Содержание:

1. Введение
2. Теоретические основы семантической паутины
3. Персональные информационные менеджеры
4. Идея персонального информационного менеджера на основе технологий семантической паутины
5. Архитектура и реализации SemanticDesktop
6. Заключение

Альтернатива – Микроформаты для HTML

Web data that can be processed by machine.

Информация, которая получена из сети и которую можно машинно обработать. Впервые термит “Семантическая паутина” произнес Тимоти Джон Бернерс-Ли создатель World Wide Web и Глава коммитета всемирной паутины. Обычно страницы в интернете сделаны так, чтобы их было легко читать людям а не машинам. В сементической же сети информация может быль легко семантически (с сохранением смысла) интерпретирована машинами. The semantic web is a vision of information that can be readily interpreted by machines, so machines can perform more of the tedious work involved in finding, combining, and acting upon information on the web.

Взаимодействие различных онтологий Semantic Web tools allow them to link their schemes and translate their terms, gradually expanding the number of people and communities whose Web software can understand one another automatically.

Рассказать о существующем поиске по ключевым словам и о его недостатках.

Perhaps the most visible examples, though limited in scope, are the tagging systems that have flourished on the Web. These systems include del.icio.us, Digg and the DOI system used by publishers, as well as the sets of custom tags available on social sites such as MySpace and Flickr. In these schemes, people select common terms to describe information they find or post on certain Web sites. Those efforts, in turn, enable Web programs and browsers to find and crudely understand the tagged information—such as finding all Flickr photographs of sunrises and sunsets taken along the coast of the Pacific Ocean. Yet the tags within one system do not work on the other, even when the same term, such as “expensive,” is used. As a result, these systems cannot scale up to analyze all the information on the Web.

# **Использование технологий Semantic Web для интеграции информационных систем**

Технологии семантической паутины сейчас привлекают к себе внимание, потому что на их основе уже создаются новые интересные инструменты. В качестве примера можно привести социальный поиск Graph Search от Facebook.

Однако сфера применения семантических технологий не ограничивается социальными сетями и поисковыми сервисами. Идея применить эти технологии для организации обмена данными между информационными системами достаточно очевидна. Если одна система передает другой не только сами данные, но и информацию об их сущности в предметной области (смысле, семантике), это позволяет лучше абстрагировать обменивающиеся системы друг от друга, чем при использовании выгрузок в XML, JSON или веб-сервисов SOA.



Рисунок 1. Преобразование данных в RDF триплеты для обмена данными между информационными системами

Сегодня существует несколько реализаций такого подхода. Большинство из них, конечно, сделано зарубежными компаниями, но есть и российские разработки. В данном подразделе я расскажу об архитектуре одной таких систем, которую российские разработчики реализовали на практике.

При создании системы было задано, что нескольким информационным системам необходимо очень быстро (с интервалом всего в несколько секунд) сообщать друг другу об изменениях, происходящих в их хранилищах данных. Поэтому архитектура сильно напоминает шину обмена сообщениями (Message Queue), основной особенностью которой является то, что содержание сообщений выражено в синтаксисе RDF, то есть представляет собой RDF-триплеты, сохраненные в документе типа RDF/XML. Со стороны каждой из интегрируемых систем работает клиентский модуль обмена, который интерпретирует получаемые сообщения, и, если необходимо, вносит соответствующие изменения в хранилище данных своей системы. Клиентский модуль является автономным, то есть, если в хранилище произошли изменения, о которых нужно сообщить другим информационным системам – он кодирует сведения об этих изменениях в RDF, и отправляет в виде сообщения по корпоративной шине.

Роль маршрутизатора в шине выполняет центральный сервер-посредник, который обладает информацией о правах доступа информационных систем к разным видам данных, гарантирует доставку, контролирует целостность данных (и, при необходимости, старается ее восстановить), а также выполняет ряд других полезных функций. На рисунке ниже показана схема программных компонентов, участвующих в обмене. В качестве примера двух обменивающихся систем взяты, с одной стороны, некое веб-приложение на PHP/MySQL, и конфигурация 1С — с другой.

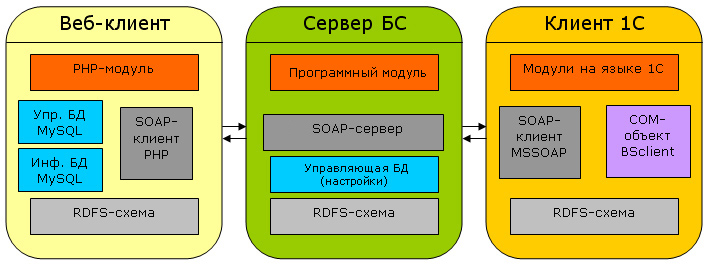


Рисунок 2. Возможная архитектура семантической корпоративной системой интеграции

Разумеется, на практике обменивающихся систем может больше, чем две.  
Преимущества этого подхода по сравнению с выгрузкой в XML или веб-сервисом SOAP достаточно очевидны, особенно если исходить из того, что необходимо связать между собой три, четыре или более систем. Они могут оперировать одними и теми же типами объектов (например, во всех информационных системах любой компании почти наверняка присутствует понятие «клиент»), но обладать разными наборами данных о них, и использовать их в разных контекстах. Также необходимо уточнить, что здесь хорошо проявляется один из принципов семантической паутины, а именно – глобальные типы. Существует множество информационных систем различного уровня, которые оперируют одинаковыми, по сути, сущностями (клиент, заказ, прибыль, работник), но в каждой системе эти сущности реализованы по-разному. Из-за эти ИС для обмена информацией друг с другом придется каждый раз преобразовывать данные из «неродного» формата. Однако если хранить информацию изначально в семантическом графе RDF, то сущность «Клиент» и в ERP системе маленького магазинчика и в CRM системе транснациональной корпорации будет представлена в одинаковом виде и обладать одним и тем же набором свойств. При традиционном подходе выгрузки в XML почти всегда будут избыточными, жестко связанными со структурой данных в базе данных системе-источнике, и потребуют написания программного кода для экспорта и импорта. Если, например, в CRM-системе хранятся таблицы с клиентами и сделками с ними, XML-выгрузка будет выглядеть примерно так:

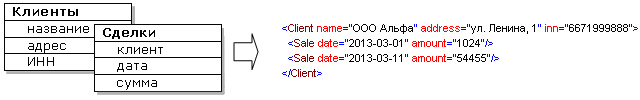


Рисунок 3. Представление информации в виде XML

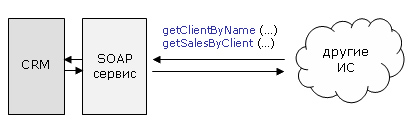
Понятно, что одной системе-потребителю этой информации сведения о сделках потребуются с детализацией по товарам, другой — без нее, зато со сведениями о грузополучателе и сроке доставки, и т.д. Использование выгрузок становится крайне проблематичным, если какие-нибудь данные могут измениться «задним числом». А уж если какая-нибудь из систем-получателей имеет право вносить изменения в эти данные (например, из системы отдела логистики могут поступать сведения о фактической отгрузке товара) — продолжать использовать выгрузки можно только из чистого упорства.  
Более прогрессивный вариант — использовать SOAP веб-сервис. Тогда схема взаимодействия системы-источника с потребителями информации будет выглядеть так:  


Рисунок 4. Использование веб-сервисов для интеграции информационных систем

Это будет удобно до тех пор, пока число сервисов (растущее пропорционально числу типов информационных объектов, которыми нужно обмениваться) не перевалит через несколько десятков. Тогда проблема мониторинга работоспособности сервисов, их документирования и поддержки начнет становиться по-настоящему критичной. Кроме того, это не решает проблему обратной связи систем: в упомянутом выше примере, когда система отдела логистики не только забирает информацию из CRM, но и может сама помещать в нее какие-либо данные, веб-сервисы придется реализовывать со стороны обеих систем, что совсем неудобно. Также программистам подобных систем придется самим заботиться о безопасности сервисов, поддержании целостности данных, обработке ошибок и т.п.

В более крупных инфраструктурах применяют MDM-системы (что требует совсем другого порядка вложений и усилий), а также шины обмена сообщениями. Шины хороши тем, что позволяют объединять большое количество информационных систем без возрастания сложности обмена в зависимости от числа систем. Фактически здесь реализуется шина, только в отличие от «классической» шины по ней будут передаваться сообщения, выглядящие примерно так:  
Информация, кодированная в виде триплетов RDF

Как уже говорилось, каждая система-отправитель перед отправкой преобразует в такой формат сведения, которыми хочет поделиться, а каждая система-получатель интерпретирует их в соответствии со своими правилами.  
Преимущества такого подхода в сравнении с классической шиной обмена сообщениями состоят в том, что информационным системам нет необходимости создавать «протокол обмена», определяющий смысловую нагрузку сообщений. Его роль выполняет онтология, известная всем обменивающимся системам и серверу (сервер обладает наиболее полной версией онтологии, клиентские системы могут использовать ее подмножества). В настройках каждого клиентского модуля производится сопоставление (mapping) элементов онтологии структурным элементам локального хранилища данных (проще говоря, таблицам и полям базы). Разумеется, можно создавать пользовательские программные обработчики для сущностей и связей, которые не имеют однозначного отображения на структуру базы данных, но в практических внедрениях объем требуемого программного кода измеряется десятками строк. Также, в отличие от классических шин обмена сообщениями, в нашем случае сервер выполняет ряд высокоуровневых полезных функций, вроде уже упомянутого восстановления целостности данных, выявления дублей объектов, разрешения конфликтов. Кстати, заодно сервер может формировать SPARQL базу данных, в которой будут аккумулированы сведения из всех обменивающихся систем, представленные в виде графа. Это обещает обеспечить такие возможности аналитики, какие не сможет предоставить ни одна из интегрируемых систем в отдельности.

Конечно же, использование описанного здесь подхода позволяет решить все перечисленные выше проблемы интеграции: обеспечить независимость процедур обмена со стороны различных систем от структуры баз друг друга, отказоустойчивость, безопасность, гибкость настройки, возможность легко использовать одну и ту же информацию в разных контекстах. Разумеется, описанный подход является далеко не единственным вариантом применения семантических технологий для интеграции данных.

# **Сравнительный анализ семантических моделей для корпоративных приложений**

Идея применения семантических моделей в корпоративных информационных системах существует давно, но устойчивая практика такого их использования еще не сформировалась. Семантические модели можно применять для интеграции данных, аналитики, управления знаниями; однако, пока нет общепринятого мнения о том, как подходить к оценке их полезности, по каким методикам должны строиться такие модели.

В этом разделе мы на практическом примере сравним аналитический потенциал моделей, построенных по правилам интеграционного стандарта ISO 15926, который предписывает использование OWL и SPARQL для выражения моделей и работы с ними, и «обычных» семантических моделей, построенных без использования этого стандарта. Решение этого вопроса позволит выбрать диапазон задач, для решения которого целесообразно применять такие высокоуровневые парадигмы семантического моделирования, как ISO 15926.

