СОЗДАНИЕ ВЕБ-СЕРВИСОВ НА ЯЗЫКЕ ПРОЛОГ

А. С. АНТОНОВ, А. А. КРИЖАНОВСКИЙ •

Санкт-Петербургский институт информатики и автоматизации РАН

СПИИРАН, 14-я линия ВО, д. 39, Санкт-Петербург, 199178

УДК 519.2

Антонов А. С., Крижановский А. А. **Создание веб-сервисов на языке Пролог** // Труды СПИИРАН. Вып. 5. — СПб.: Наука, 2007.

Аннотация. *Рассмотрены возможности создания веб-сервиса исключительно на языке Пролог без использования других языков программирования: описаны как основные возможности разбора и генерации веб-документов с помощью языка SWI-Prolog, так и различные модели HTTP серверов и клиентов, поддерживаемых языком SWI-Prolog.* — Библ. 2 назв.

UDC 519.2

Antonov A. S., Krizhanovsky A. A. **Composition of web-services using Prolog** // SPIIRAS Proceedings. Issue 5. — SPb.: Nauka, 2007.

Abstract. *The possibility of web-service creation with the help of pure Prolog (without other programming languages) is reviewed. The main features of web-documents parsing and generating by SWI-Prolog are described. The different models of HTTP servers and clients supported by SWI-Prolog are presented.* — Bibl. 2 items.

1. Введение

Постоянное развитие и расширение всемирной сети Интернет ведет к увеличению объемов обрабатываемой в сети информации. Перспективным способом организации доступа к вычислительным ресурсам Интернет являются веб-сервисы, обладающие такими достоинствами как: *инкапсуляция* (сокрытие данных), *связывание* (сервис может включать в себя ряд более простых сервисов) и стандартизация описания (описано ниже). Тем не менее остаётся вопрос первичной обработки данных Интернет.

Данные в Интернете зачастую представлены с помощью некоторого формального языка или не формализованы вовсе, то есть написаны на естественном языке. Поэтому, при разработке веб-приложений, обоснован интерес к языкам программирования (в том числе и к Прологу), направленным на обработку полуструктурированной и неструктурированной информации. Существуют два взгляда на использование Пролога в веб-приложениях. Наиболее часто Пролог используется для создания встроенного в основную программу модуля, реализующего необходимые функции, такие как поиск или работа с ограничениями. Более интересным представляется вариант создания самостоятельного веб-сервиса на языке Пролог, который будет общаться с другими сервисами посредством протокола HTTP, не используя для этого надстроек, написанных на языках высокого уровня. Для создания подобного сервиса необходима не только поддержка коммуникационного протокола, но также необходима реализация разбора, представления и генерации веб-документов в таких форматах как HTML и XML.

• Исследования, результаты которых представлены в настоящей работе, были частично выполнены в рамках проектов РФФИ # 05-01-00151 и # 06-07-89242, проекта # 16.2.35 программы «Математическое моделирование и интеллектуальные системы» Президиума РАН, проекта # О-1.9 программы «Фундаментальные основы информационных технологий и компьютерных систем» ОИТВС РАН.

Язык Пролог имеет множество реализаций, среди них выделим SWI-Prolog[1], активно развивающийся с 1987 года и распространяемый сегодня по свободной лицензии. Данная реализация имеет большое количество библиотек, расширяющих стандартный Пролог. Также стоит отметить, что SWI-Prolog имеет реализации для большинства современных платформ (Windows, Linux, MacOS X), что обеспечивает переносимость создаваемых программ. Ниже рассмотрим более подробно разбор и представление на языке SWI-Prolog веб-документов и реализацию HTTP клиента и сервера. Из других интересных реализаций можно отметить открытые проекты: XSB[2] и GNU-Prolog[3].

2. Разбор и представление HTML и XML документов

Одним из достоинств веб-сервисов является стандартизация их описания. Веб-сервисы предоставляют общедоступные интерфейсы (например, SOAP, WSDL), протоколы которых основаны на формате XML, что позволяет другим программам взаимодействовать с данными сервисами. В тоже время основным языком представления информации в Интернете является HTML. Таким образом, необходимо выполнять разбор XML документов на стороне сервиса и генерировать ответные XML сообщения (для программ) или HTML страницы (для пользователей).

