УДК 681.3

В. Дерецкий

РАЗРАБОТКА ПРИЛОЖЕНИЙ В СЕРВИС-ОРИЕНТИРОВАННОЙ АРХИТЕКТУРЕ СЕМАНТИЧЕСКОГО ВЕБ

Семантическая сервис-ориентированная архитектура основана на принципах сервисной ориентации, семантического моделирования, интеллектуальной и автоматизированной интеграции, определяет основу для новой технологии, которая обеспечивает новые средства интеграции сервисов, большую адаптивность к изменениям бизнес-требований и сред выполнения. Архитектура определяется, начиная от абстрактной бизнес-модели семантического сервисного приложения, завершая сервисами, процессами, технологиями и в соответствии с онтологией моделирования Веб-сервисов (WSMO).

Введение

Для того, чтобы отвечать требованиям бизнеса, гибкости и динамики происходящих процессов, традиционные монолитные приложения требуют замены на более мелкие единицы функциональности, которые можно комбинировать и собирать в сложные программные конструкции. Чтобы соответствовать этой парадигме информационным системам необходимо обеспечивать повторную конфигурацию для создания новых приложений, развиваемых как сервисы и наследуемые системы. Движением в этом направлении является развитие парадигмы сервис-ориентированной архитектуры (SOA) с целью разрешения проблем динамики среды и адаптивности бизнес-процессов. SOA строится как уровневая модель сервисов, которая согласовывается с принципами свободного связывания сервисов, повторного использования, подающихся обнаружению и композиции.

Идеи и существующие решения SOA, направленные на решение задач адаптивности бизнес-требований, являются трудными для измерения без надлежащей степени автоматизации. При сегодняшних сервисных технологиях таких как: WSDL, SOAP, UDDI и BPEL, которые реализуют новые возможности SOA, обеспечивая частичную совместимость с помощью унификации технологического окружения, в котором осуществляется согласование на уровне контента и процессов для каждого конкретного случая [1, 2].

Гибкость и расширяемость XML может определить только структуру и синтаксис данных. Без понятной для машины семантики сервисы могут быть размещены и связаны только во время их проектирования, что в свою очередь ограничивает возможности автоматизации. Для преодоления этих недостатков предлагается расширение SOA семантикой, которая обеспечивает масштабируемую интеграцию и адаптацию к изменениям, происходящих на этапах жизненного цикла программной системы. Семантика позволит определить развитые формальные модели, в которых определяются возможности сервисов и требования их потенциальных потребителей. Семантические данные, посредством которых осуществляется обмен между бизнес партнерами, могут быть однозначно описаны в форме и в терминах онтологий. С помощью семантических средств логического рассуждения SOA обеспечивает полную или частичную автоматизацию поиска сервисов, согласования, посредничества, запуска на выполнение и композицию. Семантические средства SOA не заменяют существующие технологии интеграции. Цель подхода состоит в построении нового уровня сервисов, основанного на принятых существующих промышленных стандартах и технологиях используемых в пределах существующих инфраструктур.

Мы представляем семантическую сервис-ориентированную архитектуру, которая соответствует основным принципам управления и, которая обеспечивает анализ, проектирование и поддержку архитектуры в части посредничества, сервисной

ориентации и логик обработки приложений. Мы впервые определяем семантическую архитектуру независимо от глобальной перспективы, в основе которой положены сервисные модели. Подробно описываем сервисы и типы обработки, которые обеспечивают архитектуру и технологии, которые используются для построения сервисных приложений.

1. Принципы управления на основе знаний

Семантическая сервис-ориентированная архитектура строится с использованием принципов, которые определяют управление на основе знаний, разработку и выполнение приложений. Эти принципы отражают фундаментальные аспекты сервис-ориентированного и распределенного окружения семантической интеграции бизнес-сервисов [3]. Принципы включают:

- принцип сервисной ориентации, обеспечивающий подходы для анализа, проектирования и выполнения приложений, которые основаны на возможностях сервисов – повторное использование, свободное связывание, абстрактное моделирование, композиционность, автономность, поиск и др.;
- семантический принцип, который обеспечивает формальное описание информации и динамические модели, допускающие автоматизацию решения задач с помощью логического рассуждения. Комбинируя этот принцип с принципом сервисной ориентации семантика позволяет определять такие характеристики сервисов: масштабируемость, семантическую интероперабельность, формальные модели сервисов и онтологий, позволяющие обеспечить частичную или полную автоматизацию решения задач, например, поиск сервисов, согласование, композицию сервисов и другие;
- принцип принятия решений, как одно из фундаментальных положений искусственного интеллекта. Он обеспечивает такую характеристику архитектуры, которая основана на целе-ориентированном поиске и выполнении сервисов. Пользователи описывают запросы как семантические цели независимые от самих сервисов, в то время как архитектура с помощью логического рассуждения над целями и описаниями сервисов обеспечивает их наилучшее достижение. Пользователям не нужно быть осведомленным в логике обработки, а заботиться только о результате и его качестве;
- принцип распределенности, который позволяет агрегировать возможности нескольких вычислительных объектов путем сотрудничества. Этот принцип обеспечивает выполнение множества сервисов в сети, обеспечивая масштабируемость и требуемое качество процесса.

2. Семантическая сервис-ориентированная архитектура

Семантическая сервис-ориентированная архитектура, построенная на вышерассмотренных принципах, показана на рис. 1. Архитектура содержит следующие основные уровни: 1) пользователей или группы пользователей; 2) принятия решений; 3) заказчиков сервисов; 4) посредников сервисов, обеспечивающие интеграцию и взаимодействие сервисов; 5) поставщиков сервисов, обеспечивающие публикацию функциональности серверных систем как бизнес-сервисов.