# **Постановка проблемы**

Необходимо кратко осветить историю вопроса, и суть взаимоотношений между ISO 15926 и «обычной» семантикой. ISO 15926 – стандарт обмена информацией, предназначенный для использования в промышленности (прежде всего, нефтегазовой). Исторически, акцент при разработке стандарта делался на обмен данными между различными организациями, т.е. между различными информационными инфраструктурами. Основные его особенности – специфический подход к классификации объектов и их отношений, учет временнóй составляющей объектов (4D моделирование), возможность моделирования жизненного цикла систем (а не просто текущего состояния той или иной системы). Стандарт содержит онтологическое ядро, и подразумевает использование общих библиотек справочных данных для создания прикладных информационных моделей. Все это обеспечивает как его преимущества (возможность создания высококачественных и релевантных моделей жизненного цикла систем, отличный потенциал использования для передачи информации между различными организациями при помощи общего «онтологического словаря»), так и недостатки (возрастающую сложность получающихся моделей, высокий «порог входа» по уровню знаний, необходимых для овладения стандартом и его использования).  
Разработка стандарта была начата еще в 1990-е годы. С появлением в середине 2000-х технологий Semantic Web, они были утверждены в качестве технологической основы для выражения данных в соответствии с ISO 15926. Таким образом, основные концепции стандарта были заложены до возникновения Semantic Web, но только появление этих технологий предоставило необходимый технологический базис для создания способа выражения данных в соответствии со стандартом, обладающего потенциалом действительно широкого распространения. Некоторая идеологическая близость, но не идентичность этих технологий заложила основу того «противоречия», которое хотелось бы разрешить. Поскольку принципы, по которым выполняется моделирование в соответствии с ISO 15926, не вполне соответствуют принципам представления объектов и отношений между ними, например, в языке OWL, объединение этих двух технологий получилось несколько синтетическим. Данные, построенные в соответствии с ISO 15926, могут быть разложены на элементарные элементы — триплеты RDF, но анализ отношений между информационными объектами, представленной в таком виде модели средствами SPARQL, будет затруднен.

**Семантические модели и их применения**

Основной движущей идеей при создании семантических технологий явилась необходимость обеспечения «понимания» алгоритмами вычислительных машин смысла (семантики) данных. Таким образом, исходной задачей этих технологий была аналитика: обеспечение возможностей извлечения знаний из связанных наборов информации.  
По мере развития этих технологий, экспериментов по их применению в различных сферах, выяснилось, что они исключительно удобны для объединения (связывания) данных из различных по структуре источников. Отсюда возникло второе направление развития инструментов, основанных на идеях семантических сетей – интеграция информационных систем.  
Никакого противоречия между аналитическим и интеграционным применением семантики нет; напротив, они находятся в неразрывном единстве. Ведь целью интеграции, как правило, является извлечение каких-то новых знаний из объединенного набора – таких знаний, которые не могли быть получены из каждого источника в отдельности. Задача упрощения переноса информации из одной информационной системы в другую тоже может быть решена при помощи семантических технологий, но является, скорее, дополнительным бонусом от их развития.

В ряде сфер применения удалось добиться существенных прорывов с использованием семантических технологий анализа информации. Особенно убедительными являются эти успехи в сфере медицины и биотехнологий. Например, на семантических технологиях строятся базы, объединяющие сведения о медицинских препаратах и их действии, клинические истории, генетическую информацию. Анализ таких баз помогает исследователям создавать новые лекарства. Это отличный пример ситуации, когда реляционные базы данных не в состоянии адекватно отразить многообразие связей между информационными объектами, и предоставить инструменты для анализа этих связей – а семантические технологии могут. Также семантические базы данных используются в здравоохранении (для анализа распространения заболеваний), и во множестве других применений.  
Средства анализа информации при помощи семантических технологий входят и в повседневную жизнь. Например, разработчики Facebook Graph Search придумали отличный пример, который позволяет на обыденном уровне продемонстрировать принципиальную новизну семантического поиска (анализа): очевидно, что ни одна из существующих поисковых машин, строящихся по принципу поиска в тексте, не сможет ответить на вопрос «Какие рестораны нравились моим друзьям?», или «В каких городах живут мои родственники?». Поиск по графу, используя формализованный набор информационных объектов (люди, рестораны, города) и отношений между ними (нравится, живет в), способен дать нужный ответ быстро и совершенно точно. При этом условия запроса можно варьировать в тех пределах, которые допускает онтология (набор тех самых типов информационных объектов и связей между ними): аналогичные вопросы можно задать не о ресторанах, а о фильмах, не о родственниках, а об одноклассниках. Понятно, что все содержимое социальной сети Facebook представляет собой огромный единый информационный граф, с миллиардами узлов и связей. Возможность анализа и использования этих связей и составляет всю его ценность, что прекрасно понимают владельцы ресурса.

Опираясь на сказанное выше, мы можем определить критерии, которые должны предъявляться к информационным моделям, строящимся по принципам семантических технологий. Часть этих критериев вытекает из общих требований к моделям, часть – из специфики технологий, связанных с условиями их практической полезности. Перечислим их.  
1. Результат выполнения какого-либо действия в реальной системе и в модели должен совпадать (отношения подобия между моделью и системой в исходном и конечном состоянии описываются одними и теми же правилами). Это требование обеспечивает прогностический потенциал модели: если оно выполнено, мы можем моделировать развитие системы, и воплощать результаты моделирования.  
2. Модель должна отражать свойства объектов и связи между ними таким образом, который делает возможным извлечение знаний из модели при помощи существующих технологий (таких, как SPARQL). Это сугубо практическое требование, обеспечивающее пригодность модели для анализа. Фактически, оно декларирует возможность выполнения расчетов на модели.  
3. Модель должна обеспечивать возможности расширения и масштабирования (укрупнения и детализации), без пересмотра ее онтологического ядра. Это требование налагает ограничения на отбор способов классификации объектов, разграничение объектов и их свойств; это требование можно серьезно детализировать.

