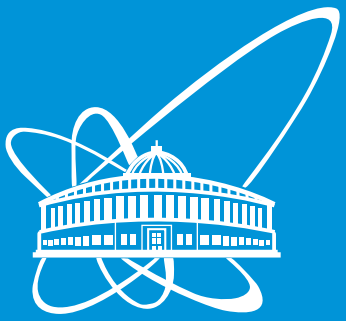

СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ

Дубна

Р10-2000-44

В.В.Галактионов

РАСШИРЯЕМЫЙ ЯЗЫК РАЗМЕТКИ XML (EXTENSIBLE MARK-UP LANGUAGE): ПРОМЫШЛЕННЫЙ СТАНДАРТ, ОПРЕДЕЛЯЮЩИЙ АРХИТЕКТУРУ ПРОГРАММНЫХ СРЕДСТВ ИНТЕРНЕТ СЛЕДУЮЩЕГО ПОКОЛЕНИЯ

2000

1. Введение

Язык гипертекстовой разметки HTML до сих пор является практически единственной технологией для разработки WWW-приложений как основных средств использования сети Интернет. HTML представляет собой набор определений тегов и их атрибутов, записанный на метаязыке SGML (Standard Generalised Markup Language) [1]. Простота и доступность языка HTML не в последнюю очередь обусловили появление новой массовой профессии – разработчиков (верстальщиков) так называемых WEB-страничек. Язык HTML достаточен для подготовки простых статических информационных систем и в техническом плане не требует сложной специальной подготовки для его использования. Однако расширение сферы применения WWW для прикладных задач потребовало новых видов услуг и типов обработки информации – интерактивного режима, обеспечения работы с базами данных, структурами данных и др. В этом плане единственным значимым явлением в преодолении статичного характера отображения информации в WWW была разработка механизма применения программируемых модулей в HTML-документах – Java-технологий (языка сценариев JavaScript и Java-апплетов), реализованных в наиболее популярных браузерах Internet Explorer (Microsoft corp.) и Netscape Navigator (Netscape). На какое-то время текущие потребности пользователей были удовлетворены, но дальнейшие "расширения" конструкторских возможностей языка HTML (стилевые таблицы CSS, DHTML, HTML 4.0 и др.) скорее напоминали "заплатки на расползающейся ткани" [4].

В дальнейшем же, особенно в связи с широким внедрением технологий WWW для индустриальных и бизнес-приложений, которые требуют взаимодействия с базами данных и обработки структурированных данных, сложилась ситуация, трудно разрешимая существующими стандартными средствами.

Основные недостатки HTML-технологий и пути преодоления возникших трудностей были впервые сформулированы Ионом Босаком (Jon Bosak), членом экспертной группы консорциума WWW (W3C, http://www.w3.org), сотрудником компании Sun и уже названным отцом новой WWW- технологии – языка XML (eXtensible Markup Language, "Расширяемый язык разметки") в публикациях [2, 3].

Наиболее часто критикуемые недостатки HTML:

- отсутствие средств обработки структурированных данных, и в этой связи особенно проявляющиеся трудности при работе с базами данных;
- трудности обработки HTML-документов для поисковых систем, поскольку средства HTML в большей степени ориентированы на способ формирования изображения, чем на описание содержательной части информации;
- ограниченный набор жестко определенных тегов разметки текста;
- очень упрощенная система гиперссылок.

Согласно рекомендациям W3C от 1998 года, XML представляет собой

компактное упрощенное подмножество языка SGML. Как и SGML, XML — это метаязык, определяющий другие языки разметки для специфических целей. Главная задача нового языка - предоставить пользователю самому создавать теги для определения элементов содержимого документа. В XML нет ни одного заранее определенного тега с фиксированным значением, и, в отличие от HTML, он никак не определяет способ отображения документа.

Итак, этапы создания новой технологии:

- 1998 год – декларация основных концепций языка XML,
- 1999 год - выработка рекомендаций (стандартов) консорциумом W3C на новую технологию (самого языка XML и его сопутствующих компонент: XSL (Extensible Stylesheet Language) и XLL (Extensible Link Language)); создание объектной модели DOM (Document Object Model) языка XML; начало создания средств для применения конечным пользователем этой технологии (синтаксический анализатор (parser) в браузере IE 5.0, X Project (библиотека Java-классов компании Sun) и др.).