В основе языка XML лежит древовидная структура представления информации с использованием специальных тегов. SWI-Prolog использует для разбора подобных документов следующую структуру термов:

```
<document> ::= list-of <content>
<content> ::= <element> | <pi> | <cdata> | <sdata> | <ndata>
<element> ::= element(<tag>, list-of <attribute>, list-of <content>)
<attribute> ::= <name> = <value>
<pi> ::= pi(<atom>)
<sdata> ::= sdata(<atom>)
<ndata> ::= ndata(<atom>)
<cdata>, <name> ::= <atom>
<value> ::= <svalue> | list-of <svalue>
<svalue> ::= <atom> | <number>
```

На использовании данной структуры основаны два интерфейса разбора веб-документов в SWI-Prolog. Базовым является метод *load_structure(+Source, -ListOfContent, +Options),* который преобразовывает поточную информацию в набор термов, описанных выше. Данный метод является базовым для других методов, направленных на разбор документов конкретного формата, переводящих тэги входного языка в триплеты вида *element(Name, Attributes, Content),* представляющие из себя термы языка Пролог. Таким образом, следующий исходный файл *test.html:*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">

<html>
     <head>
          <title>Demo</title>
```

[1] <http://www.swi-prolog.org/>
[2] <http://xsb.sourceforge.net/>
[3] <http://www.gprolog.org>

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

```
    </head>
    <body>
        <h1 align=center>This is a demo</h1>
    </body>
</html>
```

будет преобразован в следующий набор термов:

```
[element(html,
         [version= -//W3C//DTD HTML 4.0 Transitional//EN],
         [element(head,
                  [],
                  [element(title, [], [Demo])]
                 ),
          element(body,
                  [],
                  [, element(h1, [align=center], [This is a demo]),
                  ])
         ])
]
```

Другой метод, *sgml_parse/2,* позволяет разбирать веб-документы, основываясь на событиях, что удобно при анализе документов большого объема.

В основе языка HTML также лежит древовидная структура представления информации, что позволяет формализовать HTML документы следующим набором термов:

```
<html> ::= list-of <content> | <content>
<content> ::= <atom>
            | <tag>(list-of <attribute>, <html>)
            | <tag>(<html>)
            | \<rule>
<attribute> ::= <name>(<value>)
<tag>, <entity> ::= <atom>
<value> ::= <atom> | <number>
<rule> ::= <callable>
```

Наиболее прост метод создания HTML-файлов с использованием таких примитивов языка Пролог как *write/1* и *format/2,* например *format('bold').* Более удобным представляется метод *print_html/1*, опирающийся на представленный выше набор термов, обеспечивающий лучшее форматирование выходного файла и имеющий семантику термов языка Пролог. Приведенный выше фрагмент будет иметь следующую запись: *print_html(b(bold)).*

На представленные выше структуры описания опираются методы библиотеки *sgml_write.pl,* предназначенные для генерации веб-документов и позволяющие генерировать как XML, так и HTML документы на основе термов Пролога.

Таким образом, SWI-Prolog предоставляет полный набор библиотек, для обработки и генерации веб-документов различных форматов.

3. Поддержка протокола HTTP

HTTP (HyperText Transfer Protocol) является основным протоколом обмена веб-документами. Все существующие веб-браузеры и веб-серверы поддержи-

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

вают данный протокол. Первая версия протокола подразумевала взаимодействие, при котором клиентский запрос состоял из единственной строки формата <действие><путь>, а сервер отвечал запрашиваемым документом и закрывал соединение. На сегодняшний день основным является HTTP протокол версии 1.1, поддерживающий расширенный набор функций как со стороны сервера, так и со стороны клиента.

При создании веб-сервиса на языке Пролог возможны два подхода к реализации поддержки HTTP протокола. Первый, и более предпочтительный — создание HTTP сервера на языке Пролог, что избавляет от необходимости установки на компьютере дополнительного программного обеспечения для реализации HTTP сервера и от необходимости связывания Пролог-программы с этим сервером. Второй — использование HTTP сервера, созданного другими программными средствами, что неудобно, поскольку требует реализации дополнительных модулей, обеспечивающих взаимодействие стороннего HTTP сервера и Пролог-программы. Создание подобного модуля может свестись к реализации на языке высокого уровня (C++, Java) некоторой прослойки между сервером и Пролог-программой, что в свою очередь потребует создания дополнительных библиотек для связи выбранного языка высокого уровня с Прологом.

3.1. HTTP клиент

Рассматриваемый язык SWI-Prolog поддерживает две реализации HTTP клиента. Первый — простой HTTP клиент, поддерживающий исключительно HTTP метод GET посредством терма http_open(+URL, -Stream, +Options). Данный метод перенаправляет в стандартный поток ввода информацию с заданного адреса. Опции позволяют указать время ожидания, адрес прокси-сервера и другие параметры соединения. Метод http_open/3 обрабатывает только HTTP 3xx ответы. Остальные ответы считаются исключениями. Данный метод предоставляет такой же простой доступ к HTTP ресурсу как к локальному файлу. Второй HTTP клиент реализуется методами библиотеки http_client.pl и предоставляет поддержку HTTP метода POST.

3.2. HTTP сервер

В рассматриваемой реализации языка Пролог поддерживаются различные модели HTTP сервера. Первый — опирается на библиотеку *thread_http.pl* и реализует многопоточную модель HTTP сервера, аналогичную моделям, реализованным в Microsoft.NET и SUN JavaBeans. Данный сервер также может использоваться для реализации HTTPS серверов, поддерживающих протокол безопасной передачи данных SSL. Другой вариант — применяется в Unix системах и использует супер-сервер inetd, который стартует новый экземпляр сервера для каждого поступающего запроса. Особенностью данного сервера является то, что он не поддерживает опцию задания порта. Порт для сервера задается настройками inetd сервера.

Таким образом, запуск HTTP сервера осуществляется вызовом соответствующего терма. Приведём в качестве примера генерацию HTML-страниц на сервере локального компьютера, при этом первая страница будет содержать ссылку на вторую:

Труды СПИИРАН. 2007. Вып. 5. ISSN 2078-9181 (печ.), ISSN 2078-9599 (онлайн)
SPIIRAS Proceedings. 2007. Issue 5. ISSN 2078-9181 (print), ISSN 2078-9599 (online)
www.proceedings.spiiras.nw.ru

```
:- use module(library('http/thread httpd')).
:- use module(library('http/http dispatch')).
:- use module(library('http/html write')).

server(Port)  :- http server(http dispatch,  [port(Port)]).
:- http handler('/', root,  []).
:- http handler('/hello/world', hello world,  []).

root( Request)  :-
        reply html page([ title('Demo server')
             "    "      ],
                         [ p(a(href('hello/world'), hello))
                         ]).

hello world( Request)  :-
        reply html page([ title('Hello World')
             "    "      ],
                         [ h1('Hello World'),
                           p('This is my first page')
                         ]).
?- server(5000).
```

4. Заключение

Данная работа показывает гибкость языка Пролог и возможность его применения в такой современной области как создание веб-сервисов. Приведён тестовый пример генерации нескольких HTML страниц с помощью SWI-Prolog.

Объединение разнородных вычислительных ресурсов требует решения следующих задач [2]:

1. обеспечение логической связности *(logical connectivity)* разных приложений, написанных с использованием различной терминологии;
2. разрешение конфликтов между отправителем и получателем.

«Оборачивание» вычислительных ресурсов с помощью веб-сервисов и доступ к ресурсам через XML-протоколы, а также использование онтологий позволяют решить первую задачу. С решением второй задачи, на наш взгляд, могут успешно справиться программные системы, написанные на языках логического программирования [1].

Литература

1. *Wielemaker J., Huang Z., van der Meij L.* SWI-Prolog and the Web // Theory and Practice of Logic Programming (TPLP). 2007. (to appear) // <http://arxiv.org/abs/0711.0917> (по состоянию на 14.11.2007).
2. *Bressan S., Fynn K., Goh C. H., Madnick S. E., Pena T., Siegel M.* Overview of the Prolog implementation of the COntext INterchange prototype // In Proceedings of The Fifth International Conference and Exhibition on The Practical Applications of Prolog. 1997. // <http://interchange.mit.edu/coin/publications/sigmod97/sigmod97.pdf> (по состоянию на 14.11.2007).