УДК 681.3

РОЛЬ СЕМАНТИКИ В ИНТЕГРАЦИИ ПРИЛОЖЕНИЙ НА ОСНОВЕ ВЕБ-СЕРВИСОВ

Ф. Андон, В. Дерецкий

Институт программных систем НАН Украины,
03187, Киев-187, проспект Академика Глушкова, 40,
тел.: +38 044 526 5507; +38 044 526 4342; {andon, dva}@isofts.kiev.ua

В работе исследуются подходы к определению и использованию семантики для автоматической интеграции приложений на основе Веб-сервисов. Средства динамического поиска, выбора и автоматической интеграции программных компонентов для Веб-сервисов находятся в состоянии исследований и в полной мере не реализованы. Это частично объясняется отсутствием или недостаточным уровнем использования семантики в текущих стандартах сервисов. Для решения этой проблемы Семантическим сообществом разрабатываются направления «Семантический Веб» и «Семантические Веб-сервисы». Посредством кодирования требований и возможностей Веб-сервисов в машино-читаемой форме для однозначной интерпретации, семантика позволит осуществить автоматический поиск, композицию и интеграцию программных компонентов. Семантические Веб-сервисы будут использоваться в качестве средства для создания новой методологии программирования и интеграции приложений в распределенном динамическом окружении.

Approaches to definition and use of semantics for automatic integration of applications on the basis of Web services are investigated In articles . Means of dynamic service discovery, matchmaking and automatic composition of program components for Web services are in a status of researches and to the full are not realized. Level of use of semantics in current standards of services or are insufficient. For the decision of this problem the Semantic community develops directions «the Semantic Web» and «Semantic Web services». By means of coding of requirements and possibilities of Web services in the computer-readable form for unequivocal interpretation, semantics will allow to carry out automatic discovery, a composition and integration of program components. Semantic Web services will be used as means for creation of new methodology of programming and integration of applications in the distributed dynamic environment

Введение

Веб-сервис (Web service) представляет собой программную систему, которая определяется строкой URI и интерфейсами, определенными на языке XML. Описание этой программной системы может быть найдено другими программными системами, которые могут взаимодействовать с ней согласно описанию, посредством XML сообщений, передаваемых с помощью Интернет-протоколов. Усилия промышленности в последнее десятилетие, направленные на стандартизацию описания Веб-сервисов для поиска, развертывания и запуска, привели к появлению таких стандартов как, SOAP (2000), WSDL (2001) и UDDI (2002) соответственно. Эти стандарты спроектированы для представления информации об интерфейсах сервисов, их развертывании и вызове, но практически не представлены возможности сервисов. Отсутствие семантического представления ограничивает автоматическую интеграцию приложений, написанных в соответствии со стандартами Веб-сервисов.

Понятия интеграции и интероперабельности являются важными в разработке сервис-ориентированных приложений для реализации широкого круга задач в различных предметных областях: *е*-науки, *е*-правительства, *е*-торговли и других.

Интеграция приводит к «постоянному или временному расширению модулей приложения с целью объединения процессов и/или разделения информации». В результате обеспечивается взаимодействие приложения на различных уровнях, при котором независимые или гетерогенные информационные системы, их компоненты эффективно сотрудничают предопределенным и согласованным способом.

Интероперабельность (функциональная совместимость) – ключевая характеристика, которая позволяет учитывать данные и информацию для обмена и обработки в распределенном приложении. Интероперабельность является не только технической проблемой, которую необходимо решать при соединении компьютерных сетей и устройств, но также и фундаментальным требованием совместного и многократного использования знания при взаимодействии процессов и реорганизации управления для поддержки взаимодействия сервисов лучшим способом. Определены следующие три уровня интероперабельности:

технический –технические проблемы соединения компьютерных систем, определение открытых интерфейсов, форматов данных и протоколов, включая передачу данных;

семантический – обмен информацией в пределах и между приложениями, в локальном или корпоративном режиме, первоначально не разработанных для совместной работы;

организационный – моделирование бизнес-процессов, согласование информационной архитектуры с организационными целями, чтобы дать возможность процессам сотрудничать, взаимодействовать с их клиентами, используя информационные и коммуникационные технологии.

© Ф. Андон, В. Дерецкий, 2010

Если бы все поставщики сервисов во всех промышленных областях согласовали формат для представления своих сервисов, то можно было бы уменьшить или устранить требования к семантике. Однако, не все приложения и соответствующие сервисы могут быть стандартизированы. Если семантика сервиса не указана, то заказчик сервиса не сможет найти поставщика сервиса, используя только поверхностные сведения, полученные из описания интерфейса. Использование семантики для представления требований и возможностей Веб-сервисов обеспечивает автоматизацию поиска сервисов. Потребность в семантике Веб-сервисов привела к конвергенции понятий Веб-сервисов и семантического Веб – появление и развитие нового направления «семантические Веб-сервисы».

Семантические Веб-сервисы представляют собой Веб-сервисы, чьи «свойства, способности, интерфейсы и результаты кодируются в однозначной форме, поддающейся машиной обработке» [1]. Могут быть проанализированы, как «{семантические Веб} сервисы» с точки зрения сопоставления знания цели и интерфейсов или как «семантические {Веб-сервисы}», с точки зрения семантического описания сервисов для реализации поиска, композиции, запуска, мониторинга и др.

Роль семантики в жизненном цикле Веб-сервисов можно представить моделируя действия развертывания и динамические действия. В процессе моделирования действий поставщик сервисов может определять семантику, аннотируя соответствующие части Веб-сервиса, и используя понятия, полученные от семантической модели. Семантические модели обеспечивают согласование по использованию терминов и обеспечивают формальные и неформальные определения объектов, в результате чего уменьшается двусмысленность в семантике поставщика. Затем эти семантические Веб-сервисы могут быть изданы в реестре сервисов. Заказчик сервисов может описать требования сервиса, используя термины из семантической модели. Для того чтобы найти семантическое сходство между запросом сервиса и его описанием могут использоваться методы, основанные на рассуждениях.

В случаях, когда не существует согласованных сервисов, требуемая функциональность может быть скомпонована путем композиции доступных сервисов. В процессе композиции могут использоваться функциональные аннотации для того, чтобы приобщить функциональность доступных сервисов и создать полезную композицию.

