Как моделирование способствует повышению эффективности SOA

Эффективность SOA заключается в способности обеспечить динамичность бизнеса посредством интеграции и многократного использования бизнес-процессов. В SOA это достигается двумя способами: путем поддержки решений, созданных на основе многократно используемых сервисов, инкапсулирующих функциональные возможности, изолированные от конкретной реализации, и предоставления средств для управления связностью между отдельными функциональными возможностями. Связующим звеном между бизнес-требованиями и развертываемым решением, основанным на использовании сервисов, может стать моделирование. Модели SOA переводят разработку на более высокий уровень абстракции, что позволяет сосредоточиться на бизнес-сервисах. Принципы управляемой моделями разработки могут использоваться для создания реализаций SOA при помощи таких платформ, как JavaTM 2 Platform, Enterprise Edition (J2EE) или IBM® CICS®, которые позволяют выполнить функциональные и нефункциональные требования и в то же время способствуют динамичности бизнеса.

Термин «сервис-ориентированная архитектура» (service-oriented architecture, SOA) используется в нескольких значениях. Специалисты-практики обычно используют термин SOA как для определения стиля архитектуры, так и для описания общей ИТ-инфраструктуры, поддерживающей функционирование созданных при помощи этого архитектурного стиля ИТ-систем. Такое толкование с точки зрения технологий является полезным, но неполным.

Чтобы потенциал ИТ-инфраструктуры на базе SOA (далее просто SOA) проявился в полной мере, ей необходимо быть значимой для бизнеса, то есть она должна управляться бизнесом и быть предназначенной для поддержки бизнеса. Нам необходим способ разработки SOA-решений, ассоциируемых с бизнес-требованиями, которые они удовлетворяют. Но если бизнес-требования даны в виде обычного списка элементов, а уровень абстракции SOA документирован в нескольких XML-документах, описывающих коллекцию Web-сервисов, этого довольно трудно добиться.

На самом деле, нам необходим метод для формального описания бизнес-требований; это позволит перейти на более высокий уровень абстракции, благодаря чему реализация SOA будет в большей степени похожа на бизнес-сервисы, а нам станет понятно, как эти сервисы смогут решать бизнес-задачи и обеспечивать достижение целей бизнеса. Это свяжет размещаемое решение с его предполагаемой ценностью для бизнеса. В то же время нам нужен метод для изоляции бизнес-задач от постоянно изменяющихся SOA-платформ, которые их поддерживают.

Решить эти задачи можно с помощью моделирования и использования принципов разработки, управляемой моделями (model-driven development, MDD). Модели позволяют абстрагироваться от деталей реализации и сосредоточиться на тех проблемах, от которых зависят архитектурные решения. До некоторой степени описываемый нами подход применяет к разработке SOA-решений один из фундаментальных принципов SOA: изоляция проблем и слабая связанность. В этом материале мы четко разграничиваем задачи и обязанности бизнес-аналитиков и сотрудников отделов информационных технологий.

Обычно бизнес-аналитики концентрируются на организационных и операционных требованиях бизнеса, необходимых для достижения бизнес-целей и решения задач, создающих определенную бизнес-концепцию. Часто они не имеют представления (вследствие недостаточной подготовки этих специалистов) об ИТ-задачах, например, многократном использовании, сцеплении и связности, распределенности, обеспечении безопасности, персистентности, целостности данных, параллелизме, восстановлении после сбоев и т. д. Кроме того, инструменты моделирования

бизнес-процессов не часто включают функции, необходимые для решения этих задач, а если такие функции и присутствуют, то они вряд ли могут эффективно использоваться бизнес-аналитиками.

В статье демонстрируется использование UML-моделей и профиля программирования сервисов IBM® Software Service Profile для разработки SOA-решений, учитывающих бизнес-требования, но независимых от реализации. Как правило, человек лучше усваивает идеи, изучая конкретные, типичные и готовые примеры. Здесь мы как раз и используем этот подход. Мы не будем тратить много времени на подробное описание UML, а сосредоточимся на том, как использовать UML в проектировании и разработке.