2.1. Уровень пользователей представляет группы различных пользователей, использующие функциональность архитектуры для своих целей. Идентифицируют две основные группы пользователей: пользователи приложений и инженеры. Пользователи приложений составляют группу, для которой архитектура обеспечивает доступ к функциональности через специализированные приложения и интерфейсы.

Например, пользователи могут выполнять обмен информацией для приобретения продукции или выполнять размещение заказов, или выполнение финансовых сделок. Задача этого уровня состоит в том, чтобы позволить пользователям взаимодействовать с бизнес-процессами он-лайн и, при этом, сократить физические взаимодействия с процессами сервера.

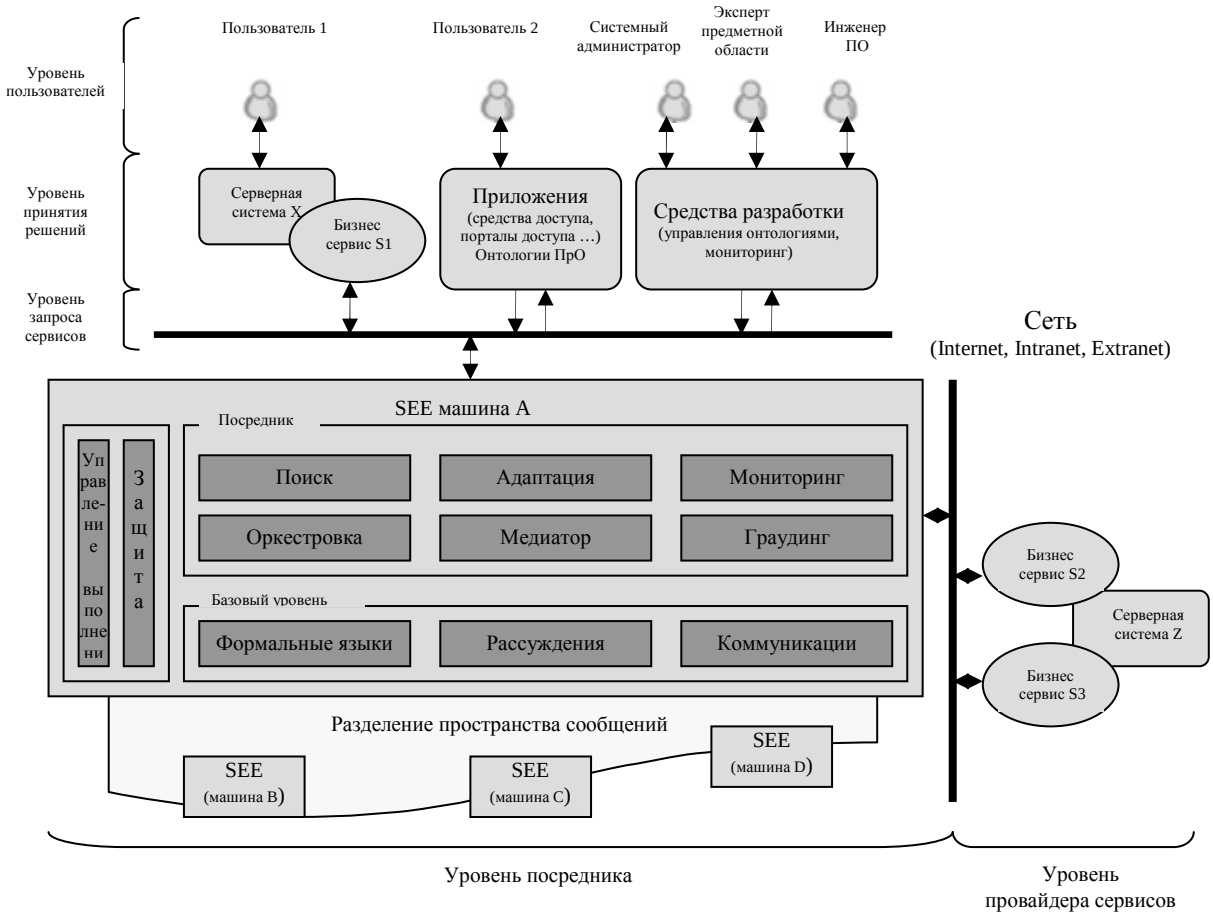

Рис. 1. Глобальная схема архитектуры

Другая группа пользователей – инженеры, которая формирует уровень, обеспечивающий развитие и администрирование задач в архитектуре. Эти задачи должны поддерживать жизненный цикл SOA, в том числе моделирование сервисов, создание, композицию, развертывание (публикация) и управление. В эту группу могут быть вовлечены такие пользователи: эксперты предметных областей (моделирование, создание онтологий), администраторы системы (развертывание, управление); конструкторы программного обеспечения и другие.

2.2. Уровень принятия решений представляет приложения и инструменты, которые поддерживают пользователей в части формирования запросов и преобразования их в форму пользовательских целей. Через уровень принятия решений, пользователь сможет формулировать проблему, обеспечивать взаимодействие с архитектурой в процессе обработки запросов и получать желаемые результаты. Серверные процессы этого уровня обеспечивают интерфейс для бизнес-процессов. Для этого строятся специализированные приложения в требуемой предметной области, используя онтологии предметной области и инструменты разработки, обеспечивающие необходимую функциональность для развития и администрирования задач в пределах архитектуры.

Функциональность средств разработки покрывают этапы жизненного цикла SOA, в том числе моделирование сервисов, создание, композицию, развертывание (публикация), управление и др.

Интегрированная среда разработки (IDE – Integration Design Environment) обеспечивает формирование семантических описаний (сервисы, цели и онтологии), создание схем посредничества между посредником и внешними системами. Объе-

диняя эту функциональность, разработчик может создавать и управлять онтологиями, Веб-сервисами, целями и посредниками, создавать онтологии для посредников, встраивать их в среду посредников.