<http://habrahabr.ru/post/178973>

3 Практическая часть

Основная идея

Эта часть является описанием основных идей, на которых строится система ПИМ. Ключевая мысль заключается в том, чтобы перенести существующие технологии СВ на персональный компьютер. Ресурсы на компьютеры получают UNI адреса а их метадата конвертируется в RDF. Существующие десктопные приложения включены в эту структуру, а не заменены. Далее в деталях рассмотрено значене UNI в контексте его использования для идентифицирования идей и концепций.

**Semantic Desktop**

СВ позволяет улучшить WWW путем включения дополнительных метаданных. Этот же подход может быть использован и на персональных компьютерах. Подобно Вебу сервер приложений публикует доступную информацию. Локальный Semantic Web server может быть использован для того, чтобы получить доступ к информации расположенной на компьютере. Файлы и другие ресурсы могут быть использованы как ресурсы веб. Они идентифицируются с помощью UNI и позволяют создавать связи между различными документами. Существующая информация представлена в виде RDF документов и может быть использована любым другим приложением. Таким образом проявляется одно из основных свойств семантической паутины – различные программы одинаково обрабатывают данные с семантическими метаданными. Единый стандарт представления информации позволяет упростить взаимодействие между приложениями. В данном случает под приложениями понимаются десктопные программы, однако подобная структура может быть масштабирована до корпоративного уровня или даже глобального.

При использовании технологий семантической паутины на персональном компьютере в системной архитектуре создается семантический уровень. С этого момента ресурсы доступ к ресурсам может быть предоставлен посредством Semantic Web Server параллельно доступа к ним через уже установленные программы. Ресурсы идентифицируются независимо от формата своих файлов. На основе этого семантического уровня появляется возможность построить систему классификации, охватывающую все существующие ресурсы на компьютере. Такие концепции как «личное» или «работа» будучи один раз представлены в системе могут быть использованы всеми приложениями. Все ресурсы могут быть идентифицированы через UNI и просмотрены в браузеро-подобных программах. Поисковик, вроде Google, может быть создан поверх системы для того чтобы создать полнотекстовый индекс всех существующих текстовых данных. В отличие однонаправленных гипертекстовых ссылок в вебе связь между локальными ресурсами является двусторонней.

Отношение между двумя ресурсами может быть установлено в любой точке системы и быть доступно из любых других мест.

Как будет показано в примере привнесение технологий СВ в архитектуру персонального компьютера может быть с пользой использовано различными способами. При получения доступа к ресурсам через Semantic Web server открывает новые возможности по использованию существующих данных.

**Принципы**

**Один компьютер – один пользователь.** Представленная система семантической паутины спроектирована с учетом того, что ею будут пользоваться один пользователь, чьи ресурсы входят в эту семантическую сеть. Другие существующие PIM тоже поддерживают только одного пользователя, но не все. В данном случае ограничение на одного пользователя упрощают систему, так как отпадает необходимость поддерживать нескольких пользователей. Программа работает на одном персональном компьютере. Это не распределенная многопользовательская система и поэтому в ней отсутствуют механизмы пользовательской аутентификации и авторизации. В системе нет встроенной возможности обмена данными, если необходимо поделиться информацией с другими пользователями, она должна быть экспортирована и импортирована вручную. Вся RDF информация хранится на одном компьютере. Таким образом, весь RDF граф принадлежат одному пользователю. Так как каждый пользователь – уникальная личность и может быть идентифицирован, программа SemanticDesktop идентифицируется с помощью *hostname* (имя хоста). Так как имя хоста (доменное имя) глобально уникально, для каждого ресурса на компьютере возможно задать уникальный идентификатор. Если у компьютера нет уникального, доменного имени можно использовать другой способ. Этот способ заключается в том, чтобы использовать альтернативный идентификатор, который является доменным именем, принадлежащим владельцу компьютера плюс какое-нибудь имя для компьютера. Например, у моего компьютера нет уникального доменного имени, но я владею доменом

**http://www.arthur.lazy-magister.org**  
В этом случае уникальное имя для моего компьютера будет

**http://www.arthur.lazy-magister.org/learnmachine**

В данном случае уникальность заключается в том, что пока я являюсь владельцем доменного имени, никто другой не может им воспользоваться. Следует отметить, что, хотя идентификатор и похож на URL, он всего лишь используется как уникальное имя и не указывает на какой-либо ресурс в сети.

