informacją, na którym fizycznym serwerze wykonywał się dany serwis pozwoli zweryfikować, czy źródłem problemu nie jest sam serwer.

Powyższy przykład jest bardzo prosty i założono w nim, że serwis *CoreService1* wywołuje wyłącznie serwis *ResourceService1*. W rzeczywistości nawet w tak prostym przepływie wykorzystywany jest cały szereg innych usług a gromadzenie informacji o wszystkich wywoływanych serwisach nie ma sensu i jest kosztowne toteż pożądaną cechą modułu monitorowania jest możliwość konfigurowania serwisów, dla których informacje bedą zapisywane.

Znaczna część operacji realizowanych w warstwie integracyjnej realizowana jest w sposób asynchroniczny. Monitorowanie opiera się w takim przypadku, podobnie jak dla wywołań synchronicznych, na zapisywaniu danych o wywołaniach poszczególnych serwisów. W celu analizowania całych przepływów, a nie tylko poszczególnych



Rysunek 1. Proces realizacji przepływu na platformie integracyjnej

serwisów, dzienniki są indeksowane unikalnym kluczem. O ile analiza wywołań synchronicznych będzie prosta (wywołania kolejnych serwisów będą sekwencyjne) elementy asynchroniczne mogą rozdzielać się na wiele niezależnych przepływów, z których każdy należy rozpatrywać indywidualnie (taki przypadek ma miejsce np. dla przepływów opartych na wzorcu *publish/subscribe*, w którym dokument publikowany przez jedno źródło, jest konsumowany przez wiele systemów.

3. Wymagania modułu monitorowania

Niniejsza sekcja zawiera opis wybranych, ważniejszych wymagań funkcjonalnych i niefunkcjonalnych, przyjętych jako podstawy implementacji modułu monitorowania. Dla lepszej przejrzystości wymagania funkcjonalne zostały dodatkowo podzielone na trzy grupy: wymagania dotyczące gromadzenia danych, wymagania określajace możliwości konfiguracyjne modułu oraz wymagania dla raportów opracowywanych na podstawie danych zebranych przez moduł monitorowania.

Wymagania, dotyczące danych gromadzonych przez moduł:

1. Zapisywanie danych odbywa się na poziomie wywołania pojedynczego, wybranego zgodnie z konfiguracją, serwisu wchodzącego w skład większego przepływu.
2. Dzienniki wywołań poszczególnych serwisów maja wspólny klucz, unikatowy na poziomie pojedynczego przepływu.
3. Ważniejsze informacje zebrane podczas wywołania pojedynczego serwisu:
   - Nazwa serwisu, którego dotyczą zebrane parametr,
   - Kod błędu z wywołania serwisu, komunikat przechwyconego wyjątku, jeśli wywołanie serwisu zakończy się wyjątkiem,
   - Czas rozpoczęcia i czas zakończenia wykonywania serwisu,
   - Nazwa serwera i nazwa użytkownika, który rozpoczął przepływ,
   - Różnorodne znaczniki potrzebne w module (np. typ wywołania, serwisu),
   - Dodatkowe identyfikatory np. poprzedniego serwera fizycznego i parametry.
4. Zapewnienie mechanizmu archiwizowania dzienników.

Wymagania, dotyczace konfiguracji modułu

1. Konfiguracja modułu monitorowania będzie realizowana poprzez odpowiednie wpisy w bazie danych, wspólnej dla wszystkich serwerów na platformie integracyjnej. Konfiguracja będzie ładowana do pamięci serwera, przy ładowaniu pakietów z serwisami, poprzez serwis oznaczony znacznikiem `autostart`. Możliwe będzie przeładowanie konfiguracji na serwerze, bez konieczności restartu serwera integracyjnego.

2. Filtrowanie serwisów, dla których mają być zapisywane dzienniki, będzie określane na poziomie: nazwa serwisu (wyrażenie regularne), nazwa użytkownika wywołującego przepływ, nazwa serwisu (wyrażenie regularne) najwyższego poziomu dla całego przepływu (wywołanego bezpośrednio przez użytkownika), nazwa serwisu (wyrażenie regularne) najwyższego poziomu warstwy wywołującej, znacznik, określający czy zapisywanie dzienika serwisu ma następować wyłącznie w przypadku, gdy serwis wywoływany jest jako serwis najwyższego poziomu (na danym serwerze).