2.3. Уровень запросов сервисов. Клиенты системы формируют цели посредством описания запросов и интерфейсов, через которые осуществляется взаимодействие с потенциальными сервисами. Средства для принятия решений используют семантическую спецификацию сервисов.

2.4. Уровень посредника. Посредник является ядром архитектуры, обеспечивающий главную функциональность в части интеграции и взаимодействия бизнес-сервисов. Посредники представляют семантическую среду выполнения сервисов (SEE). SEE определяет необходимую концептуальную функциональность, которая реализуется (полностью или частично) сервисами-посредниками. Функциональность посредника используется на следующих уровнях: вертикальный уровень, уровень посредника и базовый уровень. Среда выполнения основана на средствах WSMX и IRS-III [4, 5].

Вертикальный уровень определяет функциональность структуры посредника, которая используется базовыми уровнями, но остается невидимой для них. Эта технология использует так называемый ”Голливудский принцип”, который подразумевает ”не звоните нам, мы позвоним вам” [6]. Функциональность обеспечивает координацию и управление процессами в посреднике.

Управление выполнением основано на управлении различными сценариями, называемыми семантиками выполнения, и реализует распределенное выполнение сервисов.

Безопасность включает средства идентификации, аутентификации конфиденциальности, кодирования данных, поддержку отслеживания и безотказности в пределах сценариев выполнения.

Уровень посредника (брокера) обеспечивает:

- поиск бизнес-сервисов, которые соответствуют цели заказчика;
- оркестровку выполнения бизнес-процессов, включая переговоры между заказчиком и поставщиком сервиса в пределах бизнес-процесса;
- мониторинг осуществляет контроль выполнения сервисов и используется для сбора информации о сервисах, например, Q_OS – показатель качества, идентификация сбоев в процессе выполнения и т.д.;
- управление ошибками обеспечивает обработку ошибок при выполнении Веб-сервисов;
- адаптацию, представляющую собой согласование пользовательских предпочтений в пределах сценария выполнения, например, выбор сервиса обеспечения переговоров, заключение контракта;
- посредничество, определяющее совместимость данных и процессов на функциональном уровне;
- композицию сервисов в процессе выполнения потока работ (бизнес-процессов);
- Граундинг, устанавливающий связь между семантическим уровнем (WSMO) и синтаксическим уровнем (WSDL), используемый при запуске сервисов на выполнение.

Базовый уровень включает функциональность, необходимую для успешного выполнения процессов на уровне посредника. Базовый уровень содержит:

- формальные языки, определяющие синтаксические действия (например, анализ–Parsing), семантические языки, использующие семантическое описание сервисов, целей и онтологий;
- рассуждения, определяющие функциональность над семантическими описаниями;
- репозитории для хранения элементов системы – сервисов, онтологий;
- средства, обеспечивающие внутренние и внешние коммуникации посредника.

Посредник среды выполнения сервисов обеспечивает распределенность, в данном случае ряд систем посредника объединяются путем разделяемых сообщений и, таким образом, обеспечивают масштабируемость процессов интеграции.

2.5. Уровень поставщиков сервисов. Поставщики сервисов обеспечивают различные серверные (back-end) системы. Серверная система служит для обеспечения функциональности заданной цели бизнес-сервиса. В зависимости от развертывания архитектуры и сценариев интеграции, сервера системы могут обеспечивать одну организацию (один поставщик сервисов) или множество организаций (много поставщиков сервисов), которые связаны по сети (Интернет, Интранет или Экстранет). Архитектура может обслуживать различные модели интеграции: бизнес-бизнесу (B2B), приложение предприятию (EAI) или приложение приложению (A2A). Во всех моделях функциональность серверов представляется как семантическое описание бизнес-сервисов.

3. Модель семантических сервисов WSMO

Модель семантических сервисов формирует дополнительный уровень над существующими моделями Веб-сервисов, путем добавления семантической разметки для функциональных, не функциональных и динамических характеристик сервисов. Существует несколько инициатив в этой области, например, онтология моделирования Веб-сервисов (WSMO) [7], Owl-s [8] и WSDL-s [9].

В соответствии с требованиями архитектуры и принципами управления в предложенном подходе мы выбрали модель WSMO. Выбор WSMO, а не других спецификаций (например, Owl-s) основывался на том, что WSMO явным образом сфокусирована на принятии решений и интеграции целей и сервисов, поэтому полностью соответствует принципам семантического принятия решений и семантической организации сервисов. Кроме того, WSMO развивается как структура, которая включает: описание концептуальной модели, определяющей характеристики Веб-сервисов: онтологии, цели, Веб-сервисы и посредники. Язык моделирования Веб-сервисов (WSML) [7, 10] представляет семейство онтологических языков, основанных на логических формализмах и уровнях логической выразительности, в том числе дескриптивной логики и логики программирования. Среда выполнения Веб-сервисов (WSMX) – представляет собой средства для реализации системы посредника [11]. Инструментарий для моделирования Веб-сервисов (WSMT) используется для разработки описаний таких объектов WSMO: сервисы, цели и онтологии. Обеспечивается также создание проекций посредника на внешние системы. WSMO и его компоненты обеспечивают основы семантического моделирования сервисов и семантической технологии, которая может адаптироваться к требованиям предметных областей (например, путем расширения функциональности WSML,WSMX, WSMT и т.п.).

Концептуальная модель WSMO представляет собой модель верхнего уровня, определяющая онтологию, используемую для моделирования бизнес-сервисов. Она же содержит описание онтологии, Веб-сервисов, целей и посредников.