**У всего есть URI.**

Для работоспособности системы необходимо наличие у ресурсов отличительных идентификаторов. В настоящее время на подавляющем большинстве персональных компьютеров нет встроенной системы идентификации ресурсов. Некоторые операционные системы на базе ядра Linux частично реализуют механизм идентификации за счет использования семантической файловой системы, однако такие операционные системы очень редки. При использовании PIM у нас должна быть возможность ссылаться на объекты независимо от их принадлежности различным установленным приложениям. Другими словами у нас должна быть возможность получить данные и из контакта в email клиенте, и из музыкального файла, и из документа, расположенного в файловой системе. На данный момент всех эти ресурсы идентифицируются по-разному. Файл определяем его полное имя (адрес), у email’а есть message-id в заголовке, книга идентифицируется по Международному стандартному книжному номеру (ISBN) или по Международному стандартному серийному номеру (ISSN). Таким образом, видно, что существует множество способов идентификации ресурсов и даже несколько для одного типа. Если бы мы решились создать книжный интернет-магазин, нам бы пришлось выбирать, по какому идентификатору сохранять книги в базе данных (ISBN или ISSN). Что бы мы ни выбрали, нам бы пришлось ограничить все данные о книгах одним идентификатором, и у нас не было бы возможности его поменять. Использование же UNI для идентификации ресурсов имеет несколько преимуществ по сравнению с другими ссылочными классификаторами наподобие ISBN или ISSN. UNI адреса глобально уникальны. Их использование не ограничено локальной базой данных и могут быть использованы на мировом уровне. Другими словами, если закрепить какую-либо UNI строку за определенным ресурсом, то во всем мире этот идентификатор бы ссылался именно на этот ресурс. UNI адрес содержит идентифицирующую информацию. Вместо того чтобы поддерживать различные системы идентификации, такие как имя в файловой системе или message-id, мы решили трансформировать все идентификаторы в UNI схему. И теперь для того чтобы сохранить ссылку на любой ресурс на нашем компьютере нам необходим только UNI. Информации по каждому ресурсы может быть запрошена по его уникальному UNI. Нет необходимости указывать тип ресурсы и где его можно найти, вся информация уже содержится в самом идентификаторе.

UNI адрес строится по следующей схеме:

<protocol>://<hostname><path>?<query>#<anchor>

<протокол>://<доменное имя><путь>?<параметры>#<якорь>

Ресурсы, находящиеся на компьютере пользователя получают UNI, который содержит в себе доменное имя, принадлежащее пользователю. Таким образом, использование доменных имен в UNI делает идентификатор ресурса глобально уникальным.

Некоторые ресурсы на персональном компьютере уже идентифицируются подобным образом. Если бы на нашем хосте работал FTP сервер, ftp URI мог бы быть использован для идентификации файлов. Но так как никакого FTP сервера у нас нет мы будем использовать альтернативный способ. URI путь начинающийся на file:// завязан на файловую систему. Путь [file://artur.lazy-magister.org/data](smb://artur.lazy-magister.org/data) будет указывать на нашу домашнюю директорию. Это похоже на открытый доступ по локальной сети к файлам: домашний каталог как удаленный корневой каталог а “data” – его публичное имя. Таким образом, файлы в домашнем каталоге в нашей системе могут быть проиндексированы с помощью URI.

Протоколы в URI схеме такие как http:// и file:// могут быть использованы для доступа и идентификации ресурсов на веб-сервере.

**Вся информация в виде RDF.**

SemanticDesktop берет на себя ту роль, которую в корпоративных системах исполняет Интеграции Приложений Предприятия (*Enterprise Application Integration*, *EAI*) а именно – концентрирует все метаданные в одном месте – в семантическом графе, записанном в виде RDF. Это происходит путем трансформирования разрозненных данных в единый формат. Все вышеперечисленные типы ресурсов должны быть доступны в центральном графе. Приведение всех данных к одному формату позволяет нам использовать единый интерфейс доступа к этой информации без необходимости каждый раз извлекать данные из различных приложений, установленных на нашем компьютере. Доступ к данных происходит через локальный сервер. Также, так как мы используем технологии семантической паутины в качестве формата данных для семантического графа был выбран RDF (Resource Description Framework, Среда Описания Ресурсов). Нотация записи RDF не так важна. Существует две наиболее распространенных нотации: RDF/XML для представления графа семантической паутины в виде структурированного XML документа и N3 «Тройная нотация», которая представляет собой простой текстовый формат. Так как ядром нашей системы служит свободное программное обеспечение Apache Jena, поддерживающая все форматы, мы свободны в своем выборе. Для данной системы я выбрал формат записи RDF/XML как наиболее лаконичный.

Есть два подхода к трансформации данных. Первый показан на рисунке «Трансформация и буферизация. В данном случае задача разделятся на два процесса: извлечение данных и получение данных. Вся необходимая информация извлекается из своих приложения, трансформируется в RDF и буферизуется в базе данных. При изменении ресурсов необходимо обновиться, поэтому за изменением ресурсов всегда должно вестись наблюдение. Извлечение данных и их конвертирование в RDF – это простой процесс. Для хранение информации в виде RDF могут быть использованы существующие RDF базы данных. Эти базы данных поддерживают нативную RDF структуру, и подходят для нашей задачи. Оценив несколько существующих решений, я решил остановиться на базе данных, встроенной во фреймворк Apache Jena. Запросы информации к базе данных проходят быстро, и все содержащиеся в ней данные могут быть проиндексированы. К сожалению, при подобном подходе приходится постоянно следить за изменением внешних ресурсов, что значительно усложняет программу. Кроме того, одним из следствий этого решения является ситуация, когда вся обрабатываемая информация сохранена и в исходных файлах и в базе данных. И хотя на физическую память не налагаются большие ограничения, подобных подход будет потреблять в два раза больше дискового пространства.



Рисунок 5. Трансформация и буферизация

Во втором подходе трансформация исходных данных в RDF происходит по требованию, когда в систему поступает запрос о метаданных конкретного ресурса. Смотри рисунок «Извлечение на лету». Сначала запрос анализируется для того чтобы определить метаданные какого ресурса требуются. Затем происходит проход регистра ресурсов особым алгоритмом, чтобы определить какой именно исходный файл представляет этот ресурс. После этого файл ресурса читается, информация преобразуется в RDF и инициатору запроса. Такой подход замедляет обработку запросов в системе, так как поиск исходных файлов, извлечение информации и ее преобразование в RDF требует времени и происходит при каждом запросе. Преимущество этого метода заключается в том, что преобразование данных из исходных файлов происходит на лету и поэтому для каждого запроса гарантируется возвращение не устаревшей информации.