Хотя серия статей в целом посвящена созданию Web-сервисов из моделей SOA, для начала мы расскажем о бизнес-задачах, которые попытаемся решить, чтобы сделать основой решения некоторую ценность для бизнеса. Затем мы изучим бизнес-процессы, объясняющие, что необходимо сделать в данном бизнесе, чтобы решить эти задачи. На основе полученной информации создаются функциональные бизнес-требования, которые должно выполнять SOA-решение. Затем мы используем эти бизнес-процессы как вспомогательную информацию для идентификации сервисов, которые будут полезны для создания SOA-решений, удовлетворяющих этим требованиям.

Изучая статьи этой серии, мы будем создавать спецификации и реализации сервисов, которые будут обеспечивать выполнение описанных требований при помощи архитектуры, обеспечивающей многократное использование этих сервисов в будущем и динамичность бизнеса. И наконец, мы воспользуемся принципами MDD для создания Web-сервисов, которые можно будет развернуть в производственной среде и выполнить.

Чтобы добиться еще большей реалистичности решения, мы будем использовать имеющиеся инструменты IBM® Rational® и IBM® WebSphere®для создания артефактов решения и связи их между собой, благодаря чему мы сможем проверить соответствие нашего решения требованиям и более эффективно управлять изменениями. В таблице 1 перечислены все процессы, которые мы будем использовать при разработке примера, и инструменты для создания артефактов.

В этой серии статей мы сосредоточимся на том, как собрать бизнес-требования, создать модели сервисов, которые будут обеспечивать их выполнение, создать и разместить решения, воплощающие в жизнь эти проекты. Мы также расскажем об инструментах, которые помогут нам сделать это. В статьях будет описан минимальный набор функций моделирования SOA, предлагаемых в настоящее время инструментами IBM, которые перечислены в таблице 1; эти функции можно использовать для моделирования требований потребителя и поставщика сервиса на любом архитектурном уровне. Мы не будем рассказывать обо всех методах или технологиях сбора требований, принципах анализа и оценки реализаций сервисов или секционирования сервисов на различных архитектурных уровнях. Более подробную информацию по этим важным темам можно найти в IBM® Rational Unified Process® (RUP®) for SOA и RUP for SOMA. Эти подключаемые модули для IBM® Rational® Method Composer содержат процессы разработки, инструкции, обучающие материалы по использованию инструментов и статьи, в которых демонстрируются дополнительные методы использования инструментов для разработки моделей сервиса и решений

Пример. Обработка заказов на приобретение

Наш пример построен на примере Purchase Order Process (Обработка заказов на приобретение) из OMG UML Profile и Metamodel for Services (UPMS) RFP и примере из спецификации BPEL 1.1. Спецификация BPEL 1.1 содержит только часть решения, потому что в ней не определены коррелирующие наборы, бизнес-данные не отличаются полнотой и, кроме того, в решении не предусмотрена обработка или компенсация ошибок. В данной версии примера добавлены некоторые данные для уточнения решения — в частности, данные, необходимые для корреляции.

Мы начнем с того, что воспользуемся инструментом IBM® Rational® RequisitePro®для того, чтобы описать бизнес-мотивацию для утверждения бизнес-задач, которые необходимо выполнить, и целей, которые должны быть достигнуты. После этого мы расскажем о высокоуровневом бизнес-процессе, записанном при помощи IBM® WebSphere® Business Modeler и выражающем организационные и функциональные бизнес-требования, необходимые для обеспечения решения задач и достижения целей. Эти мотивации и модели процессов создают контекст для идентификации сервисов, отвечающих бизнес-требованиям.

Поняв бизнес-требования, мы сможем продолжить разработку сервисов, обеспечивающих выполнение этих требований. Первый шаг в разработке SOA-решения — идентификация сервисов. Для этого мы будем рассматривать бизнес-процесс как контракт для требований к сервису. После этого разрабатываются спецификации и поставщики сервисов, которые объединяются способом, обеспечивающим выполнение бизнес- требований и одновременно решающим задачи архитектуры программного обеспечения.