Онтологии обеспечивают формальное определение семантики для WSMO. Двумя ключевыми отличиями онтологий являются принципы разделяемой концептуализации и формальной семантики. Общая концептуализация представляет собой средство, обеспечивающие информационную совместимость через независимые цели и описания Веб-сервисов.

Веб-сервисы определяют функциональные возможности и один или большее число интерфейсов, которые дают клиентам сервиса возможность обращения к нему. Возможности сервиса моделируются посредством использования пред- условия и предположения о состоянии информационного пространства и внешнего мира перед выполнением, выходные пост- условия и результаты, определяющие состояние после выполнения сервиса. Интерфейсы сервиса разделяются на хореографию и оркестровку. Хореография определяет способ взаимодействия с сервисом, при этом оркестровка определяет декомпозицию его функциональности в терминах других сервисов.

Цели представляют описание запросов, которые хочет достичь пользователь. Цели WSMO описываются в терминах

предметной области выполняемого запроса заданного сервиса. Цель WSMO характеризуется возможностью запроса и интерфейсом запроса.

Посредники представляют элементы, которые нацелены на разрешение структурных, семантических или концептуальных несоответствий, которые появляются между различными компонентами в пределах среды WSMO.

Сравнение WSMO с его основным конкурентом Owl-s и другими подходами представлено в [12].

4. Сценарий примера семантической сервис-ориентированной архитектуры

В этом разделе рассмотрим пример, на котором исследуются различные аспекты семантической сервис-ориентированной архитектуры. Сценарий примера и его реализация основана на предложениях инициативы SWS Challenge [13], обеспечивающая стандартное множество сложных задач, основанных на индустриальных спецификациях и требованиях. Как показано на рис. 2, сценарий включает различных поставщиков сервисов (например, корпорации «Racer» и «Mueller»), которые предлагают поставки различной продукции через обращение к рынку «Moon». С другой стороны, есть запрашивающая сторона сервисов под названием «Blue», которая намеревается купить определенную продукцию по возможно приемлемой цене [14].

Рынок «Moon» взаимодействует с электронным рынком на уровне системы-посредника семантической сервис-ориентированной архитектуры. Следуя сценарию «Moon» использует системы для взаимоотношений с клиентами (CRM) и расчетную систему (OMS), при этом «Blue» использует стандартную систему RosettaNet8 [15].

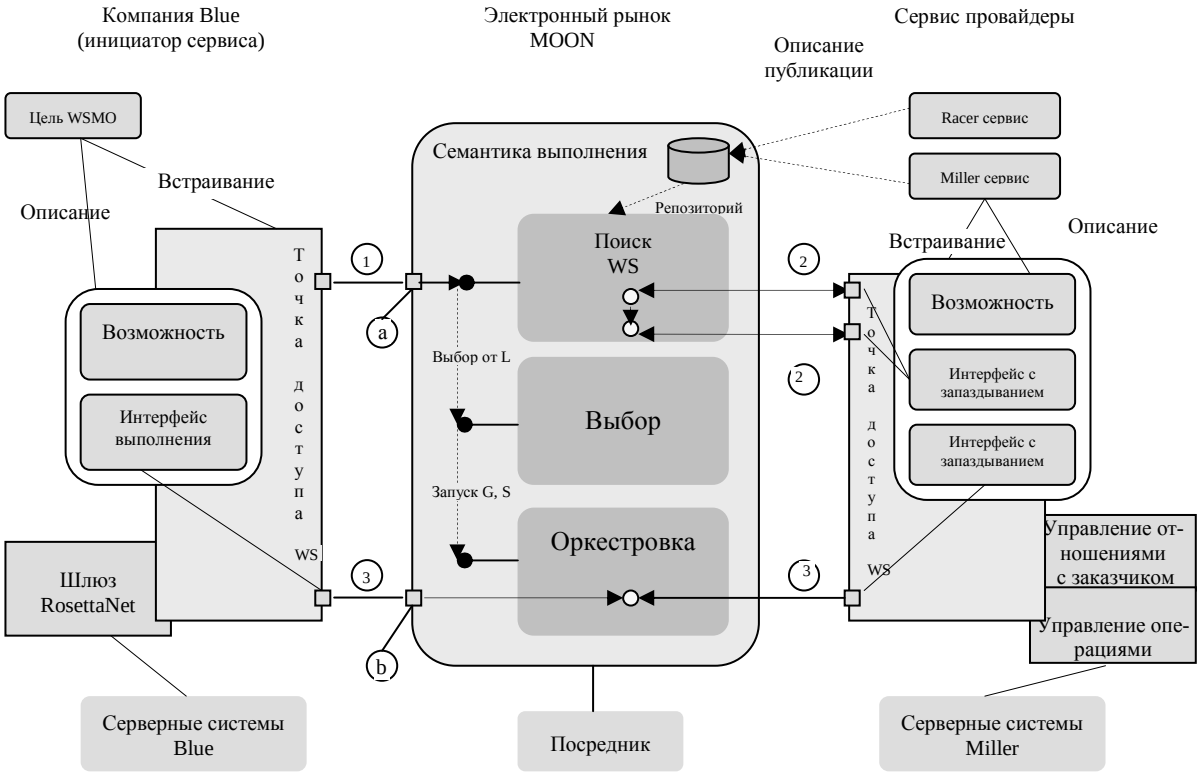

Рис. 2. Пример выполнения сервисной системы

Инженеры, представляющие пользователей и поставщиков сервисов, используют модели WSMO для моделирования сервисов и запросов к сервисам, так же используются различные онтологии и различные описания хореографии. В частности, «Blue» посылает запрос и ожидает получения ответа согласно спецификации RosettaNet PIP3A4. С другой стороны, «Mueller» для того, чтобы обработать запрос должен выполнять большое число взаимодействий с конечными системами, например, для идентификации клиента в CRM, необходимо открывать заказ в OMS, добавить элементы и закрывать заказ. Поэтому существуют проблемы функциональной совместимости между «Blue» и «Mueller» – «Blue» использует информационную модель и хореографию, определенную PIP3A4, а «Mueller», использует информационную модель и хореографию, определенную системами CRM/OMS.

