OSTIS-2016
(Open Semantic Technologies for Intelligent Systems)

УДК 004.8

ПРИНЦИПЫ И ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ SC-ХРАНИЛИЩА ПРОЕКТА OSTIS И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НА ПЛАТФОРМЕ .NET FRAMEWORK

Каешко А.И.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

ondister@gmail.com

Рассмотрен вопрос практического использования программной реализации sc-памяти проекта OSTIS в связке с программным обеспечением, реализованным на языке C#. Рассмотрены команды sctp-протокола для сетевого взаимодействия с sc-хранилищем.
Ключевые слова: OSTIS, C#, .Net Framework, sctp, sc-хранилище.

Введение

За последние несколько лет программная реализация технологий проекта OSTIS шагнула далеко вперед. Программная реализация sc-хранилища вполне может конкурировать с такими продуктами как Neo4j и HyperGraphDB. Тем не менее, при подробном изучении проекта OSTIS становится очевидным недостаток информации, касающейся программной реализации, как sc-хранилища, так и внешних инструментов взаимодействия с ним. Более того, обилие довольно сложной для понимания человеком, незнакомым с OSTIS, информации, посвященной теоретическим основам проекта, делает порог вхождения еще выше.

Итак, проект OSTIS [OSTIS, 2015] предназначен для проектирования интеллектуальных информационных систем и их компонентов. При этом такие системы будут основаны на знаниях, представленных в виде семантической сети. Однако информационные системы, реализуемые на основе технологии OSTIS, могут быть и не совсем интеллектуальными, если не стоит такой задачи, а просто служить для накопления информации, формализованной сложной моделью.

Вне зависимости от назначения, процесс проектирования интеллектуальной системы на основе технологии OSTIS производится в два основных этапа [Корончик, 2013]:

- разработка логико-семантической модели системы (sc-модель);
- разработка программного интерпретатора этой модели;

Однако это не означает, что инициировав создание информационной системы на основе технологии OSTIS каждый раз необходимо создавать новый программный интерпретатор модели. Предполагается, что технология OSTIS, является компонентной, и компоненты, в том числе и программные, создаются сообществом OSTIS. Так по состоянию на декабрь 2015 года уже созданы программные реализации sc-хранилища [sc-machine, 2015], языка SCP [SCP, 2015], а так же набор программных средств, ориентированный на взаимодействие системы с внешней средой. Базовые компоненты: sc-хранилище и интерпретатор языка SCP реализованы на языке C для 32-х и 64-х битных версий операционных систем семейства Linux и 64-х битных систем Windows.

1. Sc-хранилище

Sc-хранилище это информационная подсистема, предназначенная для хранения sc-графов в виде семантической сети [Корончик, 2013]. В упрощенном понимании оно является аналогом noSql базы данных. Семантическая сеть в sc-хранилище строится из 3 базовых элементов: узла, коннектора и ссылки. Эти элементы и образуют конструкции, называемые sc-графами, которые составляют семантическую сеть. Каждый элемент хранилища имеет уникальный адрес, состоящий их сегмента (segment) и смещения (offset). Sc-графы вне хранилища можно описать посредством семейства sc-языков, например линейного (scs) или графического (scg). Существует также транслятор

scs и scg кода в sc-хранилище [sc-machine, 2015].

Однако sc-хранилище отвечает не только за организацию хранения sc-графов, оно также предоставляет программный интерфейс для добавления, удаления и извлечения хранимой информации. На программном уровне это возможно осуществить через динамическую библиотеку sc-памяти, которая управляет sc-хранилищем. Однако, учитывая, что эта динамическая библиотека реализована на языке C, подключить ее к таким высокоуровневым языкам программирования, как C# или Java довольно проблематично.

Для решения этой проблемы, а так же для организации сетевого взаимодействия был разработан протокол sctp (Semantic Code Transport Protocol). Тем самым открылись возможности для управления семантическим хранилищем с использованием практически любого языка программирования.

2. Sctp-протокол

Sctp-протокол является бинарным протоколом обмена данными между sctp-сервером и sctp-клиентами. Sctp-сервер использует библиотеку sc-памяти для выполнения команд, поступивших от клиента, и возвращает клиенту результаты выполнения этих команд. Сервер уже реализован на языке C++ и является программным компонентом проекта OSTIS. Каждая команда состоит из заголовка и аргументов. Сами команды довольно подробно описаны в документации sctp-протокола [sctp, 2015]. Предполагается, что протокол будет мультиверсионным, однако в настоящее время существует только одна его версия. Так же предполагалось, что sctp-сервер будет асинхронным, однако в дальнейшем от асинхронной работы пришлось отказаться и сейчас сервер работает только в синхронном режиме приема и передачи данных.

Итак, протокол содержит описание, а сервер предоставляет обработку следующих основных команд:

- Проверка существования элемента с указанным адресом;
- Создание новой дуги указанного типа с указанным начальным и конечным элементом;
- Создание новой ссылки;
- Создание нового узла указанного типа;
- Создание подписки на событие;
- Удаление элемента с указанным адресом;
- Удаление подписки на событие;
- Запрос всех произошедших событий;
- Поиск элемента (узла) по его системному идентификатору;
- Поиск всех ссылок с указанным содержимым;
- Получение начального и конечного элементов дуги;
- Получение содержимого ссылки по ее адресу;
- Поиск конструкций;
- Поиск сложных конструкций;
- Установка содержимого ссылки;
- Установка системного идентификатора узла.