Описанный процесс идентификации подходящих сервисов по бизнес-модели также известен как декомпозиция предметной области. В IBM® Rational Unified Process® (RUP®) SOMA описывается метод декомпозиции предметной области и некоторые другие методы, которые, если использовать их в сочетании друг с другом, обеспечивают повышенную точность идентификации всех сервисов, необходимых бизнесу.

Бизнес-требования

Сценарий: Некий консорциум компаний принял решение об организации совместной работы по созданию сервиса многократного использования для обработки заказов на приобретение. Задачи проекта:

Утвердить общие средства обработки заказов на приобретение;

Обеспечить своевременную обработку заказов и доставку заказанных товаров;

Способствовать минимизации наличного складского запаса и расходов на обслуживание склада;

Минимизация расходов на производство и отгрузку.

Кроме того, мы создадим ключевой показатель эффективности (key performance indicator, KPI) для задачи № 2: Своевременная обработка заказов

В качестве KPI можно использовать что-нибудь такое, что мы хотим наблюдать в модели бизнес-процесса, реализующей данный бизнес-прецедент; то, что будет служить ограничением в нашем SOA- решении; то, что будет отслеживаться в нашем Web-сервисе. Благодаря этому мы сможем осуществлять мониторинг и управление сервиса, чтобы реально обеспечить выполнение задач и достижение целей бизнеса.

Организационные бизнес-процессы

Бизнес-аналитики компаний-членов консорциума анализируют требования и выносят решение о том, что предлагаемый бизнес-процесс BM® WebSphere® Business Modeler выполняет бизнес-задачи, а также не противоречит операционным ограничениям бизнеса. Для того чтобы этот процесс можно было использовать в качестве основы официального соглашения об уровне сервиса (service level agreement, SLA) между сторонами, его необходимо доработать и детализировать. Следовательно, наше SOA-решение, удовлетворяющее этим бизнес-требованиям, должно строго следовать им

Процесс «Обработка заказов на приобретение» инициализирует три параллельных деятельности: первая — это управление планированием производства и доставки, вторая — вычисление стоимости и выставление счета, а третья — доставка заказанной продукции. Обработка начинается с инициализации вычисления стоимости заказанных продуктов. Однако это еще не окончательная стоимость, потому что общая сумма счета зависит от того, где был изготовлен продукт, и от суммы расходов на доставку. Одновременно с этим заказ пересылается на производство, чтобы определить, есть ли данный продукт в наличии и на каком складе. Параллельно с этим процесс размещает запрос на доставку, а затем ожидает получения срока доставки от отдела оперативно-производственного планирования. При наличии плановой информации составление счета может быть закончено, а счет передан клиенту.

Мы воспользовались инструментом WebSphere Business Modeler, чтобы создать бизнес-показатель Максимальное время обработки заказа, который будет соответствовать KPI Максимальное время обработки заказа из бизнес-требований. Этот бизнес-показатель отслеживает время обработки процесса «Обработка заказа на приобретение» и генерирует предупреждение в том случае, если время обработки заказа превысит пять дней (рисунок 4).

Бизнес-аналитики для имитации бизнес-процесса могут воспользоваться инструментом WebSphere Business Modeler. Он может использоваться в следующих важных целях:

Во-первых, для воспроизведения бизнес-процесса с целью формирования представления о его работе. Благодаря этому бизнес-аналитики смогут согласовать операции с различными заинтересованными лицами. Клиенты часто не знают, чего хотят, пока вы не покажете им то, чего они точно не хотят. Выполнение бизнес-процесса в режиме имитации — хороший способ получить одобрение заинтересованных лиц на ранней стадии разработки, то есть до того, как деньги будут вложены в решение ошибочно поставленной задачи. Формирование представления о бизнес-процессе через программную имитацию может также способствовать выявлению новых возможностей для уточнения процесса, которые трудно обнаружить при простом просмотре модели процесса;