Пользователи и поставщики сервисов поддерживают интеграцию через реализацию адаптеров серверных систем. Как компания «Blue», так и компания «Mueller» ответственны за поддержку этих адаптеров и их интеграцию с посредниками через семантические описания и/или через интерфейсы с посредниками.

Инженеры, представляющие поставщиков и пользователей сервисов соответственно издают онтологии, использующие для спецификации цели WSMO и сервисных описаний. Кроме того, формируются правила проекции между онтологиями поставщиков и пользователей, онтологиями, используемыми в системе посредника.

Сценарий работы представлен следующим образом: все бизнес-партнеры, их модели и бизнес-сервисы используют WSMO. «Blue» посылает запрос к посреднику системы на поставку оборудования. Запрос определяется в цели WSMO. Посредник получает цель. Семантика выполнения включает: поиск, выбор и оркестровку. Согласование сервисов для целевого и потенциальных функциональных сервисов выполняется на абстрактном уровне как в процессе поиска, так и на уровне образца. Абстрактный уровень поиска позволяет минимизировать число возможных согласований Веб-сервисов, которые обеспечивают заданную цель. Поиск на уровне образца осуществляет детальное согласование, рассматривая как данные образца сервиса, так и цели. Выбор наилучшего сервиса («Mueller») основан на предпочтениях. Процесс оркестровки осуществляет выполнение переговоров между сервисами «Blue» и «Mueller», которые выполняют обработку описаний хореографии в соответствии для «Blue» и сервисов «Mueller's».

5. Представление сервиса

Семантическая среда выполнения (SEE) обеспечивает декомпозицию сервиса, которая позволяет разделить и упростить сервисы, каждый из которых может иметь свою собственную структуру. Следуя принципам SOA, архитектура SEE выделяет сервисы-посредники таким образом, отделяя описания сервисов и их интерфейсы от выполнения. Такое разделение обеспечивает гибкость и масштабируемость при модернизации или замене функциональности сервисов посредников, которые используют заданные интерфейсы.

Сервисы обеспечивают требуемую функциональность для определенной цели. Сервисная ориентация разрешает горизонтальное представление сервисов, кроме того отличают сервисы нескольких типов. По типу функциональности различаем два вида сервисов:

- бизнес сервисы, представляющие собой сервисы, которые обеспечиваются различными серверными системами поставщиков сервисов. Бизнес-сервисы являются предметом интеграции и взаимодействия в пределах архитектуры и могут обеспечивать требуемое значение для пользователей. В архитектуре, бизнес-сервисы публикуют поставщики сервисов путем формирования семантических описаний, согласованных с моделью WSMO. Бизнес-сервисы издаются и содержатся в хранилищах посредников;
- сервисы-посредники представляют собой вспомогательные средства для интеграции и взаимодействия бизнес-сервисов. Сервисы-посредники встраи-

ваются в систему-посредник.

На уровне абстракции бизнес-сервисов различают следующие два вида сервисов:

- Веб-сервисы, представляющие собой общие сервисы, имеющие несколько форм и, которые далее могут конкретизироваться (например, приобретают билет на рейс);
- сервисы, представляющие собой фактические образцы Веб-сервисов и, которые обеспечивают конкретное значение для пользователей (например, приобретают билет на рейс компании «Аэросвит» из Киева до Нью-Йорка).

5.1. Сервисы-посредники реализуют общую или частичную концептуальную функциональность посредника. Эти сервисы отражают среду реализации SEE посредника WSMX. Сервисы-посредники могут комбинироваться с процессами посредника, которые обеспечивают дополнительную функциональность системы. Основными сервисами-посредниками в WSMO являются:

Ядро можно рассматривать как сервис, который реализует управление выполнением, мониторинг, отработку ошибок и формальные языки концептуального моделирования посредника. Ядро, реализующее коммуникацию и координацию процессов посредника определяется семантикой выполнения, которая определяет взаимодействие различных сервисов-посредников, обслуживающие процесс посредника для заданной цели. Каждый процесс посредника запускается через внешний интерфейс сервисом коммуникации и связан через другие внешние интерфейсы асинхронной связью в процессе взаимодействия с посредником. В посреднике может существовать несколько семантик выполнения, которые обеспечивают процессы моделирования, создания, развертывания и управления, сокращая время разработки. Ядро посредника осуществляет также поддержку формального языка – анализатора, реализующий семантические сообщения в объектной модели WSMO4J [16]. Кроме того, ядро осуществляет распределенные действия посредника, позволяющие взаимодействие с множеством физических машин, используя общее пространство сообщений. Разделение области сообщений обеспечивает абстракцию запроса для распределенной архитектуры, которая обеспечивает масштабируемость процессов интеграции.

Коммуникационный сервис реализует коммуникацию и Граундинг на уровне концептуальной функциональности посредника. Любое сообщение направлено или послано от посредника системы передается через этот компонент. Интеграция семантических Веб-сервисов в системе осуществляется посредством сообщений, сопровождающих семантические описания данных в соответствии с моделью WSMO. Механизм, используемый для запуска сервисов, основан на SOAP и WSDL спецификациях. Поэтому, компонент коммуникации также реализует механизмы для семантик Граундинга уровня WSMO и физического уровня запуска сервисов.