При создании системы SemanticDesktop был выбран первый подход, так как он обеспечивает концентрацию всей метаинформации о ресурсах персонального компьютера в одном месте в виде семантического графа и позволяет быстрее извлекать данные и проходит по узлам графа.

В следующей главе мы подробно рассмотрим приложения, установленные на персональном компьютере и способ извлечения из них информации.

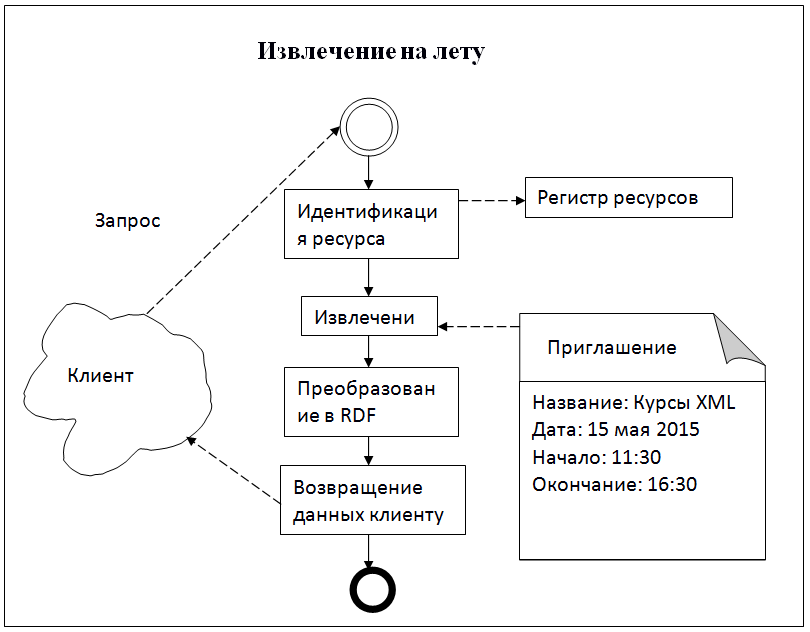


Рисунок 6. Извлечение на лету

**Целевые приложения.**

На обычном персональном компьютере существуют предустановленные программы. Эти программы (приложения) могут манипулировать, к которым операционная система предоставила доступ. Например, приложения Microsoft Word может изменять файлы документами Microsoft Word. Подобные приложения являются лучшим способом создания и изменения информации. Они могут извлекать метаданные, создавать новые файлы, удалять их, печатать их, отображать их и т.п.

В данной работе я решил называть их целевые приложения. В дальнейшем я буду часто использовать этот термин. Эти приложения отвечают за свои файлы (ресурсы). Каждый ресурс должен иметь целевое приложение, иначе его невозможно модифицировать пользователю. Важно заметить, что один из ключевых принципов системы SemanticDesktop подразумевает незаменяемость этих приложением программой семантической паутины. Другими словами, SemanticDesktop работает параллельно с уже установленными программами. Таким образом, существующая информация остается в своем нативном формате и может быть дальше модифицироваться с помощью своего целевого приложения. Система семантической паутины лишь добавляет возможность идентификации любого ресурса и извлечении информации о нем.

SemanticDesktop зависит от целевых приложения установленных на персональном компьютере и от их API (Application Programming Interface, Интерфейс Программирования Приложений), который используется для того, чтобы извлекать информации из ресурсов (файлов) этих приложений. Если пользователь захочет изменить или просмотреть информации ресурса, будет вызвано целевое приложение.

В случае если целевое приложение пригодно только для открытия своих файлов, но не может извлекать метаданные, необходима помощь стороннего программного обеспечения. Например, аудиоплееры обычно могут только проигрывать MP3 файлы, но извлекать и них метаданные, поэтому для этой цели придется использовать стороннюю библиотеку под названием «MP3File».

Основная идея нашей разрабатываемой системы заключается в том, чтобы интегрировать существующие приложения в семантическую сеть, а не заменить их.

**Ассоциативные связи между ресурсами.** Человеческий мозг хорошо справляется с запоминанием ассоциаций. Концепция ассоциативной базы данных хорошо подходит для программ типа Personal Information Management. Программы для создания ассоциативныех карт (mind map) тоже работают по этому принципу. Ресурсы в нашей SemanticDesktop системе тоже могут быть ассоциативно связаны посредством RDF триплетов. Ресурсы (файлы данных) различных программ могут быть соединены друг с другом. Таким образом, мы преодолевая границы между различными приложениями на персональном компьютере. Ресурс может быть связан сразу с несколькими другими ресурсами и классифицирован более чем в одну категорию. Самый простой способ образования связей между ресурсами заключается в том, чтобы создать RDF триплет с предикатом типа relation из вокабуляра Dublin Core, который будет соединять один объект с другим. В данном случае большей проблемой будет создание пользовательского интерфейса, позволяющего ему создавать смысловые связи между ресурсами. Необходимо будет использовать системы онтологий вроде RDFS для того чтобы определить полезные и возможные способы линковки ресурсов различного типа.

**Онтологии.**

В семантической паутине, онтологии использую для того чтобы описать метаданные об определенных ресурсах. Общие системы онтологий являют собой тезаурусы и в них могут быть указаны классификации и возможные свойства различных ресурсов. Двумя распространенными языками описания онтологий являются RDFS и OWL. Мы в системе SemanticDesktop сосредоточимся на использовании RDFS. Онтологии будут необходимы в четырех случаях.

Во-первых, сам сервер SemanticDesktop зависит от онтологий. Внутренние операционные данные, например счетчики, и конфигурационные файлы описаны в виде онтологий. Они записаны на языке RDFS и сохранены как RDF/XML файлы на сервере.