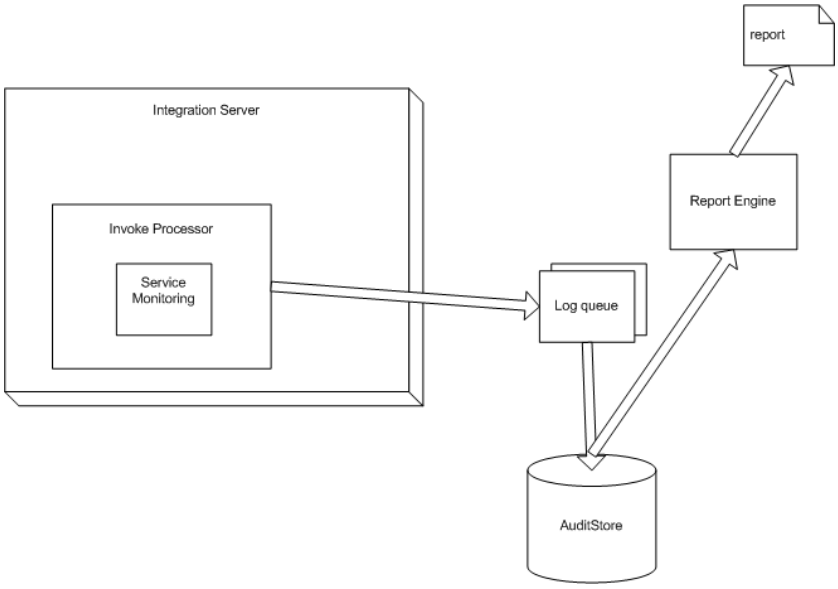
3. Dodatkowo filtrowanie będzie się odbywało na poziomie pakietów *webMethods*.

4. Na poziomie serwisu możliwe będzie określenie wyjściowych pól serwisu, które będą dodatkowo zapisane razem z dziennikiem wywołania serwisu.

Monitorowanie może być używane do bieżącego wykrywania awarii oraz do długoterminowego analizowania i optymalizowania przepływów. Cykl optymalizowania związany jest ze zmianą konfiguracji modułu monitorowania i wygląda następująco:

1. W stanie wyjściowym, zapisywanie dzienników jest włączone na wszystkich serwisach wołanych na najwyższym poziomie. Z zapisywanych danych można określić pierwszy serwis najwyższego poziomu, czyli serwis wywoływany przez klienta. Dzięki identyfikatorom można określić, jaka była fizyczna ścieżka przepływu (lista serwerów, na których wykonywany był przepływ). Jeśli serwery fizyczne zgrupowane są w warstwy logiczne, to mając mapowanie serwerów fizycznych na warstwy można określić ścieżkę przepływu poprzez logiczne warstwy serwerów. Na tym poziomie możliwe jest określenie przepływów zaprojektowanych niewłaściwie np. wielokrotne odpytania tego samego serwisu między warstwami, albo wywołania między warstwami, które są zabronione. Identyfikatory przepływów pozwalają na określenie zależności ilościowych dla serwisów wywoływanych na platformie. Zbieranie informacji o czasie wykonania poszczególnych serwisów najwyższego poziomu pozwala na określenie najwolniejszych elementów przepływu na poziomie przepływów realizowanych przez dany serwer, czy warstwę logiczną serwerów.

2. Informacje o liczbie serwisów na danym serwerze (czy warstwie serwerów) i sumaryczny czas wykonania tych serwisów, pozwalają określić najkosztowniejsze wywołania (sumaryczny czas przekłada się na liczbę zajętych zasobów serwera w postaci wątków, czy zajętych połączeń do innych systemów). Takie przepływy są kandydatem do optymalizacji. Innym kandydatem do optymalizacji są przepływy, które mimo że nie są bardzo kosztowne dla samej platformy integracyjnej, to trwają nieakceptowalnie długo z punktu widzenia klienta. Po wybraniu przepływu, który będzie optymalizowany, możliwe będzie rozszerzenie filtra tylko dla tych serwisów niskiego poziomu, które są podejrzane o nieoptymalność.



Rysunek 2. Koncepcja modułu monitorowania

Przykładowe analizy wykonywane poprzez moduł monitorowania:

- Histogram przepływów zakończonych danym kodem błędów.
- Statystyki czasu wykonania oraz liczby wywołań przepływów.
- Statystyka kosztów przepływów - koszt liczony jest jako sumaryczny czas wykonania danego typu serwisów w danej jednostce czasu.
- Charakterystyka ruchu i obciążenia na platformie integracyjnej w czasie.
- Histogram czasów wykonania ustalonego serwisu w rozbiciu na fizyczne serwery.
- Statystyka przepływów (ruch między-warstwowy na platformie integracyjnej).
- Rozbicie czasu wykonania przepływu na czasy poszczególnych serwisów.