Сервис рассуждений реализует функциональность рассуждений в посреднике. Он обеспечивает поддержку рассуждений над семантическими описаниями ресурсов. Рассуждение представляет собой важную функциональность, которая требуется в процессе выполнения, например, поиск, посредничество для данных, посредничество для процессов и т.п. К рассуждениям применяются различные требования, которые основаны на варианте языка WSML, используемого для семантических описаний. В основе рассуждений положена дескриптивная логика, которая нужна для того, чтобы использовать варианты языка WSML, язык Datalog или F-logic, положенные в основу рассуждений, когда используется WSML-flight или WSML-rule соответственно. Использование различных средств для рассуждений зависит от связи приложений и полноты используемых семантических описаний в моделировании. Компонент рассуждений в дополнении к существующим рассуждениям (WSML2Reasoner [17]) предлагает универсальный уровень, который соответствует упомянутым требованиям. В зависимости от варианта WSML, запросы передаются машине рассуждений в прозрачной форме. Рассуждения для WSML

работоспособны также в WSML WG [18].

Сервис хранения обеспечивает хранение и управление различных объектов, в том числе целей, сервисов, онтологий и посредников (схемы правил). Все объекты описываются с использованием семантического языка WSML.

Посредники данных. Посредник данных реализует концептуальную функциональность уровня посредника, которая помогает реализовать выполнение процессса, в случае если используются различные онтологии при описании сервисов. Также он используется в процессе поиска сервисов в соответствии с запросами целей и сервисами, которые удовлетворяют целям или требуются в процессе переговоров между запросами и провайдерами сервисов. Описания сервисных интерфейсов могут использовать различные онтологии.

Посредничество данных действует в соответствии со схемой правил между онтологиями, которые предварительно должны быть опубликованы в архитектуре перед тем, как реализуется посредничество. Эти правила должны быть созданы на этапе проектирования как часть средств управления онтологиями WSMT. Детальное описание посредника данных для семантических Веб-сервисов рассматривается в [4].

Процессы посредника (медиатора) реализуют функциональность уровня посредника. При реализации посредника используются различные интерфейсы хореографии, определенные в описаниях сервисов, которые используются в переговорах. Процессы-посредники применяются вместе с хореографией, медиатором данных и компонентами коммуникации, на этапе связи заказчика и поставщика сервиса. Путем анализа процесса посредник решает возможные конфликты хореографии, в том числе остановки сообщения, в случае когда сообщение не нужно для какой-либо хореографии, и в случае когда сообщения должны обмениваться в определенном порядке обеими сторонами и т.п. Детальная информация о концептуальном определении процесса посредника и конфликтов хореографии рассматривается в [5].

Усовершенствование цели представляет собой процесс создания абстрактной цели на основе конкретной цели. Абстрактная цель не содержит данных образца. Данные образца обеспечиваются отдельно из определения цели синхронно или асинхронно, тогда как конкретная цель непосредственно содержит встроенные данные образца, как часть определения WSMO. Например, WSMO конкретизирует цель, которая может содержать аксиоматику в форме ?x[namehasValue “HarryPotter"] memberOf book, тогда как абстрактная цель содержит аксиоматику в форме ?x memberOf book, где образец book (книга) указывается отдельно от определения цели.

Поиск Веб-сервисов представляет собой процесс поиска сервисов, удовлетворяющих требованиям заказчиков. Существуют несколько теоретико-множественных отношений между описаниями сервисов, например, точное соответствие, встроенное соответствие, предполагаемое соответствие, разделяемое соответствие и дизъюнктивное соответствие. Детальная информация о поиске Веб-сервисов в WSMO рассматривается в [6].

Поиск сервиса представляет собой процесс поиска сервиса, удовлетворяющего цели пользователя. Сервис согласованный на абстрактном уровне, согласовывается на уровне образца, в случае когда была бы получена дополнительная информация от поставщика сервисов, например, цена или пригодность продукта. Такая информация обычно имеет динамические характеристики и не подходит для статических характеристик описания онтологии. Для этой цели выполняется взаимодействие с запаздыванием. Детальная информация о поиске сервисов в WSMO рассмотрена в [7].

Выбор сервиса представляет собой процесс, в котором осуществляется выбор одного сервиса из множества сервисов-кандидатов, полученных в результате выполнения операции поиска, который лучше всего удовлетворяет пользовательским предпочтениям. Критерием выбора являются различные не функциональные свойства, например, уровневые соглашения сервиса (SLA), качество сервиса (QOS) и

т.п. которые могут выражать пользовательские предпочтения в виде не функциональных свойств описания цели. Детальная информация о выборе сервисов в WSMO рассматривается в [8].

Оркестровка. Процесс оркестровки осуществляет динамические переговоры между заказчиком и поставщиком сервисов, обрабатывая интерфейсы хореографии. Оркестровка использует процессы взаимодействия с посредником и коммуникацию сервисов, которые вызываются каждый раз при обмене сообщениями между заказчиком и поставщиком сервиса.

5.2. Бизнес-сервисы содержат спецификацию функциональности серверных (back-end) систем, которые описаны средствами WSMO. Описание бизнес-сервисов издаются и сохраняются в хранилищах посредника и управляются в посреднике как во время разработки (при создании сервиса), так и во время выполнения (связывание с запаздыванием и выполнение сервисов). Важным аспектом фазы создания сервисов является семантическое моделирование бизнес-сервисов, которые определяются на следующих уровнях.

Концептуальный уровень содержит спецификацию источников информации, которая используется для моделирования бизнес-сервисов. Эта информация представляет проблемно-зависимую часть, например, схемы базы данных, стандарты сообщений, B2B стандарты или различные классификаторы для классификации промышленных объектов [15]. Информаия наследуется из бизнес-процессов организации, существующих стандартов, используемых систем или существующие специификации организационных систем (например, системы планирования ресурсов предприятия).