Возможность применения новой XML-технологии для, как казалось ранее, экзотичных сфер (например, химия, ботаника, математика, лингвистика) в невиданном масштабе породила процесс разработки специализированных языков и во многих случаях их реализаций, не дожидаясь окончательной стандартизации этой технологии и получения приемлемых программных средств для ее эффективного применения. В этой связи необходимо отметить язык MathML [5] для описания математических формул и его реализацию в специализированном браузере/редакторе Amaya. В приложении А приведены примеры изображения математических формул на экране монитора и описания их на языке MathML, включенном в стандартный HTML-документ.

Данное описание не является учебным пособием по языку XML. Автором была предпринята попытка в какой-то мере систематизировать разрозненные материалы из многочисленных Интернет-источников и дать возможному читателю на примерах получить целостное и конкретное представление о новой WWW-технологии, возможно ожидающей каждого WEB-конструктора уже завтра. Язык HTML сыграл свою выдающуюся роль в становлении WWW-гиперпространства, но новые времена требуют новых решений.

Указанные в перечне литературы гиперссылки действительны на январь 2000 года.

2.Основные требования к структуре XML-документа

Язык XML, как и все подмножества SGML, включая HTML, содержит единственное фундаментальное понятие: теги – ключевые слова, обрамленные угловыми скобками. В XML задание значений тегов предоставляется пользователю – разработчику собственного языка.

В общем случае XML-документы должны удовлетворять следующим требованиям.

- Первой строкой XML-документа является заголовок или *пролог* (prolog), в котором указывается название языка, номер его версии, тип кодировки

текста и другая дополнительная информация. В настоящее время определены спецификации для XML-версии 1.0.

- Открывающие и закрывающие теги образуют *контейнеры* с произвольной глубиной вложения. При создании тегов контролируется регистр символов.
- В определении текстов тегов могут задаваться их *атрибуты*, и задаваемые их значения должны быть заключены в кавычки.
- Вся информация, располагающаяся в контейнерах, включая символы форматирования , рассматриваются в XML как данные.

Кроме того, в XML-документе допускается применение конструкций.

Комментарии

Комментарии задаются специальным тегом:

<! ----- this is comment >

Пустые теги

Пример задания пустого тега: </empty>

Инструкции

В XML-документе могут задаваться инструкции (директивы) для приложения, выполняющего его интерпретацию. Эти директивы задаются в виде

<? target instructions ?>

Специальные данные CDATA

В XML-документе могут задаваться области с ограничением действий синтаксических анализаторов, и в основном они предназначены для программ интерпретаторов. Задаются они оператором <![CDATA] . . .]]>.

Пример:

```
<item><![CDATA[Diagram:
   frobmorten <---------- fuznaten
      |         <3>      ^
      |<1>               |  <1> = fozzle
      V                  |  <2> = framboze
    staten----------------+  <3> = frenzle
             <2>
]]></item>
```

Если XML-документ соответствует вышеприведенным правилам, то он называется *формально-правильным* **(well formed)** .

Пример формально-правильного XML-документа:

```
<?xml version="1.0" encoding=”ascii-us” ?>
<petfolio>
   <pet name="ella” surname=”dibella">
```

```
        <breed>Domestic Shorthair</breed>
        <color>black/tan tiger</color>
        <age units="months">3</age>
        <weight units="lbs" uni="qwerty">1.75</weight>
        <description>Ella is a smart, poised creature full of courage and vinegar.
        </description>
    </pet>
    <pet name="borders" surname="mcgee">
        <breed>Domestic Shorthair</breed>
        <color>black/white tuxedo</color>
        <age units="weeks">10</age>
        <weight units="lbs">1.5</weight>
        <description>Borders is a playful, submissive jester. </description>
    </pet>
</petfolio>
```

Заголовком (прологом) XML-документа является первая строка:

<?xml version="1.0" encoding="ascii-us" ?>

Строка

<pet name="ella" surname="dibella">

содержит тег **<pet>** с заданием в кавычках значений его атрибутам **name** и **surname**.

Для применения кириллицы в XML-документах надо задавать соответствующий тип кодировки:

<?xml version="1.0" encoding= "windows-1251" ?>

Приведенный выше XML-документ является формально-правильным, и для проверки этого факта можно использовать любой синтаксический XML-анализатор. Самый простой вариант – использовать браузер IE 5.0 и в качестве параметра ему (операцией File/Open) задать текстовый файл с содержимым этого XML-документа.

3. Совместное использование XML- и HTML-документов

На данный момент пока не существует полных стандартных решений для интерпретации пользовательских XML-разработок. Однако возможно применение некоторых стандартных механизмов для облегчения задачи создания собственных интерпретаторов. Так, браузер IE 5.0 можно применять не только для синтаксического анализа XML-документа, но и использовать его средства для интерпретации XML-кода. Подробнее о применении этого механизма будет идти речь в следующих разделах данного описания, в частности при обсуждении объектной модели XML-документа и ее реализациях. Сейчас будет рассмотрен вопрос о включении XML-деклараций в HTML-документы.

