Разработка библиотеки для генерации семантических данных

Вагарина Н. С.[1], Апсаликов М. Ю.[2]

[1]Вагарина Наталия Сергеевна / Vagarina Natalia Sergeyevna - кандидат физико-математических наук;
[2]Апсаликов Михаил Юрьевич / Apsalikov Michael Yurievich – магистрант,
направление «Информатика и вычислительная техника»,
кафедра информационных систем и технологии,
международный факультет прикладных информационных технологий,
Саратовский государственный технический университет им. Ю. А. Гагарина, г. Саратов

Аннотация: *идея семантического веба состоит в машинном понимании смысла информации. Это означает, что при разработке семантического приложения потребуется написать два разных слоя представления: один для человека, а другой - машиночитаемый. Это значительно увеличивает время и стоимость разработки. Для решения этой проблемы в статье предложено создание библиотеки для автоматической генерации семантических данных. В статье описаны языковые и программные средства, на основе которых будет создана предлагаемая библиотека и представлен план ее программной реализации.*
Ключевые слова: *семантический веб, модель RDF, платформа ASP.NET, семантический сервис, генерация RDF-данных.*

В настоящее время делаются попытки поиска новых методов и подходов к созданию, предоставлению и обработке информации в связи с тем, что объем информации в вебе колоссален. Один из подходов для решения этой задачи – применение технологий семантического веба. Его концепция является стеком технологий, который позволяет размещать информацию в сети Интернет в виде, пригодном для машинной обработки [1].

В настоящее время технологии семантического веба, также известного как Web 3.0, практически не используются в коммерческих проектах [3]. Это связано, в первую очередь, со значительным увеличением стоимости и времени разработки.

В статье предлагается подход к решению данной проблемы путем создания библиотеки для генерации семантических данных.

Базовая модель семантического Web включает следующие компоненты (в порядке повышения уровня абстракции): URI, XML, RDF, RDF Schema, OWL, OWL Schema, SPARQL, WSDL. Ключевыми технологиями здесь являются RDF и OWL. RDF – это модель представления данных, которая позволяет описать любую информацию в виде триплетов: субъект – предикат – объект [4]. Субъект представляет собой сущность, которую необходимо описать, предикат – свойство субъекта, объект – значение этого свойства. Эта модель имеет несколько сериализаций, также известных как нотации. Однако единственной стандартизированной нотацией является RDF/XML, разработанная на основе XML. Также имеет большую популярность нотация RDF/JSON, основанная на не менее популярном формате сериализации JSON. Существуют и упрощенные сериализации, такие как N-Triples, Turtle и другие.

Технология OWL, являясь более высоким уровнем абстракции над RDF, описывает онтологии.

Онтология – это формализированное описание общепринятого понимания некоторой предметной области, с помощью которого могут общаться как люди, так и компьютерные системы. Это система понятий предметной области, которая представляется набором сущностей, соединенных различными отношениями. Онтологии имеют формальную структуру, поэтому их автоматизированная обработка имеет низкую ресурсозатратность. Онтология состоит из понятий, отношений, аксиом и отдельных экземпляров.

Основная цель Семантического Веба – это возможность семантического поиска информации, которую предоставляет логический вывод. Web 3.0, как и Web 2.0 работают на протоколах HTTP/HTTPS и в качестве идентификатора ресурсов используют URI (Uniform Resource Identifier).

Подавляющее большинство современных веб-проектов объединяет то, что они ориентированы исключительно на человеческое восприятие. То есть смысл, значение и интерпретация информации доступна людям, но не компьютерам. В отличие от этого семантический веб использует идею специальной разметки документов, с помощью которой соответствующие программы смогут самостоятельно собирать разрозненную информацию, сопоставлять данные, принимать решение об их достоверности и даже делать некоторые логические выводы, предоставляя на выходе пользователю готовый информационный продукт. Таким образом, это означает, что если заказчику программного обеспечения нужен продукт, реализующий семантические технологии, то для его разработки потребуется написать два разных слоя представления: один для человека, а другой - машиночитаемый. Данный факт значительно увеличивает время и стоимость

разработки. Как следствие, компании отказываются от использования Семантических технологий в своих проектах.