Как видно из перечня команд, sctp-протокол предоставляет широкие возможности по управлению sc-хранилищем, за исключением модели управления доступом. Поэтому при разработке реальных информационных систем работу с sc-хранилищем посредством sctp-протокола лучше оставлять полностью на серверной стороне, а общение сервера и клиента реализовать с использованием безопасных протоколов передачи данных. Так, например, реализован пользовательский интерфейс интеллектуальной метасистемы поддержки проектирования интеллектуальных систем [Корончик, 2014].

3. Реализация команд sctp-протокола на платформе .Net Framework

Изначально sctp-клиент реализовывался как часть проекта OSTIS.MED, посвященного созданию инструментария для создания медицинских информационных систем. И язык платформа .Net Framework была выбрана за ее универсальность и скорость написания кода.

Как уже указывалось ранее, sctp-протокол бинарный, поэтому основную сложность при реализации клиента на платформе .Net Framework составляли методы преобразования объектов в байтовые массивы.

Реализация клиента велась с полным следованием объектно-ориентированной парадигмы программирования. В результате было реализовано 2 библиотеки. Они не используют никаких сторонних библиотек и реализованы на платформе .Net Framewark 4.5. Таким образом, они не совместимы с операционной системой Windows XP. Первая библиотека, ostis.sctp, создавалась, как низкоуровневая библиотека, и содержит только команды протокола [dotnet.sctp, 2015]. Она поддерживает как синхронный, так и асинхронный режим работы. Однако работа с библиотекой крайне неудобна и потребует большого количества повторяемого кода. Для увеличения скорости создания приложений и удобства написания кода была создана библиотека ostis.sctp.tools, которая зависима от первой, и имплементирует абстрактную базу знаний. Кэширование коллекций часто используемых элементов базы знаний позволяет ускорить доступ к ним в реальных приложениях. Кроме этого, она содержит так называемые неатомарные команды, которые отсутствуют в описании протокола, например получение системного идентификатора узла по его адресу. Безусловно, быстродействие второй библиотеки ниже, чем базовой, однако в реальных приложениях часто важнее оптимизация цикла создания кода, чем быстродействие конечного продукта.

В процессе разработки клиента было выполнено полное документирование кода, Все команды снабжены примерами работы, а особо сложные, для лучшего понимания снабжены иллюстрациями. Так же были разработаны модульные тесты, объем кода, покрытого тестами, составил 85,6%. Работоспособность библиотеки была протестирована на операционных системах Windows (7, 8.1, 10 64 bit), Linux (Ubuntu 14.2 32 bit с фреймворком Mono).

Таким образом, удалось создать библиотеку для работы с sc-памятью на языке высокого уровня с явной типизацией, поддерживающего объектно-ориентированную парадигму программирования.

Сами библиотеки довольно хорошо документированы, поэтому подробное описание их архитектуры в рамках статьи излишне.

4. Принципы и примеры использования sctp.Net клиента

Работа с базовой библиотекой клиента происходит в несколько этапов:

- соединение с сокетом сервера;
- подготовка команды;
- отправка команды;
- получение ответа;
- работа с ответом от сервера.

Ниже представлен код метода для создания узла в базе знаний с использованием базовой библиотеки:

```
public void TestCreateNodeSync()
    {
        const string defaultAddress = SctpProtocol.TestServerIp;
        string serverAddress = defaultAddress;
        int serverPort = SctpProtocol.DefaultPortNumber;
        sctpClient = new SctpClient(serverAddress, serverPort);
        sctpClient.Connect();
        if (sctpClient.IsConnected == true)
        {
            var commandCreate = new CreateNodeCommand(ElementType.ConstantNode_c);
            var responseCreate = (CreateNodeResponse)sctpClient.Send(commandCreate);
        }
    }
```

При такой работе необходимо отслеживать доступность сокета и наличие активного соединения, если необходимо выполнить несколько команд одновременно. Более того, необходимо обрабатывать ошибки.

Использование библиотеки ostis.sctp.tools позволяет сократить объем кода и обработать ошибки:

```
public void TestNodeTools()
    {
        KnowledgeBase kb = new KnowledgeBase(Sctp.SctpProtocol.TestServerIp, Sctp.SctpProtocol.DefaultPortNumber);
        Node node = new Node(Sctp.ElementType.ConstantNode_c);
        kb.Nodes.Add(node);
    }
```

Более того, при использовании этой библиотеки можно использовать несколько баз знаний одновременно. В библиотеке второго уровня предусмотрено кэширование часто используемых элементов, что позволяет сократить время доступа к ним. Обычно в базе знаний существует множество неизменяемых отношений. На момент написания статьи библиотека ostis.sctp.tools реализована не полностью. Достаточно полно реализованы действия с узлами, а с дугами и ссылками работать можно только через свойство Commands класса базы знаний, который имплементирует все команды библиотеки ostis.sctp и содержит несколько сборных команд.