Кроме того, имитацию можно использовать для того, чтобы определить затраты на выполнение процесса и время его выполнения — ключевой момент, который необходимо учитывать, принимая бизнес-решение. Бизнес-аналитик может выделить ресурсы и указать продолжительность каждой задачи в процессе. Эти ресурсы могут включать доступность, расходы, и информацию о размещении. Имитации WebSphere Business Modeler могут затем использоваться для оценки продолжительности выполнения процесса, его стоимости и предполагаемых узких мест, которые необходимо устранить. Различающиеся вследствие изменений в бизнес-процессе выполнения имитации можно легко сравнить, чтобы определить, был ли получен эффект от конкретного изменения;

И, наконец, результаты выполнений имитаций могут использоваться для того, чтобы получить оценки ключевых показателей эффективности (KPI) бизнеса на ранних фазах проекта и гарантировать, что бизнес-процессы решают поставленные задачи и обеспечивают достижение целей. Благодаря этому у коллектива разработчиков появляется уверенность в том, что они создают правильное решение правильно поставленной задачи в правильно выбранный момент времени.

Мы также связали процесс «Обработка заказов на приобретение» в WebSphere с бизнес-прецедентом BUC1 «Обработка заказов на приобретение» в RequisitePro, чтобы показать, что процесс реализует прецедент. «Реализует» — это стандартный термин моделирования прецедентов. Мы видим, что бизнес-прецедент и процесс в пиктограмме бизнес-прецедента BUC в представлении обозревателя требований Requeiment Explorer в WebSphere Business Modeler связаны при помощи маленькой стрелки

Вы можете пользоваться перспективой Requirements в WebSphere Business Modeler для перехода между бизнес-процессами и реализуемыми ими бизнес-прецедентами в представлении обозревателя требований Requirement Explorer в WebSphere Business Modeler, клиент RequisitePro или требование в Microsoft® Word®.

В следующем разделе мы рассмотрим бизнес-процесс как соглашение о требованиях и продемонстрируем, как идентифицировать сервисы в SOA-решении для выбранных операционных бизнес- требований.

Идентификация сервисов

Некоторые бизнес-процессы могут выполняться непосредственно на платформе, например, в среде сервера процессов IBM® WebSphere® Process Server. Однако существуют такие ситуации, в которых бизнес-процессы недостаточно учитывают ИТ-задачи и, возможно, должны подвергнуться рефакторингу для улучшения поддержки многократного использования и выполнения таких нефункциональных требований, как производительность, масштабируемость и безопасность. В этом случае бизнес процессы могут рассматриваться как спецификации требований к тому, какие задачи должно выполнять SOA-решение.

Требование к сервису

Требования к процессу «Обработка заказов на приобретение» могут быть получены на основе бизнес-процесса WebSphere Business Modeler. Эти требования рассматриваются как контракт для сервиса; они показывают, какие роли участвуют в бизнес-процессе, какие задачи они должны выполнять и по каким правилам взаимодействовать. Контракт для сервиса является формальной, архитектурно-нейтральной спецификацией бизнес-требований, которая создается путем взаимодействия с потребителями и поставщиками сервиса и не решает никаких вопросов, имеющих отношение к архитектуре или реализации ИТ. В этом случае контракт для сервиса содержит ту же информацию, что и исходный бизнес-процесс; он просто представляет собой представление бизнес-процесса, созданное при открытии модели WebSphere Business Modeler при помощи IBM® Rational® Softwar Modeler или IBM® Rational® Software Architect.