В качестве решения этой проблемы предлагается вариант, когда код хотя бы одного из слоёв представления будет генерироваться автоматически. С пользовательским интерфейсом в коммерческих приложения это в принципе невозможно, потому что бизнес всегда будет диктовать новые требования и постоянно модифицировать HTML, CSS и JavaScript код. А вот с семантическим предоставлением информации – это возможно. Так как и модель базы данных, и семантическое хранилище являются машиночитаемыми. Теоретически, данную идею можно реализовать как минимум двумя путями. Например, использовать его шаблон проектирования. Очевидно, что это жизнеспособное, но не самое лучшее решение. Второй способ – создание библиотеки под определенную платформу. Этот вариант реализации предпочтительнее, так как позволяет написать код один раз для всех проектов на выбранной платформе.

На настоящий момент самая распространенная веб-платформа в крупных коммерческих проектах это ASP.NET. Библиотека ASP.NET является самым мощным фреймворком для построения веб-приложений. Она содержит в себе такие технологии, как ASP.NET WebForms, ASP.NET MVC и ASP.NET WebAPI. Последние две технологии являются самыми современными на текущий момент. Для большинства запускаемых проектов приложений под платформу .NET выбирают эти библиотеки. Также в качестве положительного решения в пользу использования платформы .NET можно указать богатые возможности языка C#: большое количество «синтаксического сахара» в языке, который значительно увеличивает скорость разработки приложения; удобство компиляции компонентов; улучшенная скорость выполнения кода в связи с тем, что CLR оптимизирует IL код под различные архитектуры процессоров; удобство отладки, предоставленное IDE Microsoft Visual Studio 2013; технология LINQ, позволяющая работать эффективно, обрабатывать данные из коллекций; наличие библиотеки с открытым исходным кодом dotNetRdf для работы с семантическими данными [2]; наличие средств для ускорения разработки, таких как ReSharper. Отдельно следует упомянуть здесь такое преимущество C# перед PHP и большинством других платформ, как метапрограммирование. В нем C# имеет самые мощные возможности на рынке, и с этим может конкурировать только Java. Только эти две платформы предоставляют полноценные возможности рефлексии, которые позволяют восстановить исходный код с точностью до названий переменных. Это связано с тем, что обе платформы используют виртуальные машины: для Java – это JVM (Java Virtual Machine), а для C# - CLR (Common Language Runtime). Использование рефлексии в разрабатываемой библиотеке, несмотря на ее негативные моменты, такие как производительность и риски нарушения целостности данных, значительно упростит код приложений, написанных на этой библиотеке. Для работы с базами данных в программных продуктах, написанных на C#, в том числе и коммерческих, часто используется ORM (Object-Relational Mapping) Entity Framework. Это библиотека с открытым исходным кодом, разработанная Microsoft, которая позволяет работать с базами данных, используя три подхода: Database First, Model First и Code First. Database First позволяет по готовой базе сгенерировать модель, Model First – по готовой модели сгенерировать базу. Эти два подхода считаются устаревшими, так как имеют недостаток: при изменении схемы базы данных необходимо модель пересоздавать. Таким образом, актуален сейчас только один подход Code First. При данном подходе модель собирается из классов C#. Это позволяет отследить изменения модели и применять их (по возможности) автоматически, либо с помощью миграций.

Вследствие всего вышеуказанного, в коммерческих проектах очень часто можно встретить связку ASP.NET + Entity Framework. Более того, разработка логики на этих технологиях занимает в среднем меньше времени по сравнению с аналогичными технологиями на рынке. Именно поэтому для разрабатываемого решения был выбран технологический стек .NET.

В начале работы над библиотекой был проведен анализ похожих существующих продуктов. Но, видимо, в силу малой распространенности технологий семантического веба среди коммерческих проектов, под платформу .NET не было найдено ни одного аналогичного продукта. Существуют решения только в виде шаблонов проектирования. В силу того, что аналогов для платформы .NET было не найдено, был произведен поиск аналогичных программных продуктов для всех платформ. В результате этого поиска была найдена единственная библиотека SuRF или SurfRDF [5]. Она написана на языке Python и соответственно может быть использована только для решений на этом языке. Исходный код проекта находится в свободном доступе на GitHub и доступен по лицензии BSD.

После анализа побочных продуктов и систематизации требований, было решено, что разрабатываемая библиотека должна быть интегрирована с Entity Framework и ASP.NET WebAPI и будет состоять из следующих модулей: реализация шаблона проектирования «Репозиторий»; базовый контроллер WebAPI; базовый контроллер семантического сервиса; интерфейсы для контроллеров; конвертер данных; менеджер конфигурации.