В процессе развертывания сервиса семантика может использоваться для нахождения образцов сервисов, которые связаны с интерфейсами сервиса. В процессе запуска сервиса семантика может использоваться для преобразования данных. В процессе выполнения, в случае неудачи выполнения сервиса, семантика предназначена для нахождения подходящего сервиса для отработки возникшей ситуации. Цикл выполнения сервисного приложения на основе семантики показан на рис. 1. Однажды определив семантику, она может быть использована для автоматизации процессов поиска сервисов, посредничества, композиции, выполнения и контроля.

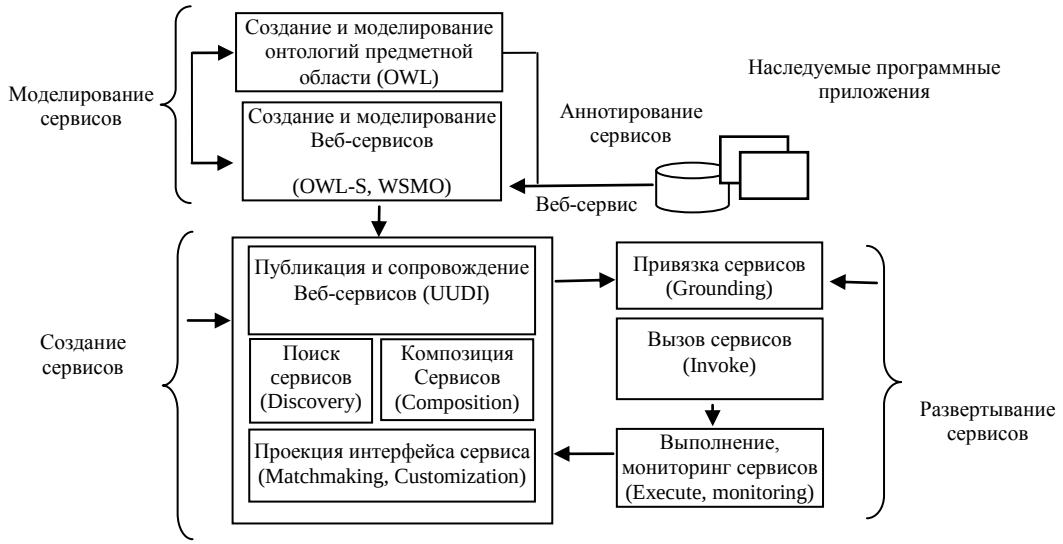

Рис. 1. Схема интеграции приложений на основе Веб-сервисов

Согласование Веб-сервисов (Matchmaking). Процесс согласования Веб-сервисов включает два шага и состоит в том, что запрос клиента (сервиса) может быть удовлетворен сервисами, которые определены в реестре. На первом шаге согласования выполняется синтаксическое сравнение параметров, описывающих триплет запроса (C, I, O) с параметрами триплета $(C, I, O)_s$ каждого доступного сервиса в реестре, и выполняется семантическое сравнение связанных понятий. Сервис и запрос согласованы, если его контекст и входы/выходы подобны с заданным порогом подобия, который может быть определен в пределах реестра. В конце этого шага получим множество сервисов, удовлетворяющих запросу клиента.

Второй шаг зависит от процесса согласования на первом шаге, т.е. первый шаг должен возвратить по крайней мере один сервис, что бы перейти ко второму шагу процесса согласования. Этот шаг включает две стадии: сравнение ограничений типов данных входа-выхода; сравнение значений ограничений.

Если у всех типов данных входа-выхода запроса и сервиса существует подобие, то конфликта нет, процесс согласования выполнен с положительной оценкой. Если параметры входа-выхода в запросе и в сервисе отличаются, то существует конфликт по типу данных. Конфликт по типу данных между входом-выходом запроса и входом-выходом сервиса означает, что сервис не может (априори) удовлетворить запросу. Однако это не значит, что данный сервис должен быть исключен из последующего рассмотрения окончательно, так как конфликт возможно будет разрешен в дальнейшем.

Определение 1 (аксиома покрытия). Пусть $A_1,...A_n$ – множество формальных понятий (концептов). Аксиома покрытия утверждает, что выражение $A := A_1 \vee ... \vee A_n$, является покрытием всех подпонятий A_1, $A_2...,A_n$ понятия A.

Аксиома покрытия, позволяет устранить различие между понятиями, которые способны привести к конфликтам типов. Аксиома может быть применима к каждому пространственному понятию, способному вызвать конфликт в онтологии предметной области. Например, понятие Цена (эквивалент понятия Деньги) можно представить множеством концептов аксиомы покрытия, так как цена продукта может быть определена в разных валютах. Таким образом, $Деньги := Евро \vee Иена \vee Доллар$.

Определение 2 (конфликт типов). Пусть C является множеством понятий, описанных в онтологии, x, y, z –три пространственных понятия, определенные в C, и заданы два правила ограничения:

$x = y$ (*запрос, ограничение типа*)

$x = z$ (*сервис, ограничение типа*)

$if \quad y \neq z \wedge (\exists c \in C \,|\, y \subseteq C \wedge z \subseteq C)$,

и если существует множество концептов аксиомы покрытия $c := x \vee ... \vee z$ тогда существует конфликт из-за различия между параметрами входа-выхода x в запросе и сервисе.

Если между запросом и сервисом выявлены конфликты, то для некоторых сервисов существует возможность их разрешения. Семантика разрешения конфликтов основана на преобразовании конфигурации Веб-сервиса, который доступен в реестре, в конфигурацию, удовлетворяющую запросам сервиса.

Процесс настройки Веб-сервисов (Customization). Процесс согласования между запросом и сервисом может выявить несколько конфликтов по типам параметров входа / выхода. Здесь используется композиция Веб-сервисов как средство для разрешения конфликтов. Другими словами, предлагается механизм, который позволяет преобразовывать сервисы, полученные из реестра, таким образом, чтобы они стали совместимыми с требованиями запросов к сервису.

Для получения контекста (ключевого слова) при разрешении конфликтов сервиса определяется ряд правил под названием «*Правила связывания контекста*». Таким образом, для каждой онтологии предметной области определяется множество правил, по одному правилу для каждого понятия, которое может вызвать конфликт типов.