Бизнес-аналитики обычно используют WebSphere Business Modeler; тогда как разработчики ИТ-архитектуры предпочитают работать в Rational Software Architect. Маловероятно, что эти инструменты будут использоваться одновременно; как правило, они не используются для доступа к рабочей области одного и того же пользователя. Чтобы разработчики ИТ-архитектуры могли

использовать результаты работы бизнес-аналитиков, в Rational Software Architect предусмотрена возможность импорта бизнес-моделей WebSphere Business Modeler в рабочую область Rational Software Architect, после чего разработчик архитектуры сможет открыть модель и просмотреть содержание работы бизнес-аналитика, интерпретированное в форме эквивалентных UML-моделей. На рисунке 6 показаны избранные фрагменты нашей модели WebSphere, интерпретированные Rational Software Architect в виде UML-диаграммы.

Давайте уделим время разбору этой диаграммы, потому что мы будем работать с такими диаграммами при разработке SOA-решения.

Кооперация <<serviceCollaboration>> «Обработка заказов на приобретение» соответствует бизнес-процессу «Обработка заказов на приобретение».

Примечание:

кооперация показана в классической нотации (секционированный прямоугольник), а не в нотации, использующей эллипсы со штрихом внутри. UML 2 допускает использование любой из двух нотаций. Для данного примера мы выбрали нотацию, использующую прямоугольники, потому что в прямоугольниках больше места для ролей.

При моделировании требований к сервису или извлечении требований из бизнес- процессов кооперация отвечает на следующие вопросы:

Какой эффект должен быть достигнут при выполнении требования? Если кооперация извлекается из бизнес-процесса, то об этом свидетельствует не что иное, как имя кооперации. Такие имена часто представляют собой глагольные фразы, показывающие, что должны выполнять роли в кооперации;

Кто участвует в работе? Роли в кооперации показывают, кто участвует в кооперации. Эти роли могут соответствовать дорожкам бизнес-процесса на диаграмме swim lanes;

Каковы ответственности ролей? (Примечание: роли в UML могут выполнять не только люди, но и, например, поставщики сервиса.) Роли кооперации обычно имеют тип «интерфейс». Ответственности ролей декларируются как операции интерфейса. Любой объект в нашем решении сервиса, который выполняет роль, должен быть способен выполнять такие ответственности. То есть объекты должны быть реализациями соответствующих операций. В бизнес-процессе роли отвечают за задачи, которые находятся в соответствующей им дорожке на диаграмме swim lanes. Интерфейсы можно получить, создав список задач, присвоенных ролям в бизнес- процессе;

Как роли взаимодействуют между собой? То есть какие роли должны общаться с другими ролями? Чтобы ответить на эти вопросы, устанавливаются зависимости между ролями. Такие взаимодействия на диаграмме изображаются при помощи соединителей между ролями в кооперации. Соединители показывают, что в бизнес-процессе существует некоторый элемент управления или данных, который на диаграмме swim lanes данного бизнес-процесса повторяется в разных дорожках;

Что представляют собой правила взаимодействия ролей? Это поведение, владельцем которого является кооперация. Поведением может считаться активность, взаимодействие, конечный автомат или неопределенное поведение (например, код). Это поведение сообщает, когда

возникает необходимость в ответственности ролей, и может демонстрировать обмен информацией между ними. Активность соответствует UML-видению бизнес- процесса;

Как понять, были ли выполнены требования? Кооперация может иметь ограничения, определяющие условия, которые должны быть истинными для выполнения требований. Ограничения соответствуют ключевым показателям эффективности (KPI) бизнес-показателей в бизнес-процессе.

«Обработка заказов на приобретение» — это кооперация сервиса, включающая четыре роли, одну из которых выполняет сам процесс. Эти роли получены на основе бизнес- процесса путем изучения задач, присвоенных ролям в дорожках на диаграмме бизнес-процесса. Инструмент WebSphere Business Modeler использует ориентированное на процессы представление бизнес-требований, тогда как в контракте для сервиса используется представление, ориентированное на роли. Это обеспечивает более сервис-ориентированное видение контрактов для бизнеса и упрощает переход от требований к процессу к архитектуре сервис-ориентированных решений.

Соглашение о сервисе может иметь ограничения, полученные на основе параметров и показателей модели мотивации бизнеса. Эти ограничения показывают, для чего предназначен контракт для сервиса и как оценить его успешность.