Существует несколько таких вариантов.

Первый вариант: встраивание XML-текста непосредственно в HTML-документ и обработка его средствами HTML:

```
<HTML>
<BODY>
<XML ID="resortXML">
   <resorts>
      <first> Calinda</first>
      <sec>  Second </sec>
      <three> Na Balam </three>
</resorts>
</XML>
<table DATASRC="#resortXML" BORDER=1>
   <tr>
       <TD><SPAN DATAFLD="first"></SPAN></TD>
       <TD><SPAN DATAFLD="sec"></SPAN></TD>
       <TD><SPAN DATAFLD="three"></SPAN></TD>
</table>
</BODY>
</HTML>
```

Второй вариант: указание ссылки в виде URL на XML-документ:

```
<html>
<body>
<xml id=xmlDoc src="university.xml"></xml>
<table datasrc="#xmlDoc" border=1>
  <thead><th>NAME</th>
         <th>LOCATION</th>
  </thead>
  <tr>   <td><span datafld="name"></span></td>
         <td><span datafld="location"></span></td>
</table>
</body>
</html>
```

Файл **university.xml,** используемый в примере, приведен в приложении Б.

4. Грамматика языка XML

Как показано в предыдущем разделе, для корректной обработки XML-документов необходимым условием является требование, чтобы этот документ был формально-правильным . Но для новых, создаваемых на базе XML, языков для массового применения важным условием является также наличие их формального описания (грамматики). Для XML существуют два подхода к заданию грамматики: **DTD**-определения и схемы данных **Semantic Schema**. Можно отметить в этой связи, что оба подхода вызывают много нареканий: DTD – за сложность для рядового потребителя, Schema – за отсутствие пока определенности в стандартизации. XML-документы, прошедшие грамматический контроль, называются *валидными* (valid).

4.1. Задание грамматики в определениях Document Type Definition, DTD

Языки, создаваемые средствами XML, в свою очередь становятся также подмножеством SGML, и для описания их грамматики применимы те же средства, что и для этих метаязыков, а именно: Document Type Definition (DTD) [1]. DTD-определения могут включаться непосредственно в заголовок XML-документа, могут подгружаться как URL-документы, либо может использоваться комбинация этих вариантов.

Ниже будет приведено несколько примеров.

Пример1. Включение DTD в XML-документе.

```
<?xml version='1.0' encoding='windows-1251'?>
<!DOCTYPE slideshow [
<!ELEMENT slideshow (slide+)>
<!ELEMENT slide (title, item*)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT item (#PCDATA | item)* >
]>

<slideshow>
   <slide>
       <title>Wake up to WonderWidgets!</title>
  </slide>
  <slide>
       <title>Overview</title>
       <item>Why <em>WonderWidgets</em> are great</item>
       <item/>
       <item>Who <em>buys</em> WonderWidgets</item>
  </slide>
</slideshow>
```

Пример 2. Использование ссылки на DTD-определение.

```
<?xml version='1.0'?>
<!DOCTYPE slideshow SYSTEM "slide.dtd">
<slideshow>
<slide>
    <title>Wake up to WonderWidgets!</title>
  </slide>
<slide>
    <title>Overview</title>
    <item>Why <em>WonderWidgets</em> are great</item>
    <item/>
    <item>Who <em>buys</em> WonderWidgets</item>
  </slide>
</slideshow>
```

Файл slide.dtd:

```
<?xml version='1.0'?>
<!ELEMENT slideshow (slide+)>
<!ELEMENT slide (title, item*)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT item (#PCDATA | item)* >
```

4.2. Использование схемы данных (Semantic Schema) для задания грамматики

Схемы данных являются альтернативным способом задания грамматики XML-документов. Они позволяют описывать правила для XML-документа средствами самого же XML и по сравнению с DTD обеспечивают более понятный способ описания грамматики языка. Надо отметить также, что схемы данных не могут полностью заменить DTD, поскольку декларации DTD являются стандартом для всех языков, совместимых с SGML, они реализованы во всех стандартных синтаксических анализаторах. К тому же рекомендации консорциума WWW по стандартизации описаний грамматики языка XML методом схемы данных не содержат окончательных их спецификаций. В приведенном ниже примере используется схема данных, реализованная компанией Microsoft в браузере IE 5.0:

```
<?xml version="1.0" ?>
 <class xmlns="x-schema:Schema.xml">
   <student studentID="13429">
    <name>James Smith</name>
    <GPA>3.8</GPA>
   </student>
 </class>
```

