Распределенные информационно-аналитические системы

Лекция № 16.

Распределенные системы на базе Web

Профессор кафедры КБ-2: д.т.н. Шатовкин Р.Р.

Учебные цели:

Изучить основы научных знаний по основам Web-сервисов; преимуществам и недостаткам Web-сервисов; взаимодействию с Web-сервисами; технологии Web-сервисов и примеру ее использования; определению сервисно-ориентированной архитектуры, ее преимуществам и требованиям к ней; стеку технологий Web-сервисов; принципам взаимодействия Web-сервисов в рамках сервисно-ориентированной архитектуры.

Учебные вопросы:

1. Основы Web-сервисов.

2. Преимущества и недостатки Web-сервисов.

3. Взаимодействия с Web-сервисами.

4. Технология Web-сервисов и пример ее использования.

5. Определение сервисно-ориентированной архитектуры, ее преимущества и требования к ней.

6. Стек технологий Web-сервисов.

7. Принципы взаимодействия Web-сервисов в рамках сервисно-ориентированной архитектуры.

1. Основы Web-сервисов

Назовем ***сервисом (service)*** ресурс, реализующий бизнес-функцию, обладающий следующими свойствами:

- является повторно используемым;

- определяется одним или несколькими явными технологически-независимыми интерфейсами;

- слабо связан с другими подобными ресурсами и может быть вызван посредством коммуникационных протоколов, обеспечивающих возможность взаимодействия ресурсов между собой.

Web-сервисом называется программная система, идентифицируемая строкой ***URI***, чьи интерфейсы и привязки определены и описаны посредством ***XML***. Описание этой программной системы может быть найдено другими программными системами, которые могут взаимодействовать с ней согласно этому описанию посредством сообщений, основанных на XML, и передаваемых с помощью Интернет-протоколов.

Web-сервисы – это новая перспективная архитектура, которая обеспечивает новый уровень распределенности. Вместо разработки или приобретения компонентов и их встраивания в ИАС предлагается покупать время их работы и формировать программную систему, которая осуществляет вызовы методов из компонентов, которые принадлежат и поддерживаются независимыми провайдерами. Благодаря **Web-сервисам** функции любой программы в сети могут стать доступными через Интернет. Самый простой пример **Web-сервиса** – система **Passport** на **Hotmail**, которая позволяет создать аутентификацию пользователей на собственном сайте.

В основе Web-сервисов лежат следующие универсальные технологии:

- TCP/IP – универсальный протокол, понимаемый всеми сетевыми устройствами, от мэйнфреймов до мобильных телефонов и PDA;

- HTML – универсальный язык разметки, применяемый для отображения информации устройствами пользователей;

- XML (Extensible Markup Language) – универсальный язык для работы с любыми типами данных.

Универсальность этих технологий – основа для понимания Web-сервисов. Они основаны только на общепринятых, открытых и формально независимых от поставщиков технологиях. Только посредством этого достигается главное преимущество Web-сервисов как концепции построения распределенных ИАС – их универсальность, то есть возможность применения для любых операционных систем, языков программирования, серверов приложений и т.д.

Таким образом, Web-сервисы решают исходную задачу – задачу интеграции приложений различной природы и построения распределенных ИАС. В этом и заключается основное принципиальное отличие Web-сервисов от предшественников.

Web-сервисы – это **XML-приложения**, осуществляющие связывание данных с программами, объектами, базами данных либо с деловыми операциями целиком. Между Web-сервисом и программой осуществляется обмен XML-документами, оформленными в виде сообщений. Стандарты Web-сервисов определяют формат таких сообщений, интерфейс, которому передается сообщение, правила привязки содержания сообщения к реализующему сервис приложению и обратно, а также механизмы публикации и поиска интерфейсов.

XML (*Xtensible Markup Language* – расширяемый язык разметки) рекомендован Консорциумом Всемирной паутины (W3C). Спецификация XML описывает XML-документы и частично описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому). XML разрабатывался как язык с простым формальным синтаксисом, удобный для создания и обработки документов программам и одновременно удобный для чтения и создания документов человеком, с подчеркиванием нацеленности на использование в Интернете. Язык называется расширяемым, поскольку он не фиксирует разметку, используемую в документах: разработчик волен создать разметку в соответствии с потребностями к конкретной области, будучи ограниченным лишь синтаксическими правилами языка. Сочетание простого формального синтаксиса, удобства для человека, расширяемости, а также базирование на кодировках Юникод для представления содержания документов привело к широкому использованию, как, собственно, XML, так и множества производных специализированных языков на базе XML в самых разнообразных программных средствах.

XML можно использовать не только для описания отдельного документа. Индивидуальный пользователь, компания или комитет по стандартам может определить необходимый набор элементов XML и структуру документа, которые будут применяться для особого класса документов. Подобный набор элементов и описание структуры документа называют **XML-приложением** или **XML-словарем**. Например, организация может определить XML-приложение для создания документов, описывающих молекулярные структуры, мультимедиа презентации или содержащих векторную графику.

Web-сервисы могут использоваться во многих приложениях. Независимо от того, откуда запускаются Web-сервисы, с настольных компьютеров клиентов или с переносных, они могут использоваться для обращения к таким интернет-приложениям, как система предварительных заказов или контроля выполнения заказов.

Web-сервисы пригодны для **B2B-интеграции (business-to-business),** замыкая приложения, выполняемые различными организациями, в один производственный процесс.

Web-сервисы также могут решать более широкую проблему интеграции приложений предприятия (Enterprise Application Integration – EAI), осуществляя связь нескольких приложений одного предприятия с несколькими другими приложениями. Во всех перечисленных случаях технологии Web-сервисов являются «связующим звеном», объединяющим различные части программного обеспечения.