Во-вторых, онтологии описывают информацию, извлекаемую из ресурсов целевых приложений. Например, программы - почтовые клиенты, такие как Microsoft Outlook могут хранить информацию о контактах электронной почты в особых файлах. В данном случае онтологии описывают структуру извлекаемой информации из ресурса. Для различных приложений и ресурсов необходимы разные онтологии.

В третьем случае использования онтологий – это публичные онтологии. Они созданы группой экспертов (таких как консорциум всемирной паутины, W3C) и описывают распространенные типы ресурсов вроде проектов, людей и документов. Эти онтологии опубликованы в интернете и их могут использовать любые заинтересованные стороны. Важная функция публичных онтологий – это глобальная стандартизация элементов семантической паутины. На данный момент подобных онтологий не очень много. В нашем проекте SemanticDesktop мы будем использовать две публичные онтологии.

* “Friend of a Friend”, созданный Dan Brickley и Libby Miller. Эта онтология содержит классы для людей и общих понятий, таких как организации и документы. Основная идея этой онтологии заключается в описании личностей и отношений между ними.
* “Dublic Core”, созданный Dublin Core Metadata Initiative. Эта онтология используется для детального описания метаданных документов. Она содержит исчерпывающий словарь предикатов, который описывает такие свойства как автор, тему, название копирайт документа.

И наконец, третий способ применения онтологий – это пользовательские онтологии созданные пользователями системы. Любой пользователь может решить создать новую онтологию, которая подходит для его специфических нужд. Специфические классы, такие как «нужно будет скоро сделать» и «относится к работе» могут быть созданные в таких онтологиях. Наша система SemanticWeb может использовать пользовательские онтологии, но в ней нет встроенного редактора, поэтому задача создание онтологии полностью ложится на плечи пользователя.

Онтологии могут быть включены просто как файлы RDF/XML или быть сохранены в центральном репозитории (базе данных). SemanticWeb умеет испортировать онтологии посредством веб-интерфейса.

**Совместное использование**

Создание ассоциативной сети ресурсов подобно переносу структуры мыслей пользователя в электронный вид. Полученный в результате семантический граф является очень субъективным и наилучшим образом подходит для пользования только одним человеком. Так как информацией подобного вида тяжело делиться с другими пользователями по сети и для того чтобы упростить разработку нашего прототипа программного обеспечения, функционал совместного использования не был включен в текущую версию продукта. Система является изолированной и в ней отсутствует возможность делиться метаданным с другими системами SemanticDesktop. Также система не поддерживает автоматическую публикацию метаданных и пересылку их посредством Email.

Перед тем как станет возможным обмен семантическими метаданными между пользователями системы, необходимо будет установить некоторые правила совместного пользования подобной информацией. Прежде всего, необходимо чтобы все пользователи сети согласились на использование общей онтологии. Для этого придется либо конвертировать метаданные конкретного пользователя в общий вид, либо заставить всех пользователей изначально пользоваться только заранее обговоренными онтологиями. Последний способ как раз представляет собой одну из основных идей семантической паутины, когда существует всеобъемлющая глобальная онтология и любой граф в этой паутине строится только на ее основе.

Если в будущем будет существовать подобная онтология, система SemanticDesktop может быть расширена посредством добавления модуля для совместного пользования метаданными. А так как для идентификации ресурсов мы уже использует глобальные идентификаторы, добавление подобного функционала не должно вызвать больших проблем.

**Снова о URL**

Всем известно, что в интернете URL используется для идентификации ресурсов сети. Анализируя его составные части – протокол, домен и путь – браузер или другие приложения могут определить местонахождение ресурса в сети и то, как именно получить к нему доступ. Спецификация URL – это один из трех столпов, на которых построена Мировая Паутина: URL, HTTP, HTML. Интернет адреса преодолевают границы персональных компьютеров. Их можно увидеть в рекламе по телевизору, на визитных карточках и даже на товарах массового потребления – адреса давно вошли в повседневную жизнь.

Кажется, вполне ясно, что собой представляет URL и для чего он нужен, но это не совсем так. С самого начала своего существования природа URL была предметом ожесточенных споров. В этом разделе мы подробно опишем роль URL в нашем приложении SemanticDesktop.

**URL и URI – Расположение и Идентификация.** Во времена зарождения интернета существовала только одна концепция – URI (тогда еще расшифровывающийся как Universal Resource Identifier “Универсальный Идентификатор Ресурса”, но потом переименованный Uniform Resource Identifier “Единый Идентификатор Ресурса”), который служил как для идентификации документов в сети так и для указания их расположения. Затем, после обсуждения в IETF (“Internet Engineering Task Force”, Инженерный совет интернета), Был придуман термин URL (Uniform Resource Locator). Различие между ними заключается в том, что при изменении месторасположения документа, его URL меняется, а вот URI остается прежним. В системе SemanticDesktop URI является одновременно и идентификатором и адресом ресурса. URI обрабатывается и анализируется с двумя целями. Во-первых, для того чтобы определить физическое расположение ресурса, и во-вторых, для того чтобы извлечь метаданные этого ресурса. По идентификатору ресурса возможно определить его целевое приложении и с его помощью (или используя стороннее программное обеспечение, если целевое приложение на это не способно) извлекается метаинформация о данном ресурсе в системе.

Это разыменовывание URI для того чтобы определить где находятся ресурсы (домен) и как получить к ним доступ (протокол) являлось одной из причин такого успеха Всемирной Паутины. Существует некоторые до сих пор нерешенные проблемы, касающиеся URI (например, отсутствие поддержки символов национальных алфавитов), но, не смотря на это, URI является лучшим способом глобальной идентификации ресурсов. Однако же вопрос, что же собой представляет URI, остается. В данный момент эта проблема неопределенности получила название «Кризис URI».