Несмотря на все достоинства при реальном использовании клиентских библиотек выявлен ряд недостатков. В первую очередь, это особенности работы с событиями и не достаточно проработанная обработка ошибок. Так, запрос произошедших событий не всегда возвращает все события. Кроме того, при реальной работе приходится производить множество обращений к базе знаний для выполнения достаточно простых операций. Поэтому производительность реальных приложений оставляет желать лучшего. Скорее всего, на данном технологическом этапе придется создавать гибридные системы, где будут использоваться несколько типов хранилищ данных, включая реляционные базы.

Заключение

Для применения технологии OSTIS в студенческих, магистерских и аспирантских проектах уже разработаны sctp-клиенты для языков Python, Java, C++. Однако степень их проработки, использованные парадигмы программирования различаются. Созданный нами клиент для sctp-сервера на языке C# может быть использован для создания как учебных, так и реальных приложений на различных платформах и языках программирования, которые поддерживает .Net Framework. Благодаря хорошей документации проект может быть использован людьми, слабо знакомыми и с технологией OSTIS и с языком C#. А sc-хранилище может быть использовано не только для реализации интеллектуальных систем, но и в проектах, где требуется гибкое и высокоэффективное семантическое хранилище, в том числе и в гибридных системах.

Библиографический список

[OSTIS, 2015] Открытая семантическая технология проектирования интеллектуальных систем [Электронный ресурс]. – 2015. - Режим доступа: http://ostis.net. – Дата доступа: 15.12.2015.

[Корончик, 2013] Реализация хранилища унифицированных семантических сетей. – В кн Междунар. научн.-техн. конф. «Открытые семантические технологии проектирования интеллектуальных систем» (OSTIS-2013). Материалы конф. [Минск, 21-23 февр. 2013 г.]. – Минск: БГУИР, 2013, с. 125-128.

[Корончик, 2014] Пользовательский интерфейс интеллектуальной метасистемы поддержки проектирования интеллектуальных систем. – В кн Междунар. научн.-техн. конф. «Открытые семантические технологии проектирования интеллектуальных систем» (OSTIS-2014). Материалы конф. [Минск, 20-22 февр. 2014 г.]. – Минск: БГУИР, 2014, с. 79-82.

[Голенков, 2012] Графодинамические модели параллельной обработки знаний / В. В. Голенков, Н. А. Гулякина // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем» - Минск, 2012

[**sctp, 2015]** Протокол sctp [Электронный ресурс]. – 2015 – Режим доступа: https://github.com/deniskoronchik/sc-machine/wiki/sctp – Дата доступа: 02.12.2015

[**sc-machine, 2015]** sc-память [Электронный ресурс]. – 2015 – Режим доступа: https://github.com/deniskoronchik – Дата доступа: 02.12.2015

[**SCP, 2015]** Интерпретатор SCP [Электронный ресурс]. – 2015 – Режим доступа: https://github.com/ShunkevichDV/scp-machine – Дата доступа: 02.12.2015

[**dotnet.sctp, 2015]** Реализация протокола sctp на платформе .Net Framework [Электронный ресурс]. – 2015 – Режим доступа: https://github.com/ondister/dotnet_sctp_client – Дата доступа: 07.12.2015

PRINCIPLES AND SOFTWARE IMPLEMENTATION OF INTERACTION OSTIS SC-MEMORY AND .NET FRAMEWORK PLATFORM

Kayeshko A.I.

Belorussian state university of informatics and radioelectronics, Minsk, Republic of Belarus

ondister@gmail.com

The article about of the practical use of a software implementation, OSTIS sc-memory in conjunction with the software is implemented in C #. Shows sctp-protocol usage for networking cooperation with sc-storage.

Introduction

Over the past few years, the software implementation of the OSTIS project technology has leaped forward. Software implementation sc-store can compete with no SQL databases such as Neo4j and Hyper Graph DB. However, the detailed study of the project OSTIS becomes apparent lack of information regarding the software implementation, as the sc-storage and external tools to interact with it. Moreover, the abundance of pretty difficult to understand a person unfamiliar with OSTIS, information dedicated to the theoretical foundations of the project, does the threshold of entering higher.

Main Part

Sc-store this is information subsystem for storing sc-graphs in a semantic network form. However, sc-store is responsible not only for the organization of storage sc-graphs, it also provides a programming interface to add, delete, and retrieve stored information.

For the organization of networking interface was developed sctp-protocol (Semantic Code Transport Protocol). Thus opening the possibility for managing semantic repository with almost any programming language.

Sctp-protocol is a binary protocol communication between the sctp- server and sctp-clients. Sctp-server uses the library sc-memory to execute commands received from the client, and returns to client results of those commands. Server has already been implemented in C ++ and is a software component of the OSTIS project. We have implemented FTP client in C # language.

Conclusion

Due to to the good documentation of the project can be used by people poorly familiar with the technologies both the OSTIS and the C #. A sc-storage can be used not only for the implementation of intelligent systems, but also in projects requiring highly flexible and semantic storage, including hybrid systems.