Файл **Schema.xml**:

```
<Schema xmlns="urn:schemas-microsoft-com:xml-data"
 xmlns:dt="urn:schemas-microsoft-com:datatypes">
 <AttributeType name="studentID" dt:type="string" required='yes'/>
 <ElementType name='name' content='textOnly'/>
 <ElementType name='GPA' content='textOnly' dt:type='float'/>
</Schema>
```

5. Объектная модель (DOM, Document Object Model) XML-документа

Одним из самых мощных средств доступа к содержимому XML-документов является Document Object Model (DOM). DOM XML-документа является представлением его внутренней структуры в виде определенных объектов,

организованных в древовидную структуру. Объектное представление структуры документа давно используется при обработке HTML-документов средствами объектно-ориентированного языка JavaScript. Объектно-ориентированный подход должен обеспечивать средства для создания самого объекта и набор методов (программ) для доступа к его свойствам (данным). В настоящее время существует две реализации объектной модели XML- документа – модель DOM Microsoft и Java Sun DOM.

5.1. Объектная модель Microsoft

Internet Explorer 5.0 (IE 5.0) является первым браузером, в котором реализована объектная модель XML-документа. Это означает, что в IE 5.0 обеспечен интерфейс доступа к содержимому документа из языка сценариев JScript (вариант JavaScript от Microsoft). Поэтому можно рассматривать объектную модель как набор объектов, их методов и событий, доступных из языка сценариев внутри HTML-страницы.

Наиболее значимыми для обработки XML-документа являются объекты XMLDOMDocument, XMLDOMNode, XMLDOMNodeList, содержащие ряд полезных методов:

- nodeName(), nodeType() – запрос информации о текущем элементе дерева;
- text(), xml(), nodeValue() – запрос содержимого узла;
- childNodes(), lastChild(), firstChild() - работа со списком дочерних элементов;
- appendChild(newChild), replaceChild(newChild, oldChild), removeChild(oldChild) - добавление и модификация элементов в объектной модели.

Ниже приведен пример HTML-странички, содержащей обращения к XML-документу (файл article.xml) и программе на JavaScript (файл convert.js) для обработки сформированного браузером объекта с заданным идентификатором xmlID.

```
<html>
<head>
 <title> Convertor </title>
</head>
<body>
  <xml id="xmlID" src="article.xml"></xml>
  <SCRIPT LANGUAGE="JavaScript1.2" SRC="convert.js">
  </SCRIPT>
</body>
</html>
```

В приложении В приведены фрагменты из реальной программы конвертации текста публикации, описанной на разработанном автором с использованием технологии языке XArt , в HTML-представление с динамической выдачей на экран монитора. При этом в программе на языке

JavaScript применены методы для работы с двумя типами объектов – XML- и HTML-документов.

5.2. Объектная модель Java SUN

Объектная модель для XML-документов реализована компанией SUN в пакете **X Project**, содержащем библиотеку Java-классов **xml.jar**. В этом пакете представлены два подхода к построению XML-объектов.

5.2.1. Объектная модель API SAX (Simple API for XML)

Метод SAX является событийно-ориентированным подходом для построения XML-объекта. Основная идея этого подхода: синтаксический анализатор при обнаружении очередного события при обработке XML-документа (синтаксической конструкции (например, тега) или находит ошибку) вызывает программу-обработчик (event hanldler) этого события с соответствующими параметрами. Написание таких программ (например, startDocument(), endDocument(), startElement(), endElement()) выполняется самим пользователем. Программы SAX-метода входят в библиотеку Java-классов **org.xml.sax**.

5.2.2. Объектная модель DOM в проекте Sun X Project

Другой подход к реализации объектной модели разработан компанией Sun в так называемом проекте **X Project** - программах из библиотеки классов **org.w3c.dom**. Идея этого подхода аналогична модели Microsoft DOM:

- синтаксический анализ входного XML-документа и создание древовидной структуры объектов и их свойств (данных);
- обеспечение методов для доступа к узлам (подчиненным объектам) и их свойствам.