Логический уровень представляет собой семантическую модель бизнес-процессов на различных стадиях выполнения. Для этой цели используем сервисную модель WSMO вместе с языком семантической разметки WSML. WSMO определяет семантику сервисов, в том числе нефункциональные свойства, функциональные свойства, интерфейсы (определяющие поведение) и онтологии, которые определяют информационные модели сервисов. Кроме того обеспечивается Граундинг от семантических описаний WSMO к WSDL и XML. Так же должны быть определены схемы для запуска сервисов.

Физический уровень представляет физическое окружение, используемое для запуска сервисов. Для этого используются WSDL и SOAP спецификации. Должен быть определен Граундинг между семантическими описаниями сервисов и WSDL. Такое определение Граундинга может быть представлено в описаниях WSMO на уровне интерфейса сервисов или описаний WSDL, используя подход семантических аннотаций для WSDL (SAWSDL) [13]. Определение Граундинга зависит от подхода моделирования.

Семантическое моделирование бизнес-сервисов использует сервисную модель WSMO и дополнительную проблемно-зависимую информацию. Важным аспектом фазы моделирования является переход от семантического описания сервиса (логический уровень) WSMO к WSDL описаниям (физический уровень). В WSMO Граундинг определяется для:

- интерфейса сервиса WSMO и сообщений WSDL. Этот вид Граундинга конкретизирует ссылки для каждого используемого понятия в интерфейсе сервиса к входным или выходным сообщениям, использованных в WSDL.
- WSMO онтологии и схемы XML. Этот вид Граундинга конкретизирует схему подъема и понижения, определяющую проекцию схем XML и онтологий для того, чтобы выполнять преобразования образца в процессе запуска сервисов.

Среди уровней моделирования семантических бизнес-сервисов отличают два подхода: нисходящий и восходящий.

Нисходящий подход. В этом подходе явно не существует представления сервиса в WSDL и, поэтому его нужно создать на этапе создания/моделирования сервиса. В данном случае сервисы проектируются таким образом, чтобы они могли обработать семантические описания онтологий и сервисов. Для Граундинга первого типа используются ссылки на понятия сервиса, которые определяются в интерфейсе

WSDL, его входных и выходных сообщений. Определение Граундинга второго типа связано с реализацией сервиса непосредственно. Семантические сообщения передаются сервису, в котором выполняется понижение. Обратно, подъем выполняется в сервисе для онтологии и передается посреднику, в котором выполняется обработка согласно семантики выполнения. Информация о Граундинге WSMO рассмотрена в [19].

Восходящий подход. В этом подходе предполагается, что существует основное представление сервиса в WSDL. Определение Граундинга определяется таким же образом, как в нисходящем подходе. Однако, существуют отличия для определения Граундинга второго типа. К описаниям WSDL прилагается схема, использующая спецификации SAWSDL. В результате понижения создается схема XML, которая передается согласно определенному интерфейсу Граундинга. В результате подъема создаются образцы онтологии, которые используются для последующего выполнения посредником. Информация о WSMO, Граудинге и использовании SAWSDL рассматривается в [20].

6. Технология управления выполнением сервисов

Управление выполнением сервисами осуществляет ядро посредника, использующее JMX-расширений Java [21]. Выполнение сервисов соответствует трем главным функциональным требованиям: управление, коммуникация и координация, обработка семантики.

6.1. Управление. Мы делаем строгое разделение между операционной логикой и логикой управления, рассматривая их как ортогональные понятия. Ядро системы управления представляет агент управления, который предлагает несколько сервисов. Основным является сервис начальной загрузки, который отвечает за конфигурирование функциональных компонент. Агент управления встраивается непосредственно в приложение. Сервис реализации управления использует методы управления через планируемые действия, и позволяет администрировать независимое управление и контролируемый интерфейс. Через этот интерфейс для каждой цели сервиса связываются ряд консолей управления. В частности, поддерживается управление через терминальный интерфейс, Веб браузер и Эклипс.

Управление выполнением сервисов также используют виртуальный инструментарий, чтобы управлять производительностью системы и метрикой повышения производительности. Может использоваться общая метрика для всех компонентов, можно использовать метрику к некоторым компонентам.

Принципы управления выполнением сервисов распределенной архитектуры служит средством для распределения компонентов. Однако, предпочитаемый путь к распределению состоит в организации системы как федерации агентов. Каждый агент имеет свой собственный сервис управления выполнением и подмножество функциональных компонентов. Для того, чтобы скрыть сложность федерации по управлению приложением, обеспечивается единый тип агента, т.е. единая точка доступа к управлению и единые интерфейсы администрирования.

6.2. Коммуникация и координация. Коммуникация основана на событиях в пределах посредника. В соответствии с [9], обмен сообщениями выполняется через кортежи, которые обеспечивает общее пространство, разрешающее взаимодействие между компонентами без прямого обмена. Это взаимодействие выполняется путем использования механизма «издание-абонирование». Пространство кортежей разрешает коммуникацию между распределенными компонентами, выполняющимися как на локальных, так и на удаленных машинах.

Технология пространства кортежей, используемая в посреднике, основана на подходе, рассмотренном в [10], в котором компоненты опубликованы и выполнена подписка на кортежи. Пространство управления передачи данных обеспечивает синхронизацию и устойчивость процесса. Пространство кортежа может быть композитным, состоящим из множества распределенных и синхронизированных

хранилищ кортежей.

Инфраструктура коммуникации взаимодействует с транспортным уровнем. Через транспортный уровень, компонент подписывается на определенный тип события. Подобный механизм применяется, когда события опубликованы.