Правило контекстного связывания представляет собой двойной предикат *ConflictResolution* (*Понятие, Контекст*). «Понятие» представляет переменные, определенные в онтологии предметной области и принадлежащие вершине аксиомы покрытия. «Контекст» представляет собой переменную, определяющую контекст (ключевые слова) сервиса для разрешения конфликтов.

Конфликт типов вызывается различием между двумя понятиями, включенных в категорию, которая определяется в первом параметре предиката *ConflictResolution*. Два противоречивых понятия восстанавливаются из концепта аннотации входа / выхода в запросе и сервисе.

Например, первое правило связывает понятие «Деньги» с ключевым словом (контекстом) сервиса разрешения конфликтов валюты: *ConversionMoney*. Второе – связывает понятие «Язык» с ключевым словом разрешения конфликтов представления: Трансляция.

ConflictResolution (*Деньги, ConversionMoney*)

ConflictResolution (*Язык, Трансляция*).

Контекст сервиса вызывается для разрешения конфликта типов параметров входа / выхода. Конфликт типов определяется посредством исследования правил контекстной ассоциации. Предикат «*ConflictResolution*», имеющий первый параметр – понятие, которое вызывает конфликт и второй параметр – переменная, указывающая название сервиса разрешения конфликта. Контекст разрешения конфликтов сервиса определен заменой переменной контекста ключевым словом, которое появляется в одном из контекстных правил ассоциации путем объединения терминов.

Преобразование значений входа / выхода сервиса осуществляется для того, чтобы сделать его сопоставимым с запросом. Преобразование значений входа / выхода осуществляется в фазе запроса сервиса. Если значения входа / выхода запроса и сервиса становятся сопоставимыми, то реализуется процедура согласования с проверкой непротиворечивости ограничений значений.

1. Мотивация применения семантики для Веб-сервисов

Вышерассмотренный подход к разрешению конфликтов учитывает синтаксические характеристики сервисов. Выявление сходства между сервисами – сложная задача, потому что терминология, используемая для описания Веб-сервисов не идентична той, которая используется в запросе. Кроме того, должны быть приняты во внимание структура и тип информации при описании сервиса. Явные семантики играют важную роль в урегулировании двусмысленностей терминологии. Далее рассмотрим необходимость определения семантики Веб-сервисов в контексте автоматического поиска сервисов, композиции, запуска на выполнение и наблюдения за выполнением сервисов (рис. 2).

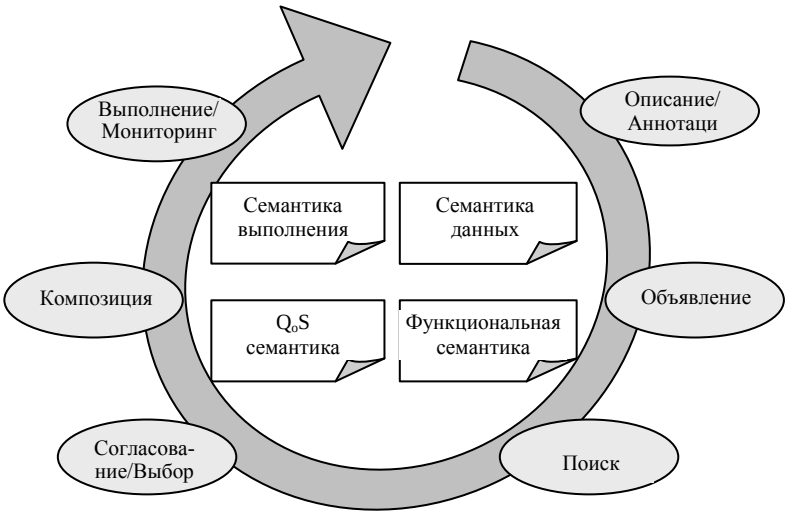

Рис. 2. Роль семантики в жизненном цикле Веб-сервисов

Поиск сервисов. Автоматический поиск сервисов использует набор характеристик, который соответствует заданным требованиям (функциональным и нефункциональным), получаемых из централизованного или распределенного репозитория сервисов. Согласование сервисов может быть синтаксическим (в зависимости от типа и структуры параметров) и/или семантическим (по лексическим характеристикам и онтологиям соответствия). В модели B2C согласование будет означать, что необходимо найти поставщиков сервисов, которые могут предложить сервисы, возможности которых соответствуют указанным требованиям, например, поиск источника Веб-сервисов, которые предлагают бронирование авиабилетов до пункта назначения и способа оплаты. С другой стороны, в модели B2B согласование состоит в том, чтобы найти поставщиков сервисов, которые ссылаются на свои сервисы без формального соглашения. Поэтому более подходящим является настройка программного обеспечения повторного использования активностей, поскольку значительная часть существующей функциональности может использоваться при создании новой функциональности. Поиск сервисов в этом контексте означает нахождение подходящей активности, которая может быть использована для полной или частичной реализации новой функциональности. Семантика может играть важную роль в реализации поиска в обеих группах. Рассмотрим несколько сценариев для исследования семантических проблем при согласовании сервисов. Для иллюстрации сценариев рассмотрим пример сервиса покупки товара, который предлагает покупателю следующие возможности:

поиск товара по каталогу;
получение информации о стоимости товара и его доставки;
добавление найденного товара в корзину покупки;
оплаты товара и его доставки после детального изучения платежей и состояния счета покупателем;
отгрузки товара покупателю после подтверждения оплаты;
доставки товара по адресу, указанному покупателем.

Синтаксические подобия и семантические различия сервисов. В отсутствие стандартизации, несколькими поставщиками сервисов могут быть предложены аналогичные Веб-сервисы с различными интерфейсами. Например, два сервиса, параметр которых принимает значение типа "string" и возвращает "float" могут выполнять совершенно разные функции. Можно использовать сервис *getStockQuote()*, который использует имя компании (string) в качестве входа и возвращает стоимость товара (float), другой сервис *itemAvailabilityCheck()*, который использует документ XML, имя которого задается в виде строки, состоящей из многих параметров, таких как *partNum*, назначение, количество и т. д., и вернуть количество (float), которое доступно на требуемую дату. Синтаксически, типы входных и выходных параметров в этом примере, согласованы, даже если имена сервисов точно не совпадают. Таким образом, синтаксическое согласование можно рассмотреть как основу согласования, которая не всегда приводит к положительному результату. Но очевидно, что эти два сервиса выполняют различные функции. Это может привести к ложным результатам при решении задачи соответствия при поиске.