Приведем несколько примеров работы с XML-документом, используя эту методику:

а) объявление библиотеки классов:

```
import com.sun.xml.*;
import org.xml.*;
import java.awt.*;
import java.io.*;
import org.w3c.dom.*;
import org.xml.sax.*;
```

б) формирование XML-объекта из входного файла с XML-документом:

```
InputSource    input  = Resolver.createInputSource (new File (argv [0]));
```

```
XmlDocument doc    = XmlDocument.createXmlDocument (input, false);
doc.getDocumentElement ().normalize ();
```

в) пример обращения к методам объекта:

```
int nodes            = doc.getLength();
ElementNode root = (ElementNode) doc.getDocumentElement ();
String name          = root.getLocalName ();
NamedNodeMap attributes = root.getAttributes ();
 int length = attributes.getLength ();
```

В приложении Г приведены примеры XML-документа с описанием таблицы химических элементов Д.И. Менделеева и фрагменты программы на языке Java.

5.2.3. Динамическое создание XML-объектов

Одной из интересных областей применения XML-технологии является использование XML в качестве промежуточного языка при обмене данными в распределенных системах, в частности в WWW. Становится уже почти классической следующая схема обмена сложными структурированными данными от одного приложения к другому, в частности между WEB-клиентами и серверами.

Серверная программа (например, Java-сервлет), используя компоненты JDBC, выполняет следующее:

- читает затребованную клиентом информацию из базы данных;
- формирует заготовку XML-объекта;
- заполняет его структуру данными;
- формирует текстовый объект типа String из XML-объекта;
- передает его клиенту.

Программа клиента на языке JavaScript, используя объектную модель DOM браузера, обрабатывает полученный XML-документ (см. раздел 5.1).

Для выполнения этой задачи программе сервера необходимо решить проблему динамического формирования и преобразования XML-объекта.

Предлагается авторская схема этого решения в рамках средств **X Project**:

```
import java.io.*;
import com.sun.xml.tree.*;
import org.w3c.dom.*;
        .....
        // создать заготовку (пустого)  XML-объекта
        XmlDocumentdoc = new XmlDocument ();
        .....
        // Используя методы XML-объекта  createElement(), appendChild (),
        // createTextNode(), setDoctype (), сформировать структуру XML-объекта
        .....
```

```
// Используя классы ввода-вывода, преобразовать XML-объект в
// текстовое представление
CharArrayWriter chw = new CharArrayWriter();
doc.write (chw);
String str = chw.toString();
.....
```

Примечание. Представляет интерес также динамическое создание средствами X Project XML-объекта в Java-программе из его текстового String-представления.

Пример. Строка String str содержит текст XML-документа.

1 вариант.

```
InputSource inputSource = new InputSource();
inputSource.setCharacterStream( new StringReader(str ) );
XmlDocument doc = XmlDocument.createXmlDocument (inputSource, false);
```

2 вариант.

```
InputSource inputSource = new InputSource ( new StringReader (aString));
XmlDocument doc = XmlDocument.createXmlDocument (inputSource, false);
```

6. Стилевые таблицы (XSL, Extensible Stylesheet Language)

Конечной целью обработки любого WEB-документа является его изображение на экране. XML по своей сути, в отличие от HTML, не содержит средств описания того, в каком формате и виде его содержимое будет выводиться на экран. В предыдущих разделах приводились примеры форматирования и индикации XML-документов с применением специальных средств, в частности языка JavaScript. Однако для этих целей предпочтительнее использовать специально предназначенные для этого средства - стилевые таблицы XSL (Extensible Stylesheet Language). Идея языка XSL во многом аналогична каскадным стилевым таблицам (CSS, Cascading Style Sheets), уже давно используемым разработчиками для оформления Web-страниц. CSS могут применяться и для описания XML-данных в ограниченном виде, поскольку использование статических ссылок на форматируемые объекты XML-документа в общем случае невозможно. На сегодняшний день единственным браузером, поддерживающим XSL, является IE 5.0. Существуют также различного рода конвертеры, преобразующие XML-данные в другие, более удобные для индикации представления, в частности в HTML .

Ниже на примерах будет продемонстрировано использование XSL для описания преобразований XML-документов.

6.1. Использование CSS таблиц

Очень простой пример:

файл **greeting.xml:**

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/css" href="greeting.css"?>
<GREETING>
Hello XML!
</GREETING>
```

файл **greeting.css:**