6.3. Семантика выполнения разрешает выполнение функциональных компонентов, определяет логику посредника, которая реализует поведение промежуточного уровня. Управление выполнением сервиса обеспечивается структурой, которая позволяет выполнять семантику на множестве компонентов с множеством выполнений. В частности, управление выполнением сервисов обеспечивает жизненный цикл семантики выполнения, управления и мониторинга. Выполнение основанное на определении семантики будет использовать и производить определенные виды событий. Одна оболочка инициирует событие с некоторым содержимым сообщения, а другая – потребляет это событие и реагирует на него.

Заключение

Семантическая сервис-ориентированная архитектура, рассмотренная в этой работе представляет новый подход к интеграции и взаимодействия сервисов с помощью семантических языков и семантических моделей сервисов. Надстраивая онтологию Веб-сервиса и принимая во внимание принципы управления сервисами, семантическое моделирование и проблемы принятия решений, архитектура обеспечивает средства частичной автоматизации задач, например, поиск, посредничество, выбор и выполнение семантических Веб-сервисов. Важным аспектом, такой основы является интеграция, которая основана на цели, поиске и запуске семантических сервисов, когда пользователи описывают запросы как цели независимо от сервисов, архитектура обеспечивает достижение цели с помощью логического рассуждения над семантическими описаниями. Пользователю не нужно быть осведомленным в обработке логики, а заботиться только о результате и его желательном качестве.

Основные принципы подхода состоят в том, что архитектура определяется от нескольких перспектив, представляя глобальный, сервисный, обрабатывающий вид и вид технологии. Глобальный вид идентифицирует несколько уровней архитектуры, в том числе совместное использование ресурсов, проблемы принятия решений, заказчики сервисов, посредники и поставщики сервисов. Ядро архитектуры представлено на уровне посредника, для которого определена концептуальная функциональность. Сервисная перспектива архитектуры определяет типы сервисов как промежуточные, так и бизнес-сервисы, каждый тип сервисов обеспечивает спецификацию характеристик, сервисы посредники осуществляют функциональность посредника, а бизнес-сервисы могут моделироваться на основе семантической модели WSMO.

Технология основана на посредничестве, которое следует за распределенным принципом и обеспечивает поддержку распределенного управления, коммуникации и координации процессов посредника.

Один из главных аспектов архитектуры состоит в обеспечении гибкой интеграции сервисов, которая адаптивна к изменениям в бизнес требованиях. Обеспечивается способность к адаптации и изменениям в серверных системах путем изменений семантических описаний сервисов. Конечная цель архитектуры в нашей работе состоит в определении уровня, в котором архитектуру можно адаптировать без изменений в коде и конфигурации. Адаптация будет исключительно основана на рассуждениях над семантическими описаниями сервисов, моделировании и связывании. Предложенное расширение решения направлено на покрытие большинства стандартов B2B и объединении инфраструктуры, например, управление политиками, гарантия качества сервисов и т.п. В качестве цели возможно использовать дополнительные сценарии интеграции B2B, сосредотачиваясь на динамической композиции сервисов, которые расширяют функциональность сервисов посредника по обработке ошибок, безопасности, оркестровки и т.п.

1. *Андон П.І., Дерецький В.О.* Проблеми композиції сервісів в семантичному Web середовищі // Матеріали Міжнар. конф. «50 років Інституту кібернетики імені В.М. Глушкова НАН України». − К.; 24–26 грудня 2007. − К.; 2008. − С. 40−53.
2. *Андон П., Дерецкий В.* Проблеми построения сервис-ориентированых прикладных информационных систем в Semantic Web среде на основе агентного подхода // Проблеми програмування. − 2006. − № 2-3. − С. 493–502.
3. http://www.sws-challenge.org
4. http://www.wsmx.org
5. http://kmi.open.ac.uk/projects/irs
6. http://www.gotdotnet.ru/blogs/bezzus/1211/
7. *Roman D., Keller U., Lausen et al*. Web Service Modeling Ontology // Applied Ontologies. – 2005. – P. 77–106.
8. *Martin D. et al*. Owl-s: Semantic markup for web services, version 1.1 available at http://www.daml.org/services/owl-s/1.1/. Member submission, W3C 2004. available from: http://www.w3.org/Submission/OWL-/
9. *Patil A., Oundhakar S., Sheth A., Verma K.* Semantic Web Services: Meteor-SWeb Service Annotation Framework // In: 13th International Conf. on World Wide Web. –2004. – P. 553–562.
10. http://www.wsmo.org/wsml
11. *Cimpian E., Mocan A.* Wsmx process mediation based on choreographies // In: Business Process Management Workshops. – 2005. – P. 130–143.
12. http://www.wsmo.org/TR/d24/d24.2/
13. http://www.sws-challenge.org
14. *Дерецкий В., Богданова М., Горошанский С.* Подход к разработке программных приложений с использованием семантических Веб-сервисов // Проблеми програмування. − 2009. − № 4. − С. 59–70.
15. http://www.rosettanet.org
16. http://wsmo4j.sourceforge.net
17. http://tools.deri.org/wsml2reasoner
18. *Web Service* Modeling Toolkit. http://sourceforge.net/projects/wsmt
19. *Kopeck´y J., Roman D., Moran M., Fensel D.* Semantic web services grounding // In: AICT/ICIW. – 2006. – P.127.
20. http://www.w3.org/ws/sawsdl/
21. *Haselwanter, Thomas* WSMX Core - A JMX Microkernel // PhD thesis, University of Innsbruck. – 2005.

Получено 04.12.2009

Об авторе:

Дерецкий Валентин Александрович,
кандидат физико-математических наук,
ведущий научный сотрудник.

Место работы автора:

Институт программных систем
НАН Украины.
03187, Киев-187,
Проспект Академика Глушкова, 40.
Тел.: 38 044 526 4342.
e-mail: dva@isofts.kiev.ua.