Синтаксически различные и семантически идентичные сервисы. В отличие от предыдущих сценариев, синтаксически разнородные сервисы могут выполнять одни и те же функции. Например, два сервиса могут иметь различные синтаксические характеристики интерфейсов. Скажем, один сервис по имени *itemAvailabilityCheck*() может иметь входной параметр *ItemInfo* в виде строки, содержащей такую информацию, как *partNum*, *dateOfDelivery*, *requestedQuantity* и представить эту информацию в документе XML, который представлен в виде строки. Другой сервис по имени *verifyInventoryAvail*() может использовать параметры по отдельности, а не упаковывать их в документ XML. Структура параметров этих двух сервисов выглядят по-разному, даже если они выполняют одну и ту же функцию. Таким образом, синтаксическое соответствие не может определить эти два сервиса как согласованные и приводит к ложным результатам. Семантические согласования, основанные на лексических признаках или именах, могут находить сходства сервиса для определения частичного соответствия.

Синтаксические и семантические различия сервисов. Определение соответствия сервисов в предыдущем примере основано на том, что различные поставщики сервисов могут выбирать различные термины для обозначения одного и того же. Параметр одного поставщика сервисов называется *itemCode* и передается другому сервису *SKU* (Stock Keeping Unit). В случае отсутствия описания семантики предметной области о том, что *SKU* является *subClassOf itemCode*, тогда эти два термина могут рассматриваться как несвязанные между собой. Если семантика предметной области явно не указана, то такие способы определения согласованности сервисов не позволяют выявить условия соответствия сервисов.

Синтаксические и семантические подобия сервисов. В некоторых сценариях интерфейсы сервисов определяют как синтаксические и семантические параметры, но фактически представляют собой различные сущности. Чтобы проиллюстрировать это рассмотрим два Веб-сервиса: *getChipQuote*() и *checkChipPrice*(). Оба сервиса содержат параметр типа «string» и возвращают параметр «float». Эти сервисы выполняют совершенно разные функции. Сервис *getChipQuote*() ссылается на предметную область полупроводников, получает спецификацию чипа в виде документа XML (строка) в качестве входного параметра и возвращает цену (с плавающей точкой), а сервис *checkChipPrice*() ссылается на предметную область Веб-игр, получает входной параметр в виде строки и возвращает цену ставки (с плавающей точкой). Если синтаксическое соответствие этих двух сервисов достигается на основе типов параметров, то определение согласованности сервисов по структурному подобию будет ошибочным. Семантические сопоставления, основанные на лексическом условии без учета предметной области и контекста этих двух сервисов, не соответствуют истине. Эти два сервиса могут быть неправильно сопоставлены при семантическом и синтаксическом согласовании. В этом примере семантика определяется через предметную область контекста. Формально предметная область контекста может быть смоделирована посредством языков разметки, таких как OWL или WSMO [2–6].

Рассмотрено четыре сценария, в которых выявлено ложное согласование сервисов. Использование семантических Веб-сервисов обеспечивает автоматическое согласование сервисов, основанное на семантическом контексте и исключении ложных случаев. Механизмы семантического согласования Веб-сервисов смогут помочь решить задачи семантического различия терминов «*SKU»* и «*itemCode»*, рассмотренных в вышеприведенном сценарии.

Вызов сервисов на выполнение. Процедура поиска сервисов (Service Discovery) позволяет определить подходящий сервис на семантическом уровне. Для этого необходимо проверить соответствие общих параметров, указанных в запросе на выполнение сервиса. Однако, чтобы автоматически применить запуск выбранного сервиса, может потребоваться выявление фактического отображения интерфейса на более детальном уровне согласования. Например, запрос сервиса имеет входной параметр «*UPC code»*, который должен быть согласован на семантическом уровне с сервисом, использующим параметр «*SKU code»*. Однако такое согласование не может быть выполнено из-за различий в синтаксическом представлении этих двух кодов (например, UPS может содержать 14 цифр, а код SKU является 12-и значным). Может потребоваться сервис преобразования кодов, который конвертирует «*UPS код»* в «*SKU код»* до того, как выполнить вызов. Вызов Веб-сервиса предполагает создание отображения интерфейсов из запросов и выбранных сервисов. Такое отображение может потребовать выполнение более глубокого семантического анализа, чем тот, который был сделан в процессе поиска сервисов. Например, параметры *FirstName* и *LastName* одного сервиса необходимо представить в виде параметра *FullName* другого сервиса. В настоящее время пользователи вручную создают эти отображения в соответствии с документацией, а затем используют инструменты или пишут код, чтобы обеспечить согласование параметров для запуска сервиса. Использование семантических Веб-сервисов для реализации вызовов могут помочь снизить нагрузку на пользователей.

Композиция сервисов. Композиция Веб-сервисов используется для получения новой функциональности на основе связывания существующих Веб-сервисов. Композиция сервисов расширяет понятие Service Discovery (Поиск сервисов) в части автоматизации интеграции сервисов для удовлетворения требований, задаваемых высокоуровневым описанием задачи. Например, выполнение задачи "разместить заказ на поставку покупки, в котором поставщики должны поставить *N* единиц товара типа *X* на дату *Y*", может потребовать использования сервисов, которые должны выполнять цифровую подпись и шифрование данных, если поставщик требует информацию, для которой необходимо обеспечить безопасность. На рис. 3 показан пример такой композиции. Если для связывания сервисов необходимы преобразования параметров, то пользователь должен выбрать эти сервисы вручную и реализовать композицию. С другой стороны, задавая семантическую разметку сервисов, информация, необходимая для выбора и композиции сервисов, доступна через систему семантических описаний запросов и возможностей сервисов, которые обеспечивают автоматическую композицию.

Веб-сервисы в целом можно разделить на два класса – простые и сложные. Семантическое Веб-сообщество определяет простые или атомарные Веб-сервисы как «единая Веб-доступная компьютерная программа, датчик или устройство, которое запускается по запросу, поступающему в виде сообщения, выполняет свою задачу и возможно, создает один ответ на запрос. С атомарными сервисами не существует постоянного взаимодействия» [6, 7]. Простой Веб-сервис, как правило, является сервисом без состояний. В отличие от сложных или композитных сервисов, которые определяются как единый сервис состоящий из нескольких атомарных сервисов, и может использовать расширенное взаимодействие или «разговор между запросом и набором сервисов» [4, 5, 7, 8]. Сложный Веб-сервис включает в себя поток данных, определенный планом выполнения сервисов от одного шага к другому, поведение которого зависит от состояния информации. Например, процесс покупки может содержать порядок, в котором осуществляется проверка наличия товара, а затем размещение заказа на поставку. Такой набор связанных и структурированных процедур, которые реализуют требуемую задачу, можно определить как композитный Веб-процесс. Далее рассмотрим механизмы определения семантики как для простых, так и композитных Веб-сервисов.