Кооперация сервиса может также реализовать прецеденты, которые документируют дополнительную информацию о функциях высокого уровня и нефункциональных требованиях к бизнес-процессу с точки зрения ключевых сторонних заинтересованных лиц или исполнителей. Эти прецеденты можно увидеть на диаграммах прецедентов в Rational Software Architect и связать с прецедентами в RequisitePro. Это поддерживает связь между контрактом для требований к сервису, бизнес-процессами, бизнес-прецедентами и бизнес-задачами и бизнес-целями.

Контракт для кооперации сервиса, показанный на рисунке 6, может быть представлением модели бизнес-процесса WebSphere Business Modeler или быть разработанным непосредственно в Rational Software Architect. Какой из двух подходов использовать зависит от предпочтений, специалиста, выполняющего моделирование, а также от того, используется ли для проверки корректности бизнес-процессов имитация. В любом случае поддерживается полная отслеживаемость. Безусловно, качество контрактов для кооперации сервисов, полученных на основе бизнес-процессов WebSphere Business Modeler будет зависеть от уровня детализации этих процессов. Взаимосвязь между моделированием бизнес-процессов и моделированием сервиса в более полном объеме рассматривается в статье Business services modeling (Моделирование бизнес-сервисов) на Web-сайте developerWorks

Организация проекта сервиса

Мы готовы к моделированию сервиса, удовлетворяющего описанным требованиям. Наше решение будет выполнять эти требования, но не обязательно ограничиваться ими. При проектировании SOA-решения необходимо идентифицировать сервисы, разработать их интерфейсы и решить, как они будут реализованы: какой поставщик сервисов будет предоставлять сервисы, какие именно сервисы и каким образом. Благодаря этому мы установим зависимости между потребителями и поставщиками сервисов и связанность в системе. Управление связанностью — основной компонент проектирования сервисов на базе SOA. Разделив бизнес-требования и сервисы, мы получим гибкость, которая позволит спроектировать SOA, отвечающую как бизнес-требованиям, так и требованиям к ИТ-системе. Этим мы

предохраним бизнес-требования от избыточности ограничений SOA-решений, которые могли бы сократить многократное использование сервисов и привести к снижению динамичности бизнеса.

Сначала мы создадим проект моделирования сервиса в Rational Software Architect с именем PurchaseOrderProcessModel. Этот проект содержит модель сервисов PurchaseOrderProcess. Данная модель состоит из информационной модели для сервисов, данных сервисов, спецификаций имеющихся и необходимых интерфейсов и компонентов, предоставляющих требуемые сервисы. Но в этой статье мы сосредоточимся, главным образом, на идентификации сервисов.

модель PurchaseOrderProcess состоит из 5 основных пакетов:

пакет org::ordermanagement содержит элементы, имеющие отношение к управлению заказами;

пакет org::crm содержит модели данных и общие интерфейсы для некоторых предписанных стандартов Управления взаимоотношениями с клиентами, о которых потребители и поставщики сервисов договорились для разработки совместно используемых сервисов;

пакет com::acme::credit содержит элементы, относящиеся к выставлению счетов и управлению кредитами;

пакет com::acme::productions содержит элементы, относящиеся к производству и планированию;

пакет com::acme::shipping содержит элементы, относящиеся к доставке.

Пакеты делят предметную область задачи на различные функциональные области. Это помогает контролировать сложность, создает необходимые пространства имен, способствует многократному использованию и хранит разделенные задачи в разных пакетах (см. рисунок 7).

Такую организацию можно назвать преобладающей организацией моделей сервиса. Чтобы документировать другие классификации для организации тех же элементов моделей, можно использовать секции.

Топология сервисов

Мы начнем моделирование сервиса с идентификации сервисов, принимающих участие в обработке заказов на приобретение. Пока мы не выясняем детали того, что делают сервисы и как они взаимодействуют. Нам просто нужно создать упрощенную схему, показывающую, что представляют собой сервисы, и вывести концепцию высокого уровня о том, как они могли бы взаимодействовать.

На рисунке 8 представлена простая диаграмма классов (или диаграмма в свободной форме) в Rational Software Architect; эта диаграмма показывает пакеты, представляющие описанные выше функциональные области бизнеса, и ключевые сервисы, которые нам предстоит создать в каждом пакете. Единственной информацией о сервисе на этот момент является имя сервиса.

Зависимости на этой диаграмме используются для того, чтобы показать, как сервисы должны быть связаны друг с другом. На данном этапе у нас нет полной спецификации сервиса и мы не знаем, какие участники будут предоставлять или запрашивать сервисы, и какие именно сервисы. Кроме того, пока не смоделирован ни один из этих сервисов. Следовательно, пока еще нет формальной декларации о том, для чего предназначены эти зависимости. Это просто обозначение реализаций-участников, которые могут присутствовать, а могут и не присутствовать в окончательных

спецификациях и реализациях сервисов. Однако для этой высокоуровневой схемы такие зависимости представляют определенную ценность, потом что именно с них начинается идентификация прецедентов и связанности между системами, управлению которыми необходимо уделить особое внимание.

Примечание:

В этой статье мы не рассказываем о том, как спроектировать и разработать топологию сервиса Подключаемый модуль IBM RUP for SOMA (см. раздел Ресурсы) содержит процесс, методы и руководство по обнаружению сервисов по функциональным областям.

Существует простой способ обнаружения необходимых сервисов — изучить бизнес- требования, идентифицированные как кооперации сервиса, и подумать о том, какие поставщики сервисов потребуются для выполнения ролей в этой кооперации. Именно этот способ используется в следующем простом примере. Мы можем оформить его в виде диаграммы, показывающей, как спецификации сервиса будут использоваться для выполнения контракта для кооперации сервиса.

На рисунке показан абстрактный компонент, моделирующий контекст для обработки заказов на приобретение. Он состоит из нескольких частей, представляющих собой спецификации сервиса, о которых мы рассказывали ранее. Наша диаграмма демонстрирует возможное использование каждой из спецификаций сервиса в конкретном контексте. Нам все еще неизвестно, какие участники предоставляют или используют сервисы, мы знаем только, что делают некоторые участники. Описанные части также являются абстрактными, потому что они представляют собой экземпляры спецификации сервиса, а не экземпляры поставщиков сервиса, предоставляющих эти сервисы.

Тем не менее, из этой диаграммы видно, как сервисы выполняют контракт для требований к кооперации сервиса. Контракт использует кооперацию и является экземпляром кооперации сервиса «Обработка заказов на приобретение». Части компонента «Обработка заказов на приобретение» привязаны к ролям, которые они выполняют в этом контракте. Благодаря этому обеспечивается формальная обратная связь с бизнес-требованиями и демонстрируется значимость для бизнеса каждого из интерфейсов сервисов, которые мы собираемся определить и реализовать. Поведение кооперации сервиса «Обработка заказов на приобретение» определяет, как компоненты сервиса должны взаимодействовать в реализации сервиса. Мы вернемся к этому моменту после определения и реализации сервиса, чтобы убедиться, что решение выполняет требования контракта.

Что дальше

В этой статье мы рассказали о методе идентификации сервисов, обеспечивающих выполнение бизнес-требований. Мы начали с выяснения бизнес-задач и бизнес-целей, способствующих осуществлению бизнес-миссии. Затем мы смоделировали бизнес- операции и бизнес-процессы, необходимые для выполнения задач и достижения целей. После этого мы стали рассматривать бизнес-процесс как кооперацию сервиса, представляющую собой контракт для бизнес-требований к сервису, который должен выполняться нашим потенциальным решением. Затем мы воспользовались этим контрактом для требований как вспомогательным материалом для идентификации необходимых сервисов и возможных связей между ними. Это обычный способ идентификации значимых для бизнеса сервисов, ассоциируемых с бизнес-задачами и бизнес-целями, которые предполагается решить.