Рассмотрим подробнее каждый из модулей. В коммерческих проектах для работы с базами данных используется шаблон проектирования «Репозиторий». Его суть в том, что он унифицирует доступ к сущностям. В данном случае его необходимо реализовать в библиотеке в связи с необходимостью унификации доступа к данным из контроллеров. Это позволит избежать дублирования кода. Как небольшое дополнение, реализация паттерна в библиотеке позволит опустить его реализацию в самом коммерческом приложении, что снизит затраченные на приложение ресурсы. Шаблон проектирования «Репозиторий» планируется создать средствами обобщенных классов.

Базовые контроллеры WebAPI из семантического сервиса необходимо реализовать для того, чтобы приложение по умолчанию поддерживало базовые операции работы с сущностями. Разработчики приложения унаследуют свои контроллеры от базового и получат всю базовую функциональность, не написав ни одной строки кода. Контроллеры напрямую планируется связать с соответствующим репозиторием при помощи обобщенных классов или рефлексии.

Интерфейсы для контроллеров WebAPI и семантического сервиса планируется создать в целях возможности юнит-тестирования. Это необязательное решение, однако оно может стать ключевым для выбора использования данной библиотеки в коммерческом проекте.

Конвертер данных – это ядро библиотеки, необходимое для конвертации сущностей в семантические данные для последующего предоставления клиентам. Однако в отношениях баз данных есть только две сущности, но нет аналога предиката. Эта проблема может быть решена с помощью атрибутов. Модуль будет считывать значение атрибута и генерировать готовый к использованию триплет. Таким образом, рабочий процесс конвертации сущности в триплет будет выглядеть следующим образом: субъект равен сущности, к которой было произведено обращение; предикат равен информации из аттрибута, который необходимо заполнить разработчику; объект равен сущности, с которой субъект связан. Возможны несколько вариантов, если разработчик не укажет предикат. Самый простой вариант – это предотвращение компиляции приложения. Также возможно сгенерировать уникальный предикат по умолчанию. В разрабатываемой библиотеке планируется реализовать оба варианта и выбирать из них на основе конфигурационного файла. Конвертер данных планируется реализовать при помощи рефлексии.

Так как библиотеку планируется реализовать для реального использования в коммерческих проектах, то необходимо уделить расширяемости большое внимание. Для этого предназначен менеджер конфигурации. Он должен считывать определенную секцию из файла конфигурации и применять соответствующие настройки, как только конфигурация была изменена.

Итак, в данной статье предлагается подход к решению такой проблемы семантического веба, как дублирование кода, которую можно решить с помощью генерации семантического слоя представления. Для этого предложено создание библиотеки под платформу .NET, которая представляет определенные перспективы при использовании в коммерческих проектах. Предполагается, что библиотека будет иметь востребованность среди проектов на платформе .NET. В любом случае, реализация данной библиотеки является шагом вперед по пути развития, использования и внедрения технологий семантического веба. Кроме того, разработанную библиотеку можно будет использовать в учебных целях, что обеспечит потребность в методическом обеспечении преподавания технологий семантического веба и явится продолжением ряда уже существующих продуктов [6, 7].

Литература

1. *Berners-Lee T.* The Semantic Web. A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities [Электронный ресурс] / T. Berners-Lee, J. Hendler, O. Lassila. Режим доступа: http://jeckle.de/files/tblSW.pdf (дата обращения 11.07.2015).
2. C#.Net library for RDF [Электронный ресурс]. – Режим доступа: http://www.dotnetrdf.org (дата обращения 13.06.2014).
3. *Kashyap V.* Real World Semantic Web Applications / V. Kashyap, L. Shklar. – IOS Press, 2002. – 197 с.
4. *Klyne G.* Resource Description Framework (RDF): Concepts and Abstract Syntax [Электронный ресурс] /G. Klyne J. Carroll. ─ Режим доступа: http://www.w3.org/TR/rdf-concepts/ (11.07.2015).
5. Python library for RDF SuRF [Электронный ресурс]. – Режим доступа: https://code.google.com/p/surfrdf/ (дата обращения 13.06.2014).
6. С. 2015615980 от 28.05.2015 Российская Федерация. MakeSense / Вагарина Н. С., Апсаликов М. Ю.; Правообладатель Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Саратовский государственный технический университет имени Гагарина Ю. А.» (СГТУ имени Гагарина Ю. А.) (RU).-№ 2015612914; заявл. 13.04.15.

7. С. 2015615979 от 28.05.2015 Российская Федерация. Программный комплекс для изучения языка запросов SPARQL / Вагарина Н. С., Попов С. М.; Правообладатель Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Саратовский государственный технический университет имени Гагарина Ю. А.» (СГТУ имени Гагарина Ю. А.) (RU).-№ 2015612916; заявл. 13.04.15.