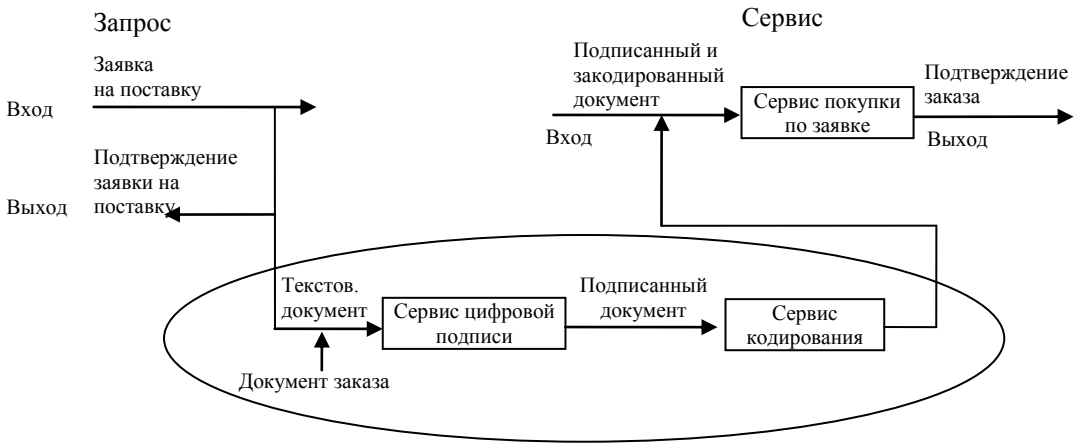

Рис. 3. Иллюстрация композиции семантических Веб-сервисов

Мониторинг выполнения сервисов. Семантика может играть важную роль в процессах контроля выполнения сервисов (monitoring). Когда сервисы становятся недоступными для взаимодействий из-за ошибки (зависание, завершения), тогда они могут быть обнаружены и связаны с сервисом отработки ситуации в месте возникновения ошибки. Это увеличивает надежность и живучесть системы, построенной на основе семантических Веб-сервисов. Предметная область играет важную роль в устранении двусмысленности в намерениях Веб-сервисов.

2. Средства для представления семантики Веб-сервисов

Языки формального представления онтологий и семантических Веб-сервисов используют дескриптивные логики [9], логики фреймов [10] и логического программирования [11]. Теория конечных автоматов (FSM) [12] и ее варианты, такие, как сети Петри [13] положены в основу моделирования распределенных взаимодействующих систем, которые моделируют Веб-процессы сложных Веб-сервисов. Такие теории, как алгебра процессов [14], исчисление ситуаций [15] и Пи-исчисление [16], представляют основу для языков типа PSL, которые представляют языки Семантических Веб-сервисов (FLOWs [17]).

Далее рассмотрим некоторые актуальные средства определения семантики Веб-сервисов. В частности, проекты и языки OWL-S , WSMO, WSDL-S и SWSA / SWSL [4, 6, 18–20].

OWL-S – одна из первых попыток определения формальной модели семантических Веб-сервисов. Язык OWL-S определяет высокоуровневые онтологии для описания свойств и возможностей Веб-сервисов. Он предназначен для того, чтобы пользователи и программные агенты обеспечивали автоматический поиск, развертывание и запуск, композицию и мониторинг Веб-сервисов в рамках заданных ограничений. Высокоуровневые конструкции определяются в виде профайла сервиса, который представляет интерфейсы сервисов, включающих входы, выходы, предусловия и постусловия, модель сервиса для представления деталей внутренней структуры, и Граундинг сервисов для обеспечения информации о том, как использовать сервис.

Принимая во внимание то, что профайл OWL-S представляет сервис как атомарный процесс, модель сервиса OWL-S определяет состояние сервиса. Описание функциональных свойств сервиса осуществляется через такие конструкции, как входы, выходы, предусловия и постусловия (иногда называемые IOPEs), OWL-S сервис использует модели потока работ, включая такие конструкции как «если-то-иначе», «повторения UNTIL» и другие. Модель Граундинга OWL-S определяет необходимые ссылки на промышленные стандарты.

Графическое представление онтологии рассматривается далее (рис. 4). Основным в этих отношениях является то, что *UPC Code* может быть передан вместо *EAN Cod* и, затем, *UPC Code* может быть использовано, как подмножество *EAN Code*. Если предметная область модели не задана, то интерфейс сервиса будет несогласованным из-за различия терминологий.

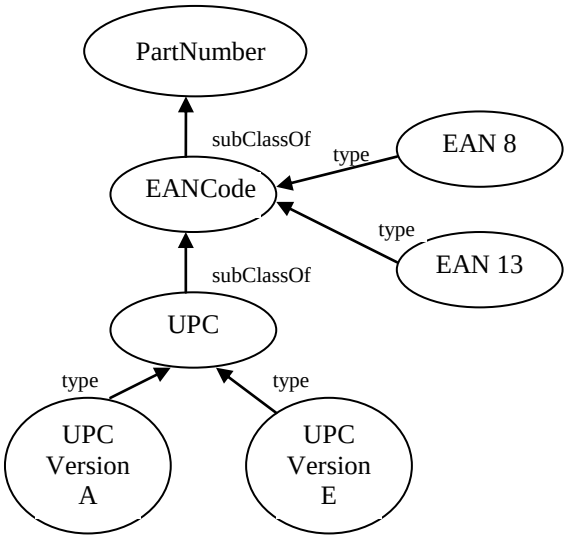

Рис. 4. Графическое представление онтологии

WSMO. Онтология моделирования Веб-сервиса (Web Service Modeling Ontology – WSMO [5]) включает четыре основных элемента: онтология, обеспечивающая терминологию, которая используется другими элементами WSMO; описание Веб-сервиса, которое определяет функциональные и поведенческие аспекты Веб-сервиса; цели, которые представляют намерения или запросы пользователя; посредники, которые направлены на автоматическую поддержку взаимодействия (интероперабельности) между различными элементами WSMO.