```
GREETING {display: block; font-size: 48pt; font-weight: bold;}
```

При обработке XML-файла браузером IE 5.0 получим на экране сообщение **Hello XML!** в соответствии с определением в CSS-таблице greeting.css

6.2. Стилевые таблицы XSL

Синтаксический анализатор при разборе XML-документа каждому элементу, найденному в XML- дереве, ставит в соответствие набор тэгов, определяющих форматирование этого элемента, в XSL-таблице.

Любое преобразование XML-документа с применением определений XSL можно рассматривать как его конвертирование в другое XML-представление. В этом ключе нужно рассматривать и XML <-> HTML - преобразование, поскольку HTML и XML являются, как уже выше отмечалось, подмножествами SGML.

В настоящее время практически используются две XSL-схемы:

- динамическое форматирование XML-документа, применяемое в браузерах для выдачи на экран его содержимого. Схема задается как: xmlns:xsl=http://www.w3.org/TR/WD-xsl;
- схема трансформирования XML-документа, используемая в конвертерах. Например, в преобразование XML-документа в HTML-представление. Задание схемы: `xmlns:xsl=`http://www.w3.org/1999/XSL/Transform .

Ниже будет рассмотрен пример с упрощенной объектной моделью XSL-документа, реализованной в конвертере **msxsl** браузера IE 5.0.

Пример, аналогичный примеру из раздела 6.1:

файл **greeting.xml:**

```
<?xml version="1.0" standalone="yes"?>
<?xml-stylesheet type="text/xsl" href="greeting.xsl"?>
<GREETING>
Hello XML!
</GREETING>
```

файл **greeting.xsl:**

```
<?xml version="1.0" ?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl">
 <xsl:template match="/">
  <HTML>
   <BODY>
    <xsl:for-each select="GREETING">
     <H1>
      <xsl:value-of/>
     </H1>
    </xsl:for-each>
   </BODY>
  </HTML>
 </xsl:template>
</xsl:stylesheet>
```

6.3. XML-конвертеры

Кроме использования XSL для динамического форматирования XML-документов, в браузерах применяются специальные программы-конвертеры, выполняющие преобразования типа "файл-файл". Примером такого конвертера могут служить программы Java-пакета **XT** (библиотека классов **xt.jar** и **xp.jar**). Параметрами этой программы являются два исходных XML- и XSL-файла, а также результирующий HTML-файл. В данном конвертере реализована объектная XSL-модель **`xmlns.`**

Пример обращения к программе конвертера:

- добавить к переменной окружения CLASSPATH доступ к файлам xt.jar и xp.jar, например:
  set CLASSPATH=%CLASSPATH%;c:\xml\xt.jar;c:\e \xp.jar
- обращение к программе конвертера:
  java -Dcom.jclark.xsl.sax.parser=com.jclark.xml.sax.CommentDriver
  com.jclark.xsl.sax.Driver test.xml test.xsl test.html

Результатом конвертирования XML-файла **test.xml** с заданием схемы форматирования в XSL-файле **test.xsl** будет HTML-файл **test.html**.

Тексты этих файлов приведены в приложении Д.

7. Язык гиперссылок - XLL, eXtensible Linking Language

XLL-протоколы содержат два типа спецификаций для управления гиперссылками между XML-документами: **XLink** и **XPointer**. XLink, XML Linking Language [7], определяет задание ссылок между XML-документами. XPointer, XML Pointer Language [8], задает адресацию частей XML-документа. В данной работе приводится минимальное описание применения средств XLL, поскольку, кроме формализованного описания включения их в XML-

документ, в данный момент нет ни одной даже демонстрационной ее реализации. (For now, we can only guess what applications might do with extended links and what sort of user interfaces they might provide [7].)

7.1. Задание гиперссылок: XLink

Язык XML имеет более развитую и гибкую, в отличие от HTML, систему гиперссылок к внешним ресурсам. Почти каждый тег XML может включать гиперссылки, и они тогда называются элементами ссылок (*linking elements).* Эти элементы идентифицируются типом ссылки **xlink:type** с двумя значениями атрибутов:

- simple
- extended

7.1.1. Простые (simple) ссылки

Простые ссылки (типа simple) во многом выполняются аналогично гиперссылкам в HTML-документе.

Каждый элемент ссылки содержит атрибут **xlink:href**, значением которого является URL ссылочного ресурса. И, последнее, атрибут **xmlns:xlink** применяется для определения версии XLink.

Пример:

```
<FOOTNOTE xmlns:xlink="http://www.w3.org/XML/XLink/0.9"
      xlink:type="simple"
      xlink:href="footnote7.xml">
  abcdef
</FOOTNOTE>
<COMPOSER xmlns:xlink=http://www.w3.org/XML/XLink/0.9
      xlink:type="simple"
      xlink:href="http://www.users.interport.net/~beand/">
  Beth Anderson
</COMPOSER>
<IMAGE xmlns:xlink=http://www.w3.org/XML/XLink/0.9
    xlink:type="simple" xlink:href="logo.gif"/>
```

Декларация удаленных ресурсов

Элементы ссылок могут содержать *дополнительно* атрибуты **xlink:role** и **xlink:title** для произвольного описания удаленных ресурсов, к которым относятся эти ссылки.

Пример:

```
<AUTHOR
 xmlns:xlink="http://www.w3.org/XML/XLink/0.9"
 xlink:href="http://www.macfaq.com/personal.html"
 xlink:title="Elliotte Rusty Harold's personal home page"
 xlink:role="information about the author of this book">
   Elliotte Rusty Harold
</AUTHOR>
```

Активизация ссылок

Элементы ссылок могут также содержать атрибуты для указания типа ассоциации удаленного ресурса по отношению к текущей странице:

- `xlink:show`
- `xlink:actuate`

Эти атрибуты являются необязательными, и реализация их зависит от выполняемого приложения.

Атрибут **`xlink:show`** дает указание на то, как содержимое ссылки будет отражаться при ее активизации (например, в этом же или в новом окне). Это могут быть значения: **`replace`**, **`new`** или **`parsed.`**

Пример:

```
<COMPOSER xmlns:xlink="http://www.w3.org/XML/XLink/0.9"
      xlink:type="simple"
      xlink:show="replace"
      xlink:href="http://www.users.interport.net/~beand/">
  Beth Anderson
</COMPOSER>
```

Если же значение атрибута - **parsed**, содержимое активизированной ссылки вставляется в существующий документ.

Пример:

```
<?xml version="1.0" standalone="no"?>
<FAMILY xmlns:xlink="http://www.w3.org/XML/XLink/0.9">
 <HUSBAND xlink:type="simple" xlink:show="parsed"
       xlink:href="ThomasCorwinAnderson.xml" />
 <WIFE   xlink:type="simple" xlink:show="parsed"
       xlink:href="LeAnahDeMintEnglish.xml" />
 <CHILD  xlink:type="simple" xlink:show="parsed"
       xlink:href="JohnJayAnderson.xml" />
</FAMILY>
```

Атрибут **xlink:actuate** задает режим ручного или автоматического выполнения. Его значениями могут быть: **user** или **auto.**

Пример:

```
<IMAGE xmlns:xlink="http://www.w3.org/XML/XLink/0.9"
    xlink:type="simple"
    xlink:href="logo.gif"
    xlink:actuate="auto"/>
```

7.1.2. Расширенные (extended) ссылки

Расширенные ссылки предназначены для выполнения многонаправленных связей между XML-документами. Они содержат ячейки (локаторы) и описание их связей. В настоящее время это наименее проработанная часть XML-технологии.

Пример:

```
<?xml version="1.0"?>
<WEBSITE xmlns:xlink="http://www.w3.org/XML/XLink/0.9"
      xlink:type="extended">
 <HOMESITE xlink:type="locator"
        xlink:href="http://metalab.unc.edu/javafaq/">
  North Carolina
 </HOMESITE>
 <MIRROR xlink:type="locator"
       xlink:href="http://sunsite.kth.se/javafaq">
  Sweden
 </MIRROR>
 <MIRROR xlink:type="locator"
       xlink:href="http://sunsite.cnlab-switch.ch/javafaq/">
  Switzerland
 </MIRROR>
</WEBSITE>
```

7.2. XML-указатели ссылок - XPointer, XML Pointer Language

Средства XPointer определяют схему адресации между отдельными частями XML-документов. Отдаленным HTML-аналогом его является схема.

Описание точки входа (якоря):

```
<H2><A NAME="xpointers">XPointers</A></H2>
```

Ссылка на этот фрагмент:

```
<A HREF="#xpointers">XPointers</A>
```

XLinks допускает более развитую систему связей между XML-документами и его частями: абсолютную (как для HTML), относительную к определенному якорю, по содержимому элемента документа, многонаправленные ссылки.

Полный формат XLinks содержит ссылку в виде URL на определенный ресурс и указатель XPointer в виде URL #XPointer.

Примеры XPointer указателей:

```
xptr(id("ebnf"))
xptr(descendant::language[position()=2])
ebnf
xptr(id("ebnf"))xptr(id("EBNF"))
```

Каждый из этих примеров выбирает определенный элемент из XML-документа:

первый – отыскивает элемент с заданным ID= ”ebnf”;

второй – выбирает первый элемент < language>, который является вторым (position()=2) вложенным (child) элементом по отношению к родительскому (parent);

третий – является укороченной формой задания ID = "ebnf";

четвертый - также указывает на элемент с ID = ”ebnf”.

Полный формат таких ссылок, включающий указатели XPointer, в предположении, что они ссылаются на один и тот же XML-документ в файле www.w3.org/xml/tr.xml:

```
http://www.w3.org/xml/tr.xml#xptr(id("ebnf"))
http://www.w3.org//xml/tr.xml#xptr
                (descendant::language[position()=2])
http://www.w3.org//xml/tr.xml#ebnf
http://www.w3.org//xml/tr.#xptr(id("ebnf"))xptr(id("EBNF"))
```

Задание XPointer-отметок выполняется функциями **id()**, **root()** или **child()**. Например, файл http://www.theharolds.com/genealog.xml содержит фрагмент:

```
<PERSON ID="p12" FATHER="p2" MOTHER="p1">
  <NAME>Honore Bellau</NAME>
</PERSON>
```

Тогда ссылка http://www.theharolds.com/genealogy.xml#xptr(id("p12")) адресует к содержимому элемента <PERSON>, а именно ”Honore Bellau”.

Эта ссылка допускает также упрощенный формат:

```
http://www.theharolds.com/genealogy.xml#p12
```

8. Заключение

Один из признаков успеха новой XML-технологии - появление, не дожидаясь окончательного утверждения на нее стандартов, многочисленных приложений, уже получивших известность, таких как: MathML (Mathematical Markup Language), CML(Chemical Markup Language), VML(Vector Markup Language), XFDL (Extensible Forms Description Language) и многих других. В консорциуме W3C рассматривается рабочий вариант стандарта XML-QL (или XQL), который может составить серьезную конкуренцию SQL. Находятся в стадии реализации коммерческие проекты, например [9], с использованием XML в качестве промежуточного языка в распределенных системах для доступа к базам данных.

XML – это в первую очередь Интернет-технология, которая в сознании массового потребителя вполне справедливо прочно ассоциируется с WWW,

где основными поставщиками главных WEB-инструментов (браузеров) являются компании Microsoft и заметно утратившая былое преимущество Netscape. Следует отметить, что Microsoft является активным участником консорциума W3C и серьезно поддерживает (но еще недостаточно!) XML-технологию, подтверждением чего является выпуск нового поколения браузера - Internet Explorer 5.0.

Можно отметить также, что о поддержке и применении XML, кроме вышеупоминавшихся компаний-производителей Интернет-продукции (Sun, Microsoft, Netscape), объявили такие известные компании, как Software AG, Lotus Development, IBM, Data Channel, Oracle и многие другие (еженедельник PC Week, номера 40-47 за 1999 год).

Автор выражает благодарность профессору В.П. Ширикову за полезные советы при обсуждении и подготовке данной публикации.

Литература

1. Overview of SGML Resources. http://www.w3.org/MarkUp/SGML/.
2. Jon Bosak.XML, Java, and the future of the Web.
3. Jon Bosak, Tim Bray. XML and the Second-Generation Web. http://www.sciam.com/1999/0599issue/0599bosak.html.
4. Дм. Кирсанов. XML против HTML.http://www.inter.net.ru/1/19.html.
5. http://www.w3.org/Math/.
6. Печерский А. Язык XML - практическое введение. http://www.citforum.ru/internet/xml/index.shtml.
7. XLinks, Chapter 16 of the XML Bible, Updated Version. http://www.metalab.unc.edu/xml/books/bible/updates/16.html.
8. XPointers, Chapter 17 of the XML Bible, Updated Version. http://www.metalab.unc.edu/xml/books/bible/updates/17.html.
9. INCO-Copernicus Project 977041 Mall2000 – Mall for on-line Buisness beyond the Year 2000. http://www-it.fmi.uni-sofia.bg/mall2000/.

Приложение А. Язык MathML

Пример 1. Текст программы на языке MathML.

```
<html>
<head>
<title>Test for MathML</title>
</head>
<body>>
<math>
 <m:mrow>
  <m:msub>
   <m:mi>&delta;</m:mi>
   <m:mi>&alpha;</m:mi>
  </m:msub>
  <m:msup>
```

```
      <m:mi>F</m:mi>
      <m:mi>&alpha;&beta;</m:mi>
    </m:msup>
    <m:mi></m:mi>
    <m:mo>=</m:mo>
    <m:mi></m:mi>
    <m:mfrac>
     <m:mrow>
      <m:mn>4</m:mn>
      <m:mi>&pi;</m:mi>
     </m:mrow>
     <m:mi>c</m:mi>
    </m:mfrac>
    <m:mi></m:mi>
    <m:msup>
     <m:mi>J</m:mi>
     <m:mrow>
      <m:mi>&beta;</m:mi>
      <m:mo></m:mo>
     </m:mrow>
    </m:msup>