Wybrane wymagania niefunkcjonalne dla modułu monitorowania usług to:

- Spełnienie ograniczeń wydajnościowych.
- Kontrolowanie zasobów serwera (pamięć, wątki) zajmowanych przez obsługę dzienników.

4. Realizacja monitorowania usług

Moduł monitorowania ruchu oraz jakości usług na platformie integracyjnej składa się z elementów odpowiedzialnych za następujące procesy:

- Zbieranie informacji o konkretnym wywołaniu oparte na mechanizmie *pre-/post invoke* oraz kolejkowania ich w pamięci lokalnego serwera.
- Przetwarzania kolejki na lokalnym serwerze, które polega na pobraniu kolejki dzienników z pamięci, i opcjonalnie: zapisaniu ich do pliku na lokalnym systemie plików, zapisaniu do bazy danych z połączenia utrzymywanego przez lokalny serwer lub wysłania listy dzienników do centralnego serwisu wystawionego na platformie integracyjnej, który zapisze dzienniki do bazy danych.
- Przetwarzanie zapisanych do bazy danych dzienników, w skład którego wchodzi wyliczanie statystyk, czyszczenie historycznych dzienników, itp.

Ogólna koncepcja realizacji modułu jest przedstawiona na rysunku 2.

4.1. Zbieranie informacji o monitorowanych usługach

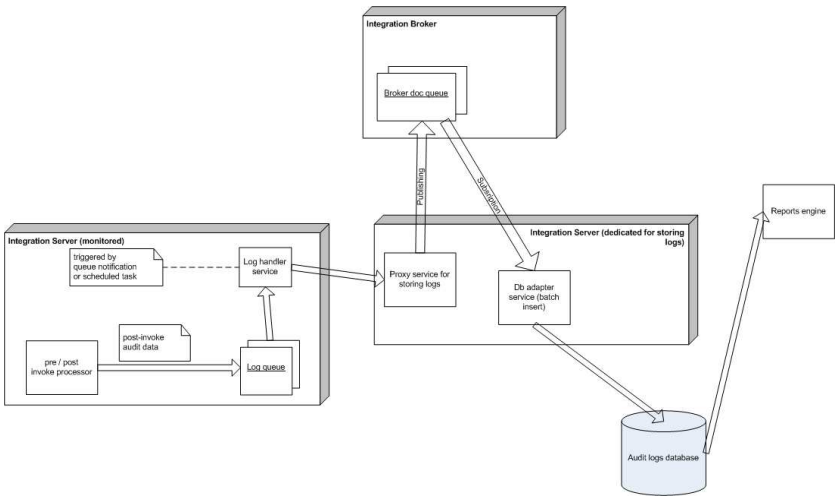
Informacje o wywołaniu konkretnego serwisu zbierane są po jego wywołaniu. Dodatkowo przed wywołaniem serwisu inicjowane są parametry niezbędne do zebrania statystyk (np. czas rozpoczęcia wywołania). Do obsługi zapisu informacji przed i po wywołaniu użyty został mechanizm dostarczany razem z serwerem integracyjnym `custom pre/post invoke` pozwalający na dołączanie własnych klas obsługujących wywołanie `com.wm.app.b2b.server.invoke.InvokeChainProcessor`. Zbieranie danych jest realizowane synchronicznie względem wywołania samego serwisu. Oznacza to, że całkowity czas wykonywania serwisu zostanie wydłużony o czas obsługi zapisu. Takie rozwiązanie wbrew pozorom jest znacznie bardziej wydajne, niż zbieranie statystyk serwisem wywołanym asynchronicznie. Wywołanie asynchroniczne wymagałoby wykonania pełnej kopii obiektów zawierających statystyki oraz danych przekazywanych z/do serwisu. Wykonanie takiej kopii dla typowej wielkości potoku (obiekt zawierający dane przekazywane między wywołaniami serwisów wewnątrz serwera integracyjnego) z użyciem API

webMethods - metoda `IDataUtil.deepClone()`

może trwać nawet pojedyncze milisekundy, co jest zbyt kosztowne przy przyjętych założeniach implementacyjnych. Zebrane parametry serwisu gromadzone są w obiekcie klasy

`mwr.service.performanceMonitor.beans.AuditData`

który wstawiany jest do kolejki dzienników w pamięci operacyjnej danego serwera integracyjnego. W celu zapewnienia możliwości dekompozycji czasu wykonania danego przepływu na czasy wykonania poszczególnych kroków serwisu, konieczne jest zapisanie kontekstu wywołania konkretnego serwisu. Z uwagi na to, że przepływ może być realizowany na więcej niż jednym fizycznym serwerze, wymagane jest przekazywanie kontekstu wywołania wraz z wywołaniem serwisu na innym



Rysunek 3. Diagram wdrożenia modułu monitorowania

serwerze. Przekazywanie tego kontekstu obsługiwane jest również za pomocą mechanizmu *pre-invoke*, który wykrywa obsługę serwisu odwołującego się do zewnętrznego serwera (czyli np. serwisu odpowiedzialnego za wysłanie komunikatu do *Brokera*, czy też wywołanie SOAP) i do wysyłanego żądania dołącza w odpowiedni sposób kontekst przepływu. W skład tego kontekstu wchodzi: nazwa użytkownika inicjującego przepływ, nazwa serwisu najwyższego poziomu oraz identyfikator dla całego przepływu, nazwa serwisu najwyższego poziomu oraz identyfikator dla serwera warstwy poprzedniej. Identyfikatory transakcji, jako elementu przepływu wykonywanego w całości na jednym serwerze pozwalają na śledzenie całego przepływu na kolejnych fizycznych serwerach (czy też warstwach serwerów). Nazwa użytkownika inicjującego przepływ oraz serwisy top-level warstwy wyższej i całego przepływu, przekazywane są po to, aby przy wywołaniu dowolnego serwisu, można było określić reguły filtrowania dzienników dla danego serwisu.

4.2. Składowanie zebranych informacji

Dziennik z danymi zebranymi podczas wywołania serwisu jest wstawiany do kolejki opartej na tablicy obiektów. Zapis do kolejki jest synchronizowany, ponieważ może być wykonywany z wielu wątków serwera równocześnie. W momencie, gdy zajętość kolejki osiąga określony poziom, wywoływany jest asynchronicznie serwis obsługujący zebrane dzienniki. Serwis obsługujący kolejkę, pobiera obiekty, a następnie czyści kolejkę i wykonuje zapis do bazy. Obiekt kolejki obsługuje ponadto licznik aktualnie wykonywanych serwisów obsługujących zapisy. Licznik jest zwiększany przy pobraniu zapisów z kolejki i zmniejszany przy zakończeniu wykonywaniu serwisu obsługującego. Licznik pozwala ograniczać liczbę zasobów serwera (wątków) równolegle obsługujących dzienniki. Wywołanie serwisu zapisującego dzienniki następuje przy przekroczeniu zdefiniowanego progu zajętości kolejki oraz za każdym

razem, kiedy wielkość kolejki przekroczy ustalony próg. Wszystkie parametry kolejki są konfigurowalne.

Serwis obsługujący kolejkę dzienników odpowiedzialny jest za zapisanie ich do bazy danych. Przy dużej liczbie zapisywanych dzienników, obciążenie związane z zapisywaniem do bazy danych mogłoby wpływać na wydajność pozostałych przepływów realizowanych na serwerze. Ponadto zapisywanie dzienników do bazy danych bezpośrednio z serwera wymaga utrzymywania połączeń do bazy danych dzienników na każdym serwerze, co w przypadku rozbudowanej platformy integracyjnej, może stanowić problem dla samej bazy danych. Dlatego zdecydowano się na zbudowanie usługi działającej na dedykowanym serwerze integracyjnym, która będzie centralnym punktem zapisywania dzienników. Usługa będzie standardowo udostępniona jako webService. Centralny mechanizm zapisywania dzienników składa się z dwóch części: serwisów wysyłających dzienniki do *Brokera* komunikatów asynchronicznych oraz serwisu odbierającego dzienniki z *Brokera* i zapisującego je do bazy danych. Wysyłanie do *Brokera* zapewnia kolejkowanie dzienników i pozwala na uniezależnienie od chwilowych niedostępności bazy danych dzienników, oraz problemów wydajnościowych przy zapisywaniu dzienników. Serwis obsługujący komunikację z *Brokerem* i zapisujący dzienniki do bazy danych powinien być zainstalowany na dedykowanym serwerze (serwerach) integracyjnym połączonym z dedykowanym *Brokerem* tak, aby obsługa dzienników nie wpływała na wydajność pozostałych przepływów realizowanych na platformie. Rysunek 3 prezentuje ogólny zarys przyjętej koncepcji realizacji modułu. Od momentu zapisania dzienników do kolejki na *Brokerze*, można uznać, że są one zabezpieczone i trafią do bazy danych dzienników po czasie zależnym od wydajności zapisu do bazy oraz od liczby dzienników generowanych na platformie integracyjnej. Zapis do relacyjnej bazy danych realizowany jest poprzez serwis adapterowy JDBC. Jako metoda zapisu paczki dzienników do bazy został wybrany interfejs `batchInsert`. Takie wstawianie, według przeprowadzonego (Warda 2006) eksperymentu, jest ok. 10-krotnie szybsze, niż wstawianie za pomocą `Insert` w pętli kolejnych rekordów. Dzienniki zapisywane są w relacyjnej bazie danych w strukturze dwóch tabel: informacja o wywołaniu serwisu oraz dodatkowe dane zebrane z wyjścia serwisu. Schemat bazy danych zawiera dodatkowo tabele konfiguracyjne: filtrowanie dzienników dla określonych serwisów, definicje pól dokumentu wyjściowego serwisu, które są zapisywane dla wywołania danego serwisu.

5. Podsumowanie

Zrealizowany moduł monitorowania usług jest narzędziem pozwalającym prowadzić zaawansowane analizy zjawisk panujących na platformie integracyjnej. Moduł został przygotowany do zastosowania w środowisku integracyjnym *webMethods* i różni się istotnie od innych narzędzi monitorowania. Istnieją narzędzia monitorujące procesy biznesowe ale nie pozwalają one na monitorowanie usług na poziomie szyny usług (ang. Enterprise Service Bus). Inne narzędzia monitorują usługi wykonywane tylko na pojedynczym serwerze albo w przypadku realizacji rozproszonej dostarczają jedynie listę niepowiązanych ze sobą serwisów. Zrealizowany

w Instytucie Informatyki PW moduł monitorowania dostarcza użytkownikowi pełen kontekst przepływu, nawet dla realizacji rozproszonej, toteż może być bardzo przydatny dla osób zarządzających czy administrujących platformą integracyjną. Monitorowanie wiąże się oczywiście z narzutem czasowym. Wykonane eksperymenty (Warda 2006) pokazują, że dla przepływu składającego się ze 100 wywołań serwisów narzut wynosił od 12 do 31 ms co jest wartością akceptowalną i niższą niż narzuty oprogramowania dostarczanego przez producenta platformy integracyjnej i producenta serwera.

Analiza wyników zebranych przez zaprojektowany moduł monitorowania pozwala na skalowanie i optymalizowanie architektury integracyjnej oraz na optymalizowanie przepływów w tej architekturze. Zebrane wyniki mogłyby również posłużyć jako element decyzyjny w zakresie sterowania ruchem na platformie, w celu zapewnienia wysokiej jakości usług dla przepływów realizowanych na platformie.

Literatura

Alonso G., Casati F., Kuno H., Machiraju V. (2004) *Web Services: Concepts, Architectures and Applications*. Springer Verlag.

Buhler A. P., Starr C., Schroder W. H., Vidal J. M. (2004) *Preparing for Service Oriented Computing: a composite design pattern for stubless Web Service invocation*. U.S. National Science Foundation, USA.

Dustdar S., Haslinger S. (2005) *Testing of Service Oriented Architectures - A Practical Approach*. Distributed Systems Group. Viena Univerisity of Technology. Austria.

Hohpe G., Woolf B. (2003) *Enterprise Integration Patterns*. Addison-Wesley.

McGovern J., Sims O., Jain A., Little M. (2006) *Enterprise Service Oriented Architectures*. Springer Verlag.

Warda, M. (2006) *Architektury Integracyjne Systemów Informacyjnych*. Praca dyplomowa magisterska. Instytut Informatyki PW.

http://www.webmethods.com (2007).

http://www.w3.org/TR/soap/ (2007).

http://www-306.ibm.com/software/integration (2007).

A tool for monitoring services

The design of a module for monitoring services in enterprise application integration is presented. This module was implemented in the Institute of Computer Science Warsaw University of Technology for integration platform *webMethods*. This tool can be used to improve the quality of services in an integrated enterprise architecture.