Подобно тому как OWL-S основан на языке онтологий OWL, WSMO основан на языке Web Service Modeling Language (WSML) [21]. OWL определяется несколькими версиями, а именно OWL-DL, OWL-Lite, OWL-Full, а WSML включает такие версии, как WSML-Core, WSML-DL, WSML-Rule, и WSML-Flight. Концептуальная разница между OWL и WSML состоит главным образом в использовании языков формальных логик. При этом как OWL полагается на дескриптивные логики (DL), WSML основывается на различных логических формализмах, а именно: дескриптивная логика, логики первого порядка и логического программирования (F-logic).

Общим для WSMO и OWL-S является то, что онтологии имеют важное назначение для поддержки автоматизации процесса поиска, композиции и взаимодействия Веб-сервисов. Но, несмотря на объединяющее представление, OWL-S и WSMO сильно отличаются в деталях и подходах для достижения результатов. Принимая во внимание то, что OWL-S определяет набор онтологий, которые поддерживают рассуждения о Веб-сервисах, WSMO определяет концептуальные границы, в пределах которых эти онтологии должны быть созданы. Еще одно различие между OWL-S и WSMO состоит в том, что хотя OWL-S не различает типы Веб-сервисов, WSMO делает акцент на спецификацию посредников, которые обеспечивают решение проблем взаимодействия (оркестровка, хореография) между Веб-сервисами [6].

WSDL-S [18] обеспечивает усиление семантических аннотаций для WSDL спецификаций. В WSDL-S пользователи могут добавлять семантические аннотации для WSDL документов с помощью расширения структуры спецификации. Семантические аннотации могут быть ссылками на понятия, определенные во внешней онтологии. Пользователи могут использовать OWL, WSMO или любой другой язык моделирования. Рис. 5 представляет пример назначения WSDL-S. Пример показывает как реализованы ассоциации между понятиями WSDL и их семантическими аннотациями, которые поддерживаются URI-ссылками, когда модель предметной области является внешней по отношению к Веб-сервису.

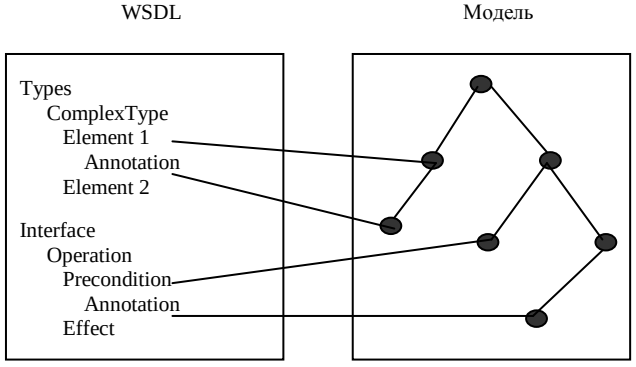

Рис. 5. WSDL-S: представления и ассоциации семантики с элементами WSDL

3. Сравнительные характеристики средств представления и моделирования семантических Веб-процессов

В табл. 1 обобщены языки представления семантических Веб-сервисов, представленных W3C.

Таблица 1

Предложения W3C	Комментарии	Семантический язык	Формализм	Отношения с WSDL
OWL-S	OWL онтологии высокого уровня для Веб-сервисов.	OWL	Дескриптивная логика	Определяет подключение к WSDL посредством модели Граундинга, но существует дублирование в определении входов, выходов и операций.
WSDL-S	Использование расширения элементов в WSDL, чтобы аннотировать элементы с условиями внешних онтологий.	Дополнение к языкам онтологий (может работать с OWL, WSMO, UML, XML или любой другой язык моделирования)	Пользователи по своему усмотрению выбирают формальный аппарат	Указывает аннотации в WSDL, как расширения элементов.
WSMO	Онтологии моделирования Веб-сервисов выражается с помощью WSML.	WSML	Дескриптивные логики, логики первого порядка и логики программирования (F-Logic)	Определяет связь с WSDL посредством модели Граундинга , но существует дублирование входов, выходов и операций

Средства OWL-S, WSMO и WSDL-S были представлены W3C как альтернативные предложения для определения схем семантики Веб-сервисов. Хотя все три предложения используют аналогичные концептуальные основы, различия состоят в возможностях. Предложения OWL-S представляет собой основу для простых (атомарных), а также для комплексных сервисов. WSMO определяет схему хореографии и оркестровки Веб-сервисов. С другой стороны, WSDL-S предлагает подход к добавлению семантики Веб-сервисов, исключая при этом спецификацию модели. Кроме того, с OWL-S и WSDL сервисами связаны языки OWL и WSMO, онтология WSDL-S может работать с любым языком онтологий. С другой стороны, WSDL-S реализует добавление семантических аннотаций к простым Веб-сервисам.

Работы по созданию архитектур для семантических Веб-сервисов были начаты в рамках инициатив семантических Веб-сервисов (SWSI). К ним относятся Semantic Web Services Architecture (SWSA) [19] и Web Services Language Requirements (SWSL) [20]. SWSL выдвигает следующие функциональные требования для любого языка Семантических Веб-сервисов: рекламирование и согласование, переговоров и заключения контрактов, моделирования процессов. Также определены формализмы, которые могут адекватно поддерживать эти функциональные требования. Для создания эталонной среды реализации WSMO разработана архитектура Web Service Modeling Execution Environment (WSMX) [22]. Архитектура реализует интеграцию бизнес-приложений, в которых обеспечиваются возможности семантических расширений комплексных сервисов для различных бизнес-приложений.

Сравнительный анализ языков для представления семантики Веб-процессов. Далее рассмотрим средства представления семантики для композитных Веб-сервисов.

Хотя язык OWL обеспечивает описания семантических моделей Веб-сервисов, он имеет определенные ограничения, которые становятся все более очевидными при моделировании сложных сервисов и Веб-процессов. OWL имеет набор конструкций для представления классов, но он не предусматривает использование переменных, особенно при определении связанных классов в онтологии. Такое ограничение касается входов и выходов композитного сервиса. Кроме того, нет конструктора композиции, так как невозможно получить отношения между композитными свойствами. Стандартным примером здесь является взаимосвязи между композицией «родительских» и «дочерних» свойств с «непрямыми родственниками». Другие языки моделирования сложных процессов такие, как конечные автоматы, Пи-исчисления и сети-Петри умеют выражать аспекты процесса, но не выражают такие характеристики, как условные результаты (постусловия) и нефункциональные ограничения [23]. Чтобы преодолеть эти ограничения разработан новый язык, который называется FLOWS, представляющий онтологии для Веб-сервисов в логике первого порядка. FLOWS [17] основан на стандарте ISO 18629 и добавляет расширения к PSL для выражения управления и упорядочения.

В табл. 2 приведены семантические языки для представления композитных Веб-процессов.

Таблица 2.

Предложения	Комментарии	Язык онтологий	Формализм
OWL-S. Модель процессов	OWL-онтологии для представления Веб-процессов	OWL	Дескриптивные логики
FLOWs	Сочетание правил логики первого порядка для представления Веб-сервисов. Основан на процессах Language Specification Process (PSL). FLOWs расширяет OWL-S моделями процессов.	SWSO (FLOWs + ROWs)	Логика первого порядка
WSMO оркестровка	Модели оркестровки в WSMO ссылаются на комплексные сервисы с состояниями.	WSML	Дескриптивные логики, логики первого порядка и логики программирования (F-Logic)

Значительный объем исследований было выполнено в области поиска Веб-сервисов и композиции.

Поиск сервисов. Подходы к поиску сервисов, основанные на методах искусственного интеллекта, согласования схем баз данных, инженерии программных систем и других, были направлены на получение синтаксических и семантических соответствий сервисов.

В работе [24] представляются модели согласования на основе рассуждений с использованием DL. Вводится степень соответствия с использованием точных понятий.

Проблема автоматического поиска сервисов исследовалась в [7, 25]. Схема соответствия анализирует структурное сходство и возможность использования проблемно-ориентированных онтологий для нахождения соответствующих Веб-сервисов. Проблемно-ориентированные онтологические сходства выводятся путем логического вывода. Семантические аннотации объединяются с описаниями Веб-сервисов.

Композиция сервисов. Исследования по композиции сервисов были сосредоточены на двух основных направлениях. Первое направление использует подходы планирования или логические алгоритмы для получения композиции, при этом второе направление использует методы поиска информации и композиции соответствующих сервисов, решая задачи семантической однозначности сервисов.

Общий обзор подходов к планированию композиции Веб-сервисов представлен в [25–27], в котором была сделана первая попытка использования планирования композиции Веб-сервисов. Метод композиции Веб-сервисов применяет механизмы логического вывода на предопределенных шаблонах планов.

Работы второго направления используют модели двудольных графов и решают проблему получения максимального соответствия при решении семантической однозначности, используя независимые онтологии предметной области и подходы по анализу естественно-языковых текстов. Контекстные условия представляет собой новую технику объединения предметных областей (например, тезауруса) с областью, представленной в виде проблемной онтологии и подходов планирования для получения композиции. Методы анализа текста, такие как Tokenization, обеспечивают семантическое согласование терминов, используемых в описании Веб-сервисов.

В табл. 3 приведены некоторые характеристики семантических Веб-сервисов, по которым исследования не проводились или проводились частично.

Таблица 3

Рубрики	Виды и подходы
Типы семантических Веб-сервисов	Простые, Комплексные
Свойства семантических Веб-сервисов	Функциональная, Нефункциональные
Источники семантической информации	Модели предметной области (онтология), предметно-независимые модели (словари) и их комбинации
Методы поиска сервисов	Информационный поиск (анализ текста, векторное пространство, вероятностные модели), AI методы (машинного обучения, семантических рассуждений), Математические методы моделирования (линейное программирование) и их комбинации
Методы композиции сервисов	Информационный поиск (анализ текста, векторное пространство, вероятностные модели), AI методы (машинного обучения, семантическое рассуждение, планирование), Математические методы моделирования (линейное программирование) и их комбинации
Подходы рекламирования сервисов	Централизованные (например, UDDI), децентрализованные / Peer-To-Peer (P2P)
Виды реестров сервисов	Распределенные, централизованные

4. Потенциальные выгоды, получаемые от применения семантических Веб-сервисов

Далее рассмотрим некоторые из потенциальных преимуществ применения Семантических Веб-сервисов [28].

Уменьшение затрат и времени на разработку приложений. Веб-сервисы обеспечивают основу для облегчения интеграции систем путем использования подходов, основанных на стандартах. Использование Веб-сервисов положено в основу интеграционных решений во многих отраслях. Но потребности в ручной работе не позволяют усилить эффективность разработки. Семантические Веб-сервисы открывают перспективы автоматизации задач интеграции. Автоматизация семантических Веб-сервисов может сэкономить время на разработку и сократить затраты на внедрение. Технологически это кажется правдоподобным. Однако, эти утверждения еще предстоит проверить на строгих исследованиях с применением новых методологий в реальных сценариях и архитектурах.

Развитие гибких и надежных систем. Семантика может помочь не только в автоматизации разработки приложений, но также использоваться при выполнении приложений. Среда выполнения в бизнес-контексте является динамической: сервисы, которые были доступны минуту назад, в любой момент могут стать недоступными. Во время выполнения должны быть предусмотрены процессы, реагирующие на изменения, не нарушая существующей среды выполнения. Могут быть использованы методы обнаружения сервисов и отображения интерфейсов во время динамического выполнения, для того чтобы найти новые или взаимозаменяемые сервисы. Это делает программное обеспечение систем более надежным и гибким.

Формальные модели способствуют лучшему использованию и поддержке программных систем. Семантический подход к Веб-сервисам инициирует дисциплины для развития формальных подходов в процессе интеграции приложений. Формализация предоставляет дополнительные преимущества поддержки сложных программных систем в процессе их изменения.

Заключение

Веб-сервисы становятся важным технологическим компонентом в области методологии программирования и интеграции приложений. Крупные предприятия развертывают сотни Веб-сервисов. В связи с этим возникает необходимость в средствах обнаружения и композиции сервисов и приложений. Семантика обеспечивает автоматизацию динамического поиска, композиции и исполнения Веб-сервисов. Акценты смещаются от синтаксической к семантической интероперабельности на основе подходов принятия решений. Основной мотивацией развития семантических методологий являются:

–постоянно растущие требования рынка IT-проектов к высокой скорости разработки решений, гибкости, простоте в обслуживании, что определяет необходимость новых, основанных на стандартах архитектур и методологий;

–требования IT-проектов к высокой степени интеграции данных и приложений, управление бизнес-процессами корпораций, эффективность использования ресурсов;

–растущие расходы на проверку и тестирование приложений требуют увеличения производительности разработки с использованием средств автоматизации.

Для поддержки интеграции разрабатываются языки моделирования, которые эволюционируют от синтаксической ориентации и значительной доли человеческого участия к семантической и машинно-интерпретируемой ориентации. С появлением языков семантического моделирования, таких как Resource Description Framework (RDF) и Web Ontology Language (OWL), появилась необходимая основа для интеграции приложений на основе семантики. Семантические Веб-сервисы, использующие языки моделирования OWL и WSMO, направлены на реализацию возможности автоматизации процессов разработки и интеграции приложений.

1. *McIlraith S., Son T.C., Zeng H.* Semantic Web services // IEEE Intelligent Systems. Special Issue on the Semantic Web. – 2001. – Vol. 16. – N. 2. – P. 46–53.
2. *McIlraith S., Son T.C., & Zeng, H.* Mobilizing the Semantic Web with DAML–enabled Web services // Paper presented at the Semantic Web Workshop. – 2001. Hong Kong. China.
3. *OWL:* Web ontology language // OWL Technical Committee. – 2004. A W3C specification. Retrieved. – October 25 2006. – from http://www.w3.org/2004/OWL/
4. OWL–S: Semantic markup for Web services // OWL–S Technical Committee. – 2004. W3C member submission. Retrieved. – October 25 2006. – from http://www.w3.org/Submission/2004/SUBM–OWL–S–20041122/
5. OWLS' relationship to selected other technologies // OWL–S Technical Committee. – October 25 2006 – from http://www.w3.org/Submission/2004/SUBM–OWL–S–related–20041122/#wsmo
6. Web service modeling ontology (WSMO) // WSMO Technical Committee – 2005. A W3C Member Submission. Retrieved – October 25 2006. – from http://www.w3.org/Submission/WSMO/
7. *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», 24–26 грудня 2007. − К., 2008. − С. 40−53.
8. *Дерецкий В.* Подход к композиции Веб-сервисов на основе спецификации функциональной семантики // Проблеми програмування. – 2009. – № 2. С. 30–39.

9. *Baader F., Calvaneese D., McGuinness D., Nardi, D., Patel-Schneider P*. The description logics handbook. Theory, implementation and tools. – 2003. Cambridge University Press.
10. *Kifer M., Lausen G., Wu J.* Logical foundations of object–oriented and frame–based languages // J. of the ACM. – 1995. – N 42. – P. 741–843.
11. *Baral C., Gelfond M.* Logic programming and knowledge representation // J. of Logic Programming. – 1994. – Vol. 19. – N. 20. – P.73–148.
12. *Gill A. Introduction to the theory of finite-state machines*. // McGraw–Hill. – 1962.
13. *Reisig W.* Petri nets, an introduction. // In W. Brauer, G. Rozenberg, A. Salomaa (Eds.). Monographs on theoretical computer science. – 1985. Berlin. Springer Verlag.
14. *Fokkink W. J.* Introduction to process algebra // Texts in Theoretical Computer Science an EATCS Series). – 2000. Springer.
15. *Levesque H., Pirri F., Reiter R.* Foundations for the situation calculus. // Electronic Transactions on Artificial Intelligence. – 1998. – Vol. 2. – N 3–4. – P. 159–178.
16. *Sangiorgi D., Walker D.* The Pi-calculus: A theory of mobile processes. // Cambridge University Press. – 2001.
17. *Gruninger M., Hull R., McIlraith S.* A first order logic ontology for Web services (FLOWs). – 2005. // Paper presented at the W3C Workshop on Frameworks for Semantics in Web Services. Innsbruck. Austria. Retrieved. – October 25 2006. from http://www.w3.org/2005/04/FSWS/Submissions/59/Flows.pdf
18. *WSDL–S* Web services semantics – WSDL–S // WSDL–S Technical Committee. – 2005. W3C Member Submission. Retrieved, October 25 2006. from http://www.w3.org/Submission/WSDL–S/
19. Semantic Web services architecture (SWSA) // SWSA Technical Committee. – 2005. Retrieved. – October 25 2006. from http://www.daml.org/services/swsa/note/
20. *Semantic* Web service language (SWSL) // SWSL Technical Committee. A W3C Member Submission. – 2005. Retrieved, October 25 2006. from http://www.w3.org/Submission/SWSF–SWSL/
21. *Bruijn J., Fensel D., Keller U., Kifer M., Lausen H., Krummenacher R., Polleres A., Predoiu L.* Web ser vice modeling language (WSML) // A W3C submission. – 2005. Retrieved, October 25 2006. from http://www.w3.org/Submission/WSML/
22. *Web service* execution environment (WSMX) // WSMX Technical Committee. – 2004. Retrieved, October 25 2006. from http://www.wsmx.org/
23. *Gruninger M., Hull R., McIlraith S.* A first order logic ontology for Semantic Web services // Paper presented at the Frameworks for Semantics in Web Services W3C. – 2005. Workshop. Innsbruck. Austria.
24. *Li L., Horrocks I.* A software framework for matchmaking based on Semantic Web terminology // In Proceedings of the WWW Conf. –2003. Retrieved, October 25 2006. from http://www.cs.man.ac.uk/~horrocks/Publications/download/2003/p815-li.pdf
25. *Madhavan J., Bernstein P., Rahm E.* Generic schema matching with cupid // In Proceedings of the 27th VLDB Conf. – 2001. Italy. Rome. Retrieved, October 25 2006.
26. *Ponnekanti S., Fox A.* SWORD: A developer toolkit for Web service composition. // In Proceedings of the 11th International World Wide Web Conf. – 2002.
27. *Peer J.* Web service composition as a planning problem // A survey Technical report. University of St. Gallen. – March 2005.
28. *Дерецкий В.* Разработка приложений в сервис-ориентированной архитектуре семантического Веб // Проблеми програмування. – 2010. – № 1. С. 66–78.