**Кризис URI.** Когда задается цель дать возможность пользователям классифицировать различные ресурсы, прежде всего, необходимо определить саму классификацию. Во введении мы показали пример создания классификации. Файлы, например, могут быть поделены на две группы: «личное» и «работа», а для того чтобы выразить это разграничение они должны быть сохранены в двух разных директориях, названных «личное» и «работа». В Семантической Паутине тоже возможно осуществить подобную классификацию. В RDF онтологии необходимо представление классов. Мы можем определить классы в файле RDF схемы и обозначить объекты как экземпляры этих классов. Необходимым условием для этого будет возможность идентифицировать классы с помощью URI. Таким образом, нам нужны идентификаторы для классов «личное» и «работа» и мы будем их использовать в нашем программном обеспечении. Но перед этим надо ответить на вопрос, что собой представляет URI и что он может идентифицировать. Неоднозначность концепции идентификации получила название «Кризис URI». Единственный URI может иметь различные значения. У него может быть четыре аспекта. Например, используя определение концепции лени (lazy) можно показать четыре аспекта. Что идентифицирует следующий URI:

**“http://x.org/lazy”**?

Само название **http://x.org/lazy**, это строка, которая соответствует структуре URI.

* Определенную концепцию лени
* Адрес в сети, по которому располагается документ
* Сам документ, который описывает концепцию лени и к которому можно получить доступ и в дальнейшем обработать.

Когда URI используется в приложениях семантической паутины, он может использоваться во всех вышеперечисленных значениях. W3C Technical Architecture Group (подразделение консорциума Всемирной Паутины), озаботившись разграничением *концепции* от *веб адреса*, разработал два подхода к решению проблемы двойственности. Опишем их коротко.

Различные URI

Первый подход заключается в том, чтобы использовать разные URI для разных контекстов, в которых находится ресурс. Попросту говоря, необходимо использовать новую систему для определения концепций, а URI идентифицирующие веб-адрес уже существуют. URI-концепции могут содержать в себе части, которые бы отличали их от адресов: например, можно использовать другой протокол (rdfp://, info:// и т.д.). Однако это подразумевает создание еще одного протокола передачи данных, что не очень практично. Также можно использовать идентификатор фрагмента (символ #) в конце строки URI для того, чтобы обозначить концепцию (…lazy#concept). Идентификатор фрагмента часто используются для указания на часть веб-документа и, таким образом, может быть использован для идентификации ресурса. И все же идентификаторы фрагмента – неустойчивое решение. Специальное хост-имя в пределах сервера, которое будет содержать в себе только термины концепций, может быть вставлено перед именем домена. Например, сравните “**http://x.org/lazy**” и “**http://concept.x.org/lazy”**. Но для этого понадобится устанавливать отдельный сервер. Недостаток всего этого подхода заключается в том, что URI для идентификации концепции может уже существовать в сети, и невозможно будет различать концепцию и адрес документа.

Различные контексты.

Второй подход – это определение различных контекстов для URI. Попросту говоря, Один и тот же URI используется как «идентификатор концепции» или «идентификатор ресурса». Приложения должны будут интерпретировать URI по-разному в зависимости от того, в каком контексте его рассматривает, ресурс или концепция. Проблема заключается в том, что текущая спецификация RDF не допускает подобного. Также неочевидно, какие правила использования различных концепций.

Решение этой проблемы не было принято сообществом по причине сильной философской направленности. Что «есть» веб-сайт и что «есть» концепция и как их различить? Многие компании сильно привязаны к своим веб воплощениям. Так в чем же отличие сайта “www.amazon.com” и компанией, расположенной в Сиэтле, особенно с точки зрения пользователя. Приложение «Я купил книгу на Амазоне» и «Я купил книгу на www.amazon.com» не особенно отличаются. Что же такое «есть» Амазон и что подразумевают когда о нем думают и говорят? Термин «www.amazon.com» - это символ, который мы используем в нашем языки и он что-то означает. Вопрос о том, что такое символ слишком философский, поэтому я не буду его рассматривать в этой работе. Вместо этого я обращу свое внимание на практическое использование URI для идентификации концепций и вещей.

URI можно использовать для идентификации различных сущностей.

Для того чтобы идентифицировать книгу «Два капитана» Вениамина Каверина мы можем воспользоваться страницей на веб сайте Амазон, на которой описывается эта книга. Если ввести URL адрес в адресную строку программы браузера отобразится веб страница с этой книгой. В высказывании «Я хочу почитать книгу “Два капитана”», URL страницы на Амазоне URL страницы можно использовать для идентификации книги. Существует несколько URI, которые ведут на страницу с этой книгой, поэтому здесь присутствует неточность и двусмысленность. Для людей, URI адреса вполне подходят – чтобы определить на какой ресурс он ссылается достаточно просто вставить его в адресную строку. Также очень просто идентифицировать самих людей, если они сохранены в приложениях вроде электронной книги контактов (адресная книга). После этого можно взять URI записи в адресной книге и использовать его в дальнейшем для ссылки на этого человека, что мы и будем делать в приложении SemanticDesktop.

Проблемы

На текущий момент не существует единой онтологии и списка ресурсов и свойств этих ресурсов. Dublin core и FOAF не обеспечивают необходимой универсальности. В будущем ожидается существование множества онтологий, которые в дальшейшем должны будут объединиться. Консорциум W3C рекомендует пользоваться существующими онтологиями а не изобретать свои.

// TODO добавить в граф семантической паутины

фотографии

автомобили

книги

посещенные места

музыка

кулинария

генеология