ТЕХНОЛОГИИ РАСПРЕДЕЛЕННОГО ВЗАИМОДЕЙСТВИЯ В КОРПОРАТИВНОМ ИНФОРМАЦИОННОМ УПРАВЛЕНИИ

Ю.В. Бородаенко, В.А. Вишняков
Минский институт управления, г. Минск, Беларусь
jborodaenko@mail.ru

Передача файлов (file transfer) представляет собой один из подходов к обеспечению обмена данными между приложениями, не требующий привлечения внешних инструментов или интеграционных пакетов. Реализация интеграционного решения на основе передачи файлов связана с определенными соглашениями между приложениями, касающимися имени файла и каталога, механизмов обновления файла, блокировки доступа к файлам и т.д. Преимуществом технологии передачи файлов является независимость взаимодействующих приложений друг от друга, недостатками данной технологии являются проблемы совместимости и синхронизации данных.

Интеграция приложений на основе разделяемых БД (shared database) обеспечивает целостность данных на основе системы управления транзакциями и совместимость форматов данных, поскольку все приложения используют универсальный язык SQL. Недостатком данного подхода к интеграции приложений является отсутствие инкапсуляции данных, что приводит к негибкости информационной инфраструктуры и ее слабой адаптируемости к изменениям в бизнесе.

Подход к интеграции приложений на основе удаленного вызова процедур RPC (Remote Procedure Call) обеспечивает принцип инкапсуляции данных. Ряд технологий, таких, как CORBA (Common Object Request Architecture),DCOM (Distributed Component Object Model),Java RMI (Remote Method Invocation), .NET Remoting, используют данный подход, но ни одна из перечисленных технологий не смогла стать универсальной, поскольку определяла свои собственные форматы данных, протоколы обмена данными и программные интерфейсы. К тому же, объектные технологии не решают проблему зависимости одного приложения от другого в процессе их взаимодействия. Например, необходимость выполнения операций в строгой последовательности ограничивает возможности вносить изменения в одно приложение без соответствующих изменений в другом приложении. Данные аспекты интеграции, выраженные в требовании слабой связанности взаимодействующих систем, способствовали развитию технологий на основе обмена документами в формате XML.

Особенности сервис-ориентированной архитектуры SOA. Интеграция приложений на основе сервис-ориентированной архитектуры SOA (Service Oriented Architecture) и Web-сервисов использует XML для обмена сообщениями и направлена на информационную поддержку производственных процессов. Архитектурный стиль SOA способствует построению корпоративных бизнес-решений, обладающих способностью расширять либо изменять функциональность по требованию используемыми сервисами с описанными, широкодоступными и стандартизованными интерфейсами.

Одним из ключевых принципов SOA, определяющим ее отличие от традиционных архитектур, является выделение единиц логики обработки (функций, методов) приложения в автономные программные модули – сервисы, не зависящие друг от друга. Данный принцип способствует тому, что алгоритмы и данные не являются больше частью локальной инфрастуктуры приложения, а расположены вне его, что существенно повышает гибкость приложений и возможности многократного использования его функций. При интеграции *композитное* приложение имеет свой собственный пользовательский интерфейс, но уровень бизнес-логики формируется за счет интеграции функций, предоставляемых *компонентными* приложениями.

Основными характеристиками SOA являются следующие: *распределенность* – функциональные элементы приложений могут быть расположены в различных вычислительных системах, и взаимодействовать между собой через локальные сети и Интернет; возможность *динамического поиска* и подключения нужных функциональных модулей; ориентация на *бизнес-процессы* – система строится с расчетом решения определенных задач. В терминах SOA, бизнес-процесс состоит из совокупностей операций, которые исполняются в заданном порядке согласно заданному набору бизнес-правил; упорядочение, отбор и исполнение операций называется *хореографией* сервиса.

Принципы преобразования функций приложения в Web-сервисы следующие:

1. Выделение функций в Web-сервисы (принцип автономности).
2. Отделение реализации сервиса от интерфейса сервиса (принцип абстракции).
3. Пакетный режим обработки данных посредством крупномодульного (coarse-grained) интерфейса, обеспечивающего однократный обмен документами между провайдером сервиса и клиентом вместо многократных вызовов процедур API.
4. Независимость от состояния сервиса: поведение провайдера сервиса не зависит от предыдущих обращений клиентов к данному сервису (statelessness).
5. Описание интерфейса на основе стандарта WSDL (Web Services Description Language),использующего синтаксис XML.
6. Возможность отыскания нужного сервиса в каталоге сервисов UDDI (Universal Description, Discovery, and Integration).
7. Взаимодействие сервисов на основе протоколов SOAP (Simple Object Access Protocol) или XML/RPC.

Принципы *автономности* и *абстракции* унаследованы из теории ООП, при этом принцип наследования не используется в SOA с целью исключения взаимных зависимостей между сервисами. Принцип *пакетной обработки данных* обусловлен тем, что при использовании Интернет в качестве среды взаимодействия приложений стало целесообразным объединять данные, запрашиваемые у сервера, в один пакет, вместо многократных вызовов методов объекта. При однократном обращении к провайдеру сервиса состояние о клиенте не сохраняется, что обеспечивает слабую связанность взаимодействующих приложений.

Этапы внедрения сервис-ориентированной архитектуры SOA на предприятии: 1 – разработка Web-сервисов для приложений; 2 – разработка Web-сервисов для производственных процессов; 3 – обеспечение технической инфраструктуры сервисов; 4 – разработка Web-сервисов для взаимодействия с внешними бизнес-партнерами.

Первые два этапа связаны с разработкой сервисов, затрагивающих функции приложения, и сервисов, объединяющих функции в бизнес-процессы. Третий этап связан с необходимостью использования технической инфраструктуры Web-сервисов, осуществляющей обеспечение безопасности и надежности данных и управление вызовами приложений (сервисов) в определенной последовательности (функция оркестровки). К технической инфрастуктуре, выполняющей указанные функции, можно отнести продукты класса EAI (Enterprise Application Integration) и ESB (Enterprise Service Bus). Интеграционное решение EAI построено на централизованных принципах, что означает, что все функции промежуточной обработки сообщений выполняются единым интеграционным сервером. Среди преимуществ EAI выделяется эффективность администрирования корпоративной системы, среди недостатков – высокая стоимость приобретения (от 50 тыс. долларов США), не позволяющая предприятиям малого бизнеса инвестировать в интеграционное решение данного класса.

На заключительном этапе развития SOA сервисный подход распространяется на взаимодействие с бизнес-партнерами B2B, что позволяет использовать стандартные модели электронного бизнеса ebXML и RosettaNet. Интеграция внутренних и внешних бизнес-процессов обеспечивает возможность производства по требованию (on demand business), являющуюся важным конкурентным преимуществом в современных условиях.