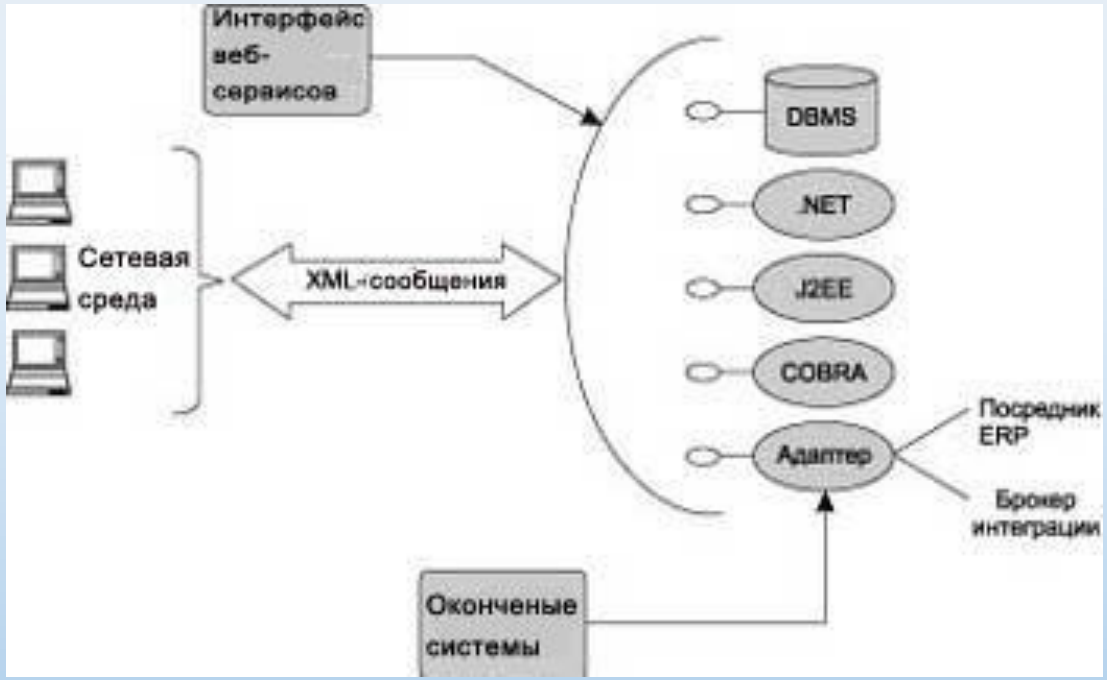

Рисунок 1 – Web-сервисы взаимодействуют с прикладными системами

Как видно из рисунка 1, Web-сервисы представляют собой оболочку, обеспечивающую стандартный способ взаимодействия с прикладными программными средами, такими как системы управления базами данных (СУБД), .NET, J2EE (Java2 Platform, Enterprise Edition), CORBA (Common Object Request Broker Architecture), посредники пакетов планирования ресурсов предприятия (Enterprise Resource Planning, ERP), брокеров интеграции и пр.

Интерфейсы Web-сервисов получают из сетевой среды **стандартные XML-сообщения**, преобразуют **XML-данные** в формат, «понимаемый» конкретной прикладной программной системой, и отправляют ответное сообщение (последнее – не обязательно). Программная реализация Web-сервисов (базовое программное обеспечение, нижний уровень) может быть создана на любом языке программирования с использованием любой операционной системы и любого **связующего программного обеспечения (middleware)**.

Использование Web-сервисов очень выгодно с коммерческой точки зрения. За счет повсеместного распространения Web-сервисов Интернет становится более эффективным, особенно при осуществлении коммерческих сделок. Сочетая прямой доступ к программным приложениям и коммерческим документам, Web-сервисы следующего поколения Сети обеспечат полностью автоматическое взаимодействие, что позволит обращаться непосредственно к данным программ, игнорируя знакомые Web-страницы. Более того, основные компоненты Web-сервисов, скорее всего, будут предоставляться и публиковаться множеством различных компаний, специализирующихся на отдельных функциональных элементах (проверка полномочий, координация сделок, ведение счетов). Это обеспечит непосредственное взаимодействие «приложение-приложение» – принцип, лежащий в основе Web-сервисов и определяющий их суть и реализацию.

Технология Web-сервисов существует на очень высоком уровне абстракции, позволяя поддерживать множество одновременных определений, которые иногда бывают противоречивы. На простейшем уровне Web-сервисы могут восприниматься как интернет-ориентированные текстовые брокеры интеграции. Любые данные могут преобразовываться в ASCII-текст и обратно, и этот подход в течение долгого времени был общим знаменателем для систем графического вывода и систем управления базами данных. Ориентированные на использование текста системы также лежат в основе успешного развития Интернета, на котором базируется дополнительная абстракция Web-сервисов. Любой компьютер или операционная система может поддерживать HTML, браузеры и Web-сервисы; и при получении по сети файлов им совершенно безразлично и даже неизвестно, с каким типом прикладной системы они взаимодействуют.

2. Преимущества и недостатки Web-сервисов

Web-сервисы поддерживают несколько способов обмена сообщениями. Уровень абстракции, на котором оперируют Web-сервисы, подразумевает такие стили взаимодействия, как эмуляцию **удаленного вызова процедуры (Remote Procedure Call, RPC)**, асинхронный обмен сообщениями, однонаправленную передачу сообщений, широковещание и публикацию/подписку.

Основные СУБД, такие как Oracle, SQL Server и DB2, поддерживают анализ XML и службы преобразования, обеспечивая непосредственное взаимодействие между Web-сервисами и СУБД. Производители связующего программного обеспечения обычно также предоставляют возможность привязки Web-сервисов к своим программным системам (серверам приложений и брокерам интеграции).

Следовательно, для пользователя взаимодействие с Web-сервисами может проявляться в интерактивной или пакетной форме, поддерживающей синхронную и асинхронную модели связи; а также как пользовательский интерфейс, написанный с использованием Java, VB (Visual Basic), офисных приложений, браузеров или «толстых» клиентов СУБД.

Такое взаимодействие может привязываться к любому типу базовой (более низкого уровня) программной системы.

К преимуществам Web-сервисов можно отнести следующее:

1. Web-сервисы позволяют компании интегрировать свои бизнес-процессы с бизнес-процессами бизнес-партнеров и клиентов при меньшей стоимости, чем с использованием других интеграционных технологий. Стоимость подобных решений на основе Web-сервисов доступна даже для SMB (Small and Medium Business), что откроет для таких компаний новые перспективы развития.

2. Поскольку Web-сервисы организуются в публичные реестры (UDDI-реестры, ebXML-реестры или иные), доступные заинтересованным лицам по всему миру, порог выхода компаний на новые рынки снижается, возможности же для наращивания клиентской базы напротив возрастают.

3. Web-сервисы обеспечивают преемственность в отношении уже имеющихся в компании ИАС, то есть можно сказать, что Web-сервисы надстраиваются *над* существующими ИАС, но не *вместо* них. Таким образом, обеспечивается сохранность уже сделанных инвестиций в IT-инфраструктуру и не идет увеличения требуемых, поскольку нет необходимости в радикальных изменениях.

4. Построение новых корпоративных решений с применением Web-сервисов реализуется быстрее и совокупно дешевле, поскольку основное внимание сосредотачивается на создании бизнес-логики решения, программирование самих Web-сервисов лишь по необходимости «обрамляет» этот процесс, не требуя больших трудозатрат за счет эффективного применения повторно используемого кода и адаптированных средств разработки (IDE и SDK).

К недостаткам Web-сервисов можно отнести:

1. Стандарты интеграции бизнес-процессов, вопросы управления транзакциями и выработка единых бизнес- и IT-политик взаимодействующих посредством Web-сервисов компаний находятся пока на стадии разработки (мы отметим следующие начинания: Web Services Flow Language (WSFL), Business Process Execution Language 4 Web Services (BPEL4WS корпорации IBM, XLANG корпорации Microsoft и спецификации WS-Coordination и WS-Transaction – результат сотрудничества IBM, Microsoft и BEA). Очевидно, без их четкой формализации и опубликования построение ИАС на основе Web-сервисов может идти лишь с переменным успехом.

2. Динамическое использование информации бизнес-реестров Web-сервисов, вызов Web-сервисов, требует решения вопросов доверительности отношений между различными бизнес-реестрами. Кроме того, есть трудности в совместном использовании бизнес-реестров различных форматов (например, задача поиска определенного Web-сервиса в UDDI-реестре и ebXML-реестре требует различных подходов в силу различия XML-документов, описывающих один и тот же Web-сервис в каждом из этих реестров. Хотя, надо отметить, что есть попытки решить эту проблему созданием единого браузера реестров. В качестве примера - графическая утилита Registry Browser корпорации Sun Microsystems, реализующая набор интерфейсов JAXR (Java API for XML Registries).

3. Добавление к функциям сервера приложений функциональности провайдера Web-сервисов в силу новизны технологий может представлять определенную трудность.

4. Вопросы безопасности функционирования ИАС на основе Web-сервисов пока не урегулированы до конца. Спецификация WS-Security – продукт деятельности корпораций IBM и Microsoft – в настоящее время достаточно молода, не «устоялась» и частично все еще дорабатывается. Однако, в силу общности положений спецификации WS-Security, уже готовится к выпуску следующий слой спецификаций, посвященных вопросам безопасности: Web Services Policy Assertions, Web Services Policy Attachments, Web Services Policy Framework, Web Services Trust, Web Services Secure Conversation, Web Services Federation.

3. Взаимодействия с Web-сервисами

Web-сервисы выполняют RPC- и документно-ориентированное взаимодействия. Стандарты и технологии Web-сервисов обычно подразумевают ***два основных типа моделей*** взаимодействия приложений:

- удаленный вызов процедуры (онлайновая);
- документно-ориентированный (пакетная).

RPC-ориентированные взаимодействия удобны для краткого обмена данными. В RPC-ориентированном взаимодействии запросы Web-сервисов приобретают форму вызова метода или процедуры с соответствующими входными или выходными параметрами. В отличие от документно-ориентированного взаимодействия, RPC-ориентированное взаимодействие производит отправку документа, специально отформатированного для передачи в отдельную логическую программу или базу данных (рисунок 2).

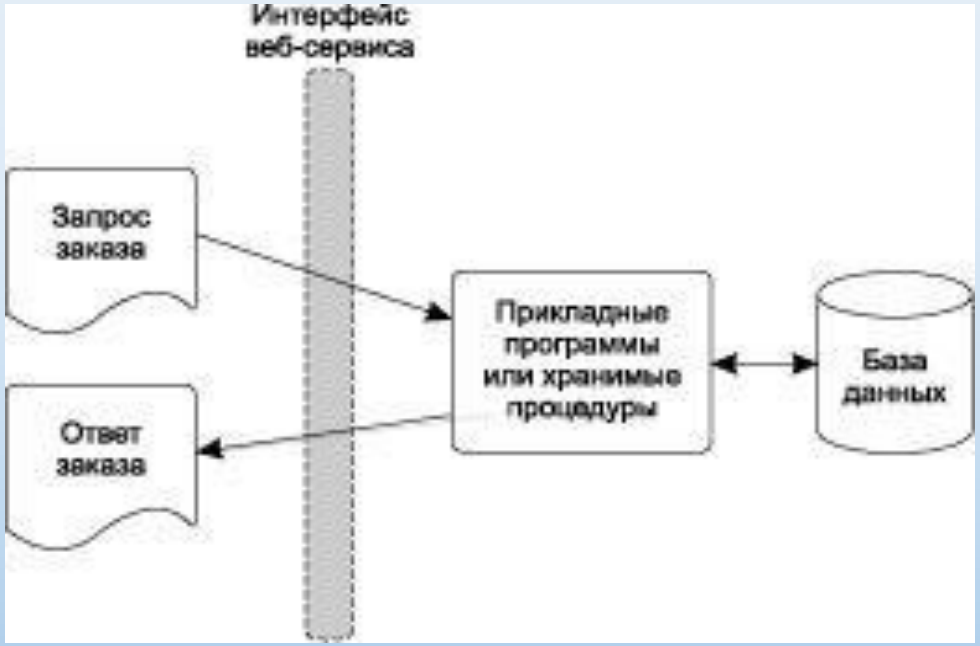

Рисунок 2 – Web-сервисы поддерживают интерактивный заказ в форме запроса/ответа

Документно-ориентированные взаимодействия удобны для обмена большими объемами данных. При документно-ориентированном взаимодействии запросы Web-сервиса имеют форму завершенного XML-документа, предназначенного для обработки целиком.

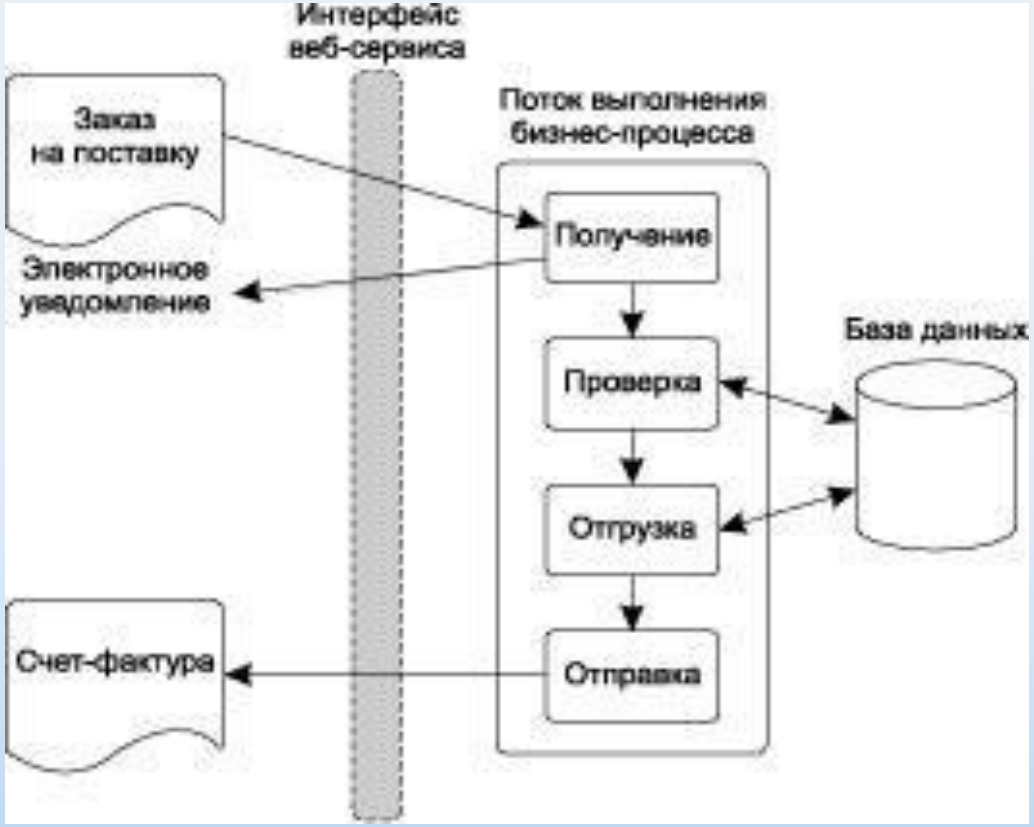

Рисунок 3 – Web-сервис обрабатывает полный заказ на поставку

Документно-ориентированные взаимодействия зачастую предполагают, что использующие Web-сервисы стороны заранее согласовали порядок оформления общих документов, таких как заказ на приобретение, счет за доставку или общий счет. Эти стороны обычно идентифицируются как «торговые партнеры» или «сотрудничающие партнеры». Торговые партнеры также согласовывают общий поток выполнения процесса или модель взаимодействия при обмене документами, например, оговаривают необходимость подтверждения квитанции заказа на приобретение, передачу специальной информации о состоянии в ответ на запрос заказа или отправку сигнала оповещения по электронной почте после отгрузки заказа. В ходе реализации бизнес-процесса необходим обмен полными документами. Если до этого документ содержал общую, фрагментированную информацию, то теперь требуется согласованное заполнение специальных разделов, таких как цена покупки или обязательная дата доставки (рисунок 3).

4. Технология Web-сервисов и пример ее использования

Порядок описания, поиска и взаимодействия Web-сервисов друг с другом определяют стандарты. Взаимодействующие через Интернет программы должны уметь обнаруживать друг друга, находить информацию, позволяющую им осуществить связь, понимать, какая модель должна быть применена (простая, типа «запрос/ответ», или более сложная последовательность), и договариваться об использовании таких услуг, как защита информации, подтверждение передачи сообщений и составление сделок. Некоторые из этих сервисов реализуются существующими технологиями и предлагаемыми стандартами, а другие – нет.

Использующее Web-сервисы сообщество стремится удовлетворить все эти требования, но это эволюционный процесс, как и сам Интернет. С самого начала инфраструктура и стандарты Web-сервисов подразумевали возможность расширения (так же как до них XML и HTML), что позволяет использовать их сразу же после появления новых стандартов и технологий.

Web-сервисы требуют использования нескольких смежных XML-технологий:

Язык XML – фундамент, на котором строятся Web-сервисы. Он предоставляет язык определения данных и порядок их обработки. XML представляет семейство связанных спецификаций, публикуемых и поддерживаемых интернет-консорциумом (World Wide Web Consortium, W3C) и другими организациями.

SOAP (Simple Object Access Protocol), разработанный консорциумом W3C, определяет формат запросов к Web-сервисам. Сообщения между Web-сервисом и его пользователем пакуются в так называемые SOAP-конверты (SOAP envelopes, иногда их еще называют XML-конвертами). Само сообщение может содержать либо запрос на осуществление какого-либо действия, либо ответ – результат выполнения этого действия.

WSDL (Web Services Description Language) – технология, основанная на XML, определяющая интерфейсы Web-сервисов, типы данных и сообщений, а также модели взаимодействия и протоколы связывания. Перед развертыванием сервиса разработчик составляет его описание на языке WSDL, указывает адрес Web-сервиса, поддерживаемые протоколы, перечень допустимых операций, форматы запросов и ответов.

Технология UDDI (Universal Description, Discovery and Integration) – реестр Web-сервисов и механизм поиска. Он используется для хранения и упорядочения деловой информации, а также для нахождения указателей на интерфейсы Web-сервисов.

Пример использования. Стандарты Web-сервисов обычно используются совместно и согласованно. После обнаружения WSDL в UDDI или другом месте генерируется SOAP-сообщение для отправки на удаленный сайт.

Как видно из рисунка 4, при предоставлении документа по адресу Web-сервиса программа использует XML-схему определенного типа (такую как WSDL), позволяющую преобразовать данные из ее входного источника (в этом примере структурированный файл) и на основе того же WSDL-файла создать экземпляр XML-документа в формате, согласованном с целевым Web-сервисом. WSDL-файл используется для определения как входного, так и выходного преобразования данных.

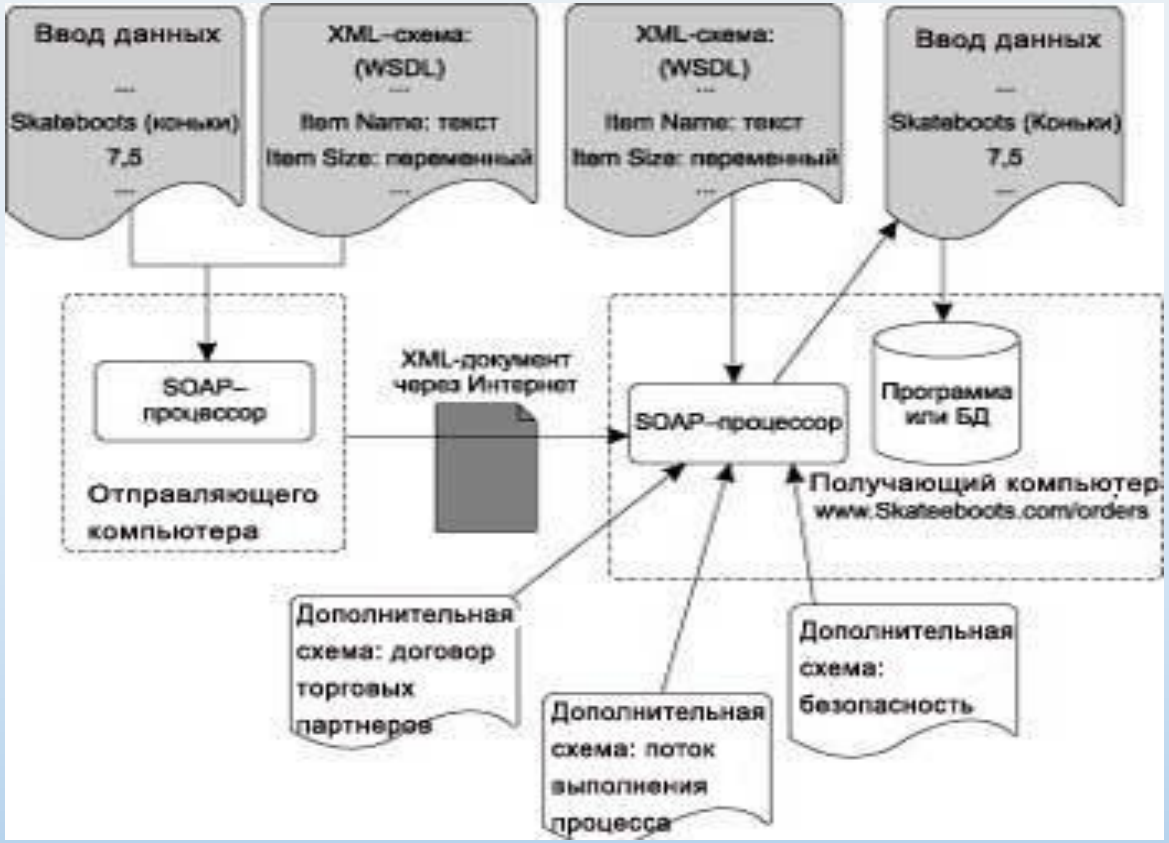

Рисунок 4 – Web-сервисы используют XML-документы и осуществляют преобразование данных

SOAP-процессор отправляющего компьютера преобразует данные из собственного («родного») формата в тип данных, предопределенный в соответствии с содержащейся в WSDL-файле XML-схемой на основе таблиц преобразования для текстов, значений с плавающей точкой и других данных. Таблицы преобразования «связывают» собственные типы данных с соответствующими конкретной XML-схеме. (Стандартное преобразование типов широко используется в Java, Visual Basic, CORBA и других известных системах. Многие средства XML позволяют настраивать специальные преобразования типов.) SOAP-процессор получающего компьютера выполняет обратное преобразование данных из типов XML-схемы в собственные типы данных.

Файлы описаний Web-сервисов обычно регистрируются с помощью URL. URL, повсеместно используемый в Сети, указывает на IP-адрес, соответствующий Web-ресурсу. Схемы Web-сервисов являются одной из форм Web-ресурса, они содержатся в доступных через Интернет файлах и к ним применим тот же механизм, что используется при загрузке HTML-файлов. Главное отличие между загрузкой HTML-файла и обращением к ресурсу Web-сервиса заключается в том, что Web-сервис оперирует XML-документами, а не HTML-документами и опирается на соответствующие технологии, такие как использование схем, преобразование, проверка подлинности, что и обеспечивает поддержку удаленного соединения приложений. Но способ, согласно которому схемы Web- сервисов публикуются и загружаются, одинаков: HTTP-операция по указанному URL.

Для проверки достоверности сообщений Web-сервисы используют XML-схемы. После получения документа реализация Web-сервиса сначала должна проанализировать XML-сообщение и удостовериться в корректности данных, выполнить проверку качества услуг (Quality-of-Service), такую как проверку политики безопасности или соглашений торговых партнеров, а затем произвести последовательность связанных с данным документом коммерческих операций. Web-сервис на вымышленном нами сайте skateboots.com размещен в папке skateboots.com/ order, на которую и указывает URL.

Web-сервис, доступный по данному интернет-адресу, идентифицируется с помощью публичного WSDL-файла, который может быть загружен на отправляющий компьютер и использоваться при генерации сообщения. Компания Skateboots Company также осуществляет отправку в общедоступный каталог UDDI-листинга, позволяющего клиентам находить компанию с помощью технологии UDDI. В общем случае, любой, кто хочет взаимодействовать с Web-сервисом, размещающим или контролирующим по Сети заказы для Skateboots Company, для генерации сообщения должен найти способ получения и использования WSDL-файла.

Размещенные по адресу skateboots.com программы представляют собой прослушивающий HTTP-процесс, связанный с соответствующими Web-сервисами для распознавания XML-сообщений, определенных в данном формате. Эти программы включают в себя XML-анализаторы и преобразователи. Кроме того, они осуществляют конвертацию данных SOAP-сообщения в форматы, необходимые для системы ввода заказов компании Skateboots Company.

Технологии Web-сервисов сформировались из основных структур. Этих технологий достаточно для построения, развертывания и публикации базовых Web-сервисов. На самом деле, необходим лишь базовый протокол SOAP. С момента появления Web-сервисов к ним все время добавляются другие технологии. Хотя для деловой связи, а также для создания «моста» с несовместимыми технологиями вполне достаточно базовых принципов, данная форма Web-взаимодействия тем не менее была одобрена очень быстро.

5. Определение сервисно-ориентированной архитектуры, ее преимущества и требования к ней

С функциональной точки зрения бизнес-приложение распадается на совокупность взаимодействующих между собой сервисов. Эту совокупность взаимодействующих сервисов можно отождествить с еще одним ключевым понятием – сервисно-ориентированной архитектурой.

Компонентная модель, состоящая из отдельных функциональных модулей приложений, называемых сервисами, имеющих определенные согласно некоторым общим правилам интерфейсы и механизм взаимодействия между собой, называется **сервисно-ориентированной архитектурой (Service-Oriented Architecture – SOA)**.

Переходя в мир абстракций, можно дать следующее, более сильное, определение сервисно-ориентированной архитектуры.

Архитектура приложений, в рамках которой все функции приложения являются независимыми сервисами с четко определенными интерфейсами, которые можно вызывать в нужном порядке с целью формирования бизнес-процессов, называется **сервисно-ориентированной архитектурой**.

На сегодняшний день устоявшегося, принятого IT-сообществом, определения SOA нет. Здесь мы приводим определение, которое, на наш взгляд, наиболее полно и точно отражает современное состояние этой концепции).

Поясним второе определение:

- **«все функции приложения»** – как мы уже упомянули, любое приложение с функциональной точки зрения может быть представлено совокупностью функций; ресурс, реализующий функцию, есть сервис. Таким образом, приведенное определение требует для представления и реализации любого приложения в рамках SOA проведения его полной декомпозиции до уровня отдельных функций;

- **«являются независимыми сервисами»** – в понятие независимости сервиса вкладывается следующий смысл: сервисы функционируют независимо от других информационных систем, являются функционально самостоятельными объектами. Они представляют собой «черные ящики» для любых внешних приложений: внешние приложения не знают, как сервис формирует из входных данных выходные. Все, что им известно - что необходимо подать на вход сервиса и что следует ожидать на его выходе;

- **«с четко определенными интерфейсами»** – функция (или функции), которую реализует данный сервис, должна быть однозначно описана согласно определенным, принятым для всех сервисов, правилам. Должен быть описан набор и типы входных данных, а также набор и типы выходных данных;

- **«с ... интерфейсами, которые можно вызывать»** – данное требование обусловлено необходимостью обеспечения взаимодействия между различными сервисами: для внешних по отношению к сервису информационных систем не должно иметь значения на каком языке программирования реализован сервис (точнее, здесь, Web-сервис), на какой программно-аппаратной платформе он функционирует, локально или удаленно он расположен. Внешняя информационная система должна иметь возможность взаимодействовать с сервисом (то есть передать ему входные данные и получить выходные) вне зависимости от указанных его особенностей.

Преимущества сервисно-ориентированной архитектуры. Концепции SOA и Web-сервисов не новы; на протяжении последних десяти лет были и другие технологии построения распределенных информационных систем, например, CORBA и DCOM. Однако, со временем решения, построенные на их основе перестали удовлетворять основным требованиям, предъявляемым к бизнес-решениям: максимально низкая TCO, максимально высокий ROI, короткий цикл разработки и внедрения информационной системы, широкие возможности интеграции разнородных систем.

В отличие от известных моделей технологии Web-сервисов и SOA:

- являются open-source;

- основаны на общепринятых и общеупотребимых технологиях и стандартах: XML, HTTP, TCP/IP;

- позволяют достичь наилучших показателей TCO и ROI, нежели какие-либо иные существующие на сегодняшний день интеграционные технологии;

- находятся на уровне зрелости, необходимом и достаточном для успешного применения большим количеством ISV (Independent Software Provider) по всему миру.

Требования к сервисно-ориентированной архитектуре. SOA, будучи практической концепцией, должна соответствовать определенным требованиям, предъявляемых к ней современным состоянием бизнес-отношений и информационных технологий, а также тенденциями их совместного развития:

1) обеспечивать преемственность инвестиций в IT, сохранение существующих информационных систем и их совместное эффективное использование для повышения ROI от IT-вложений;

2) обеспечивать реализацию различных типов интеграции:

- *пользовательская интеграция (user integration)* – обеспечение взаимодействия информационной системы с конкретным пользователем;
- *интеграция приложений (application connectivity)* – обеспечение взаимодействия приложений;
- *интеграция процессов (process integration)* – интеграция бизнес-процессов;
- *информационная интеграция (information integration)* – интеграция с целью обеспечения доступности информации и данных;
- *интеграция новых приложений (build to integrate)* – интеграция новых приложений и сервисов в существующие информационные системы.

3) обеспечивать поэтапность внедрения вновь созданных и миграции существующих информационных систем;

4) иметь стандартизованную технологическую обеспеченность реализации и инструментарий разработки, предоставляющие наилучшие возможности повторного использования приложений, внедрения новых и миграции существующих информационных систем;

5) позволять реализацию различных моделей построения информационных систем, в особенности таких как портальные решения, grid-системы.

Сегодняшний уровень развития SOA позволяет утверждать, что все указанные требования в той или иной мере выполняются.

Необходимо отметить, что **SOA – не есть синоним Web-сервисов**, а **Web-сервисы – не есть единственный способ реализации SOA**. SOA не является технологией или набором технологий, это - концепция, абстрактное представление реализации информационных систем с помощью сервисов безотносительно конкретных технологий. Как нетрудно заметить, в SOA присутствуют элементы объектного подхода к построению информационных систем: декомпозиция (приложений на отдельные функции) и инкапсуляция (сервисы как «черные ящики»). Однако, подчеркнем, термин «объектно-ориентированный» по отношению к SOA не является корректным, так как в SOA отсутствуют все необходимые элементы объектно-ориентированной парадигмы. Более правильно называть SOA концепцией, использующей объектный подход.

Web-сервисы, в свою очередь, являются лишь технологиями, с помощью которых можно эффективно реализовать сервисно-ориентированную архитектуру. Существуют и другие технологии, с применением которых можно реализовать SOA – например, упомянутая выше CORBA.

6. Стек технологий Web-сервисов

Концепция сервисно-ориентированной архитектуры подразумевает реализацию бизнес-процессов предприятия в виде совокупности сервисов, взаимодействующих друг с другом либо с пользователями в определенной последовательности и в соответствии с определенными правилами. Отметим, что, вообще говоря, это весьма нетривиальная задача – определить эту «определенную последовательность» и «определенные правила» таким стандартным для разработки приложений образом, чтобы охватить весь спектр существующих сценариев взаимодействия бизнес-объектов, учитывая исторически сложившееся многообразие технической и технологической реализации этого взаимодействия и самих бизнес-объектов. Решить эту задачу в рамках какой-либо единой технологии пока не удалось. И хотя тенденция консолидации существующего множества технологий Web-сервисов в настоящее время налицо, для реализации сервисно-ориентированных архитектур с помощью Web-сервисов сейчас применяется совокупность технологий, образующих так называемый ***стек технологий Web-сервисов*** (рисунок 5).

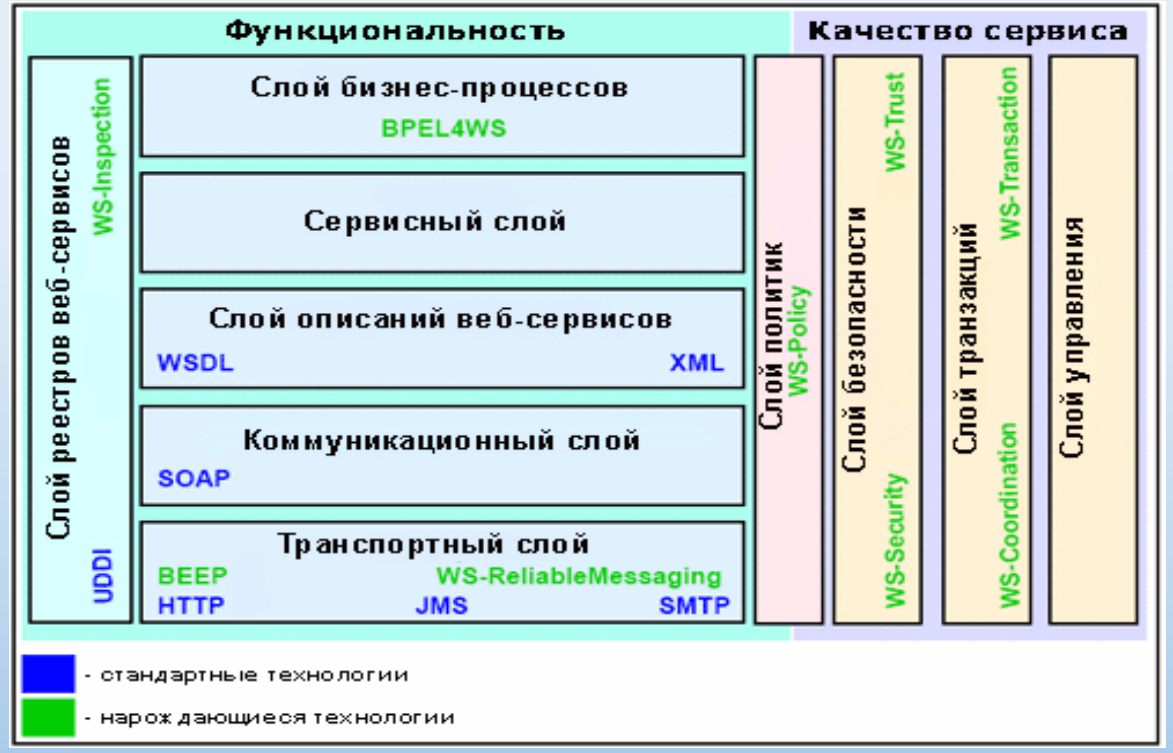

Рисунок 5 – Стек технологий Web-сервисов

Стек технологий Web-сервисов принципиально разбивается на следующие ***две составляющие***:

- технологии, обеспечивающие функциональность Web-сервисов (Functions);
- технологии, обеспечивающие качество сервиса Web-сервисов (Quality of service).

Эти составляющие в свою очередь образуются несколькими слоями (layers):

- технологии, обеспечивающие функциональность Web-сервисов:
- транспортный слой (transport layer);
- коммуникационный слой (service communication layer);
- слой описаний сервисов (service description layer);
- сервисный слой (service layer);
- слой бизнес-процессов (business process layer);
- слой реестров сервисов (service registry layer).
- технологии, обеспечивающие качество сервиса Web-сервисов:
- слой политик (policy layer);
- слой безопасности (security layer);
- слой транзакций (transaction layer);
- слой управления (management layer).

В целях понимания назначения слоев, дадим краткое описание каждого из них.

№ п/п	Наименование слоя	Назначение слоя	Технологии, реализующие слой
Функциональность (Functions)			
1	Транспортный слой (Transport layer)	Описывает средства обмена данными между Web-сервисами	Стандартные: HTTP, JMS (для Java-приложений), SMTP Нарождающиеся: WS-Reliable Messaging, BEEP
2	Коммуникационный слой (Service communication layer)	Описывает средства формализации механизмов использования транспортных протоколов Web-сервисами. Используя метафоры, можно отождествить транспортный протокол с дорогой между Web-сервисами, а механизмы его использования, определяемые коммуникационным слоем, с грузовыми машинами, перевозящими по ней от сервиса к сервису сообщения	Стандартные: SOAP Нарождающиеся: REST
3	Слой описаний сервисов (Service description layer)	Описывает средства формализации интерфейсов Web-сервисов с целью обеспечения их функционирования независимо от программно-аппаратной платформы реализации или языка программирования. Различают два вида описаний сервиса: - операционное (operational); - полное (complete)	Стандартные: XML, WSDL Нарождающиеся: ebXML
4	Сервисный слой (Service layer)	Описывает программное обеспечение, вызываемое с помощью WSDL-описаний интерфейсов Web-сервисов. В частности, это сами Web-сервисы	
5	Слой бизнес-процессов (Business process layer)	Описывает возможности организации Web-сервисов для реализации бизнес-процессов и потоков работ. При этом определяются правила, задающие последовательность взаимодействия Web-сервисов с целью удовлетворения бизнес-требованиям	Стандартные: в настоящее время нет Нарождающиеся: BPEL4WS
6	Слой реестров сервисов (Service registry layer)	Описывает возможности организации Web-сервисов в иерархические библиотеки, позволяющие публикацию, поиск и вызов Web-сервисов по их WSDL-описаниям интерфейсов	Стандартные: UDDI Нарождающиеся: WS-Inspection

№ п/п	Наименование слоя	Назначение слоя	Технологии, реализующие слой
Качество сервиса (Quality of service)			
7	Слой политик (Policy layer)	Описывает правила и условия, согласно которым Web-сервисы могут быть использованы. Поскольку данные правила и условия относятся как к функциональному аспекту Web-сервисов, так и к аспекту обеспечения качества сервиса, данный слой является общим для обоих аспектов	Стандартные: в настоящее время нет Нарождающиеся: WS-Policy, WS-PolicyAssertions и WS-PolicyAttachment
8	Слой безопасности (Security layer)	Описывает возможности обеспечения безопасности Web-сервисов и безопасности их функционирования (авторизация, аутентификация и разделение доступа)	Стандартные: WS-Security Нарождающиеся: WS-SecureConversation, WS-Federation, WS-Authorization, WS-Trust и WS-Privacy
9	Слой транзакций (Transaction layer)	Описывает свойство транзакционности распределенных систем на основе Web-сервисов для обеспечения надежности их функционирования	Стандартные: в настоящее время нет Нарождающиеся: WS-Transaction и WS-Coordination
10	Слой управления (Management layer)	Описывает возможности управления Web-сервисами и характеристиками их функционирования	

Представленный стек технологий Web-сервисов вводит иерархию во множество технологий Web-сервисов в соответствии с их функциональным назначением. При этом в таблице указаны лишь наиболее широко применяемые и устоявшиеся технологии.

Стандартными названы технологии, получившие официальный статус стандартов международных консорциумов по разработке IT-стандартов (W3C).

В действительности, спектр технологий, описывающих те или иные аспекты использования Web-сервисов либо сервисно-ориентированных архитектур, крайне широк, в частности, потому, что процесс разработки данных технологий является открытым – любая компания, некоммерческое объединение специалистов или даже один специалист может разработать и опубликовать спецификацию разработанной им технологии.

В настоящее время это даже стало серьезной проблемой рынка Web-сервисов – количество спецификаций стало столь велико, что при фактической нерегламентированности глобального процесса их разработки, ввода в действие и использования, появляется опасность ввергнуть индустрию в хаос и технологическую разобщенность. А технологическая разобщенность – как раз то, от чего хотели уйти прежде всего, разрабатывая Web-сервисы и закладывая в качестве их концептуальной и технологической основы открытые и наиболее широко применяемые IT-технологии.

7. Принципы взаимодействия Web-сервисов в рамках сервисно-ориентированной архитектуры

Итак, ***технологический фундамент Web-сервисов образуется следующими технологиями***:

- eXtensible Markup Language (XML);
- Simple Object Access Protocol (SOAP);
- Universal Description, Discovery and Integration (UDDI);
- Web Services Description Language (WSDL).

Данные технологии обеспечивают реализацию базовых свойств Web-сервиса, описанных в его определении. Они же лежат в основе взаимодействия Web-сервисов в рамках сервисно-ориентированной архитектуры (рисунок 6).

Рисунок 6 – Взаимодействие между компонентами сервисно-ориентированной архитектуры

Различают следующие три основных архитектурных компонента сервисно-ориентированной архитектуры:

- ***пользователь сервиса***: приложение, программный модуль либо сервис, осуществляющий поиск и вызов необходимого сервиса из реестра сервисов по описанию сервиса, а также использующий сервис, предоставляемый провайдером сервиса, в соответствии с интерфейсом сервиса;
- ***провайдер сервиса***: приложение, программный модуль либо сервис, осуществляющий реализацию сервиса в виде Web-сервиса, прием и исполнение запросов пользователей сервиса, а также публикацию сервиса в реестре сервисов;
- ***реестр сервисов***: библиотека сервисов, предоставляющая пользователям сервиса средства поиска и вызова необходимого сервиса и принимающая запросы провайдеров сервисов на публикацию сервисов.

Каждый компонент может играть либо лишь одну роль (быть, например, только пользователем сервиса) либо одновременно сразу несколько ролей (например, быть провайдером одних сервисов и пользователем других).

Заметим, что в данном описании компонентов сервисно-ориентированной архитектуры и взаимодействия между ними следует различать термины «сервис» и «Web-сервис». Под сервисом понимается бизнес-функция, под Web-сервисом – программная реализация бизнес-функции (сервиса).

В ходе взаимодействия друг с другом компоненты сервисно-ориентированной архитектуры выполняют следующие основные операции:

- ***публикация***: для того, чтобы сервис был доступным (вызываемым) пользователям сервиса, необходимо сделать его интерфейс известным им;

- ***поиск***: пользователь сервиса должен иметь возможность найти в реестре сервисов необходимый сервис, удовлетворяющий заданным критериям;

- ***связывание и вызов***: после получения описания сервиса, пользователь сервиса должен иметь возможность вызвать и использовать сервис в соответствии с описанием сервиса.

Рассматривая взаимодействие компонентов сервисно-ориентированной архитектуры, необходимо отметить наличие (и различие) следующих двух артефактов:

- ***описание сервиса***: определяет формат запроса и отклика при взаимодействии пользователя сервиса и провайдера сервиса, а также требуемое качество сервиса;

- ***сервис***: собственно сервис, который может быть вызван и использован пользователем сервиса в соответствии с опубликованным интерфейсом сервиса.

С целью выработки и популяризации стандартов, описывающих взаимодействие Web-сервисов в рамках COA, создано международное объединение около 150 ведущих компаний, называющееся WS-I (Web Services Interoperability Organization). Важным результатом работы данного объединения стало создание и утверждение в 2003 году так называемого WS-I Basic Profile 1.0 - пакета базовых спецификаций Web-сервисов, взаимоувязанных с целью обеспечения широких возможностей взаимодействия Web-сервисов.

В настоящее время в данный профиль входят следующие спецификации технологий:

SOAP 1.1;

WSDL 1.1;

UDDI 2.0;

XML 1.0;

XML Schema Part 1: Structures;

XML Schema Part 2: Datatypes;

RFC2246: The Transport Layer Security Protocol 1.0;

RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile;

RFC2616: HyperText Transfer Protocol 1.1;

RFC2818: HTTP over TLS;

RFC2965: HTTP State Management Mechanism;

Secure Sockets Layer Protocol 3.0.

Кроме того, в профиле описаны сценарии, демонстрирующие основные методы организации взаимодействия между Web-сервисами.

Для создания Web-сервисов могут использоваться такие инструменты, как Open Net (компании Sun), .NET (Microsoft) и Web Services (IBM). Существуют также инструменты с открытыми кодами (open source frameworks), например, проект Mono Project (www.go-mono.com), который предоставляет систему компиляции, выполнения кода и библиотек для работы одних и тех же Web-сервисов на всех платформах, включая Unix.

Выводы

В ходе лекции рассмотрены следующие вопросы:

- основы Web-сервисов;

- преимущества и недостатки Web-сервисов;

- взаимодействия с Web-сервисами;

- технология Web-сервисов и пример ее использования;

- определение сервисно-ориентированной архитектуры, ее преимущества и требования к ней;

- стек технологий Web-сервисов;

- принципы взаимодействия Web-сервисов в рамках сервисно-ориентированной архитектуры.

Задание на самостоятельную работу

1. Конспект лекций.

Вид и тема следующего занятия

Практическое занятие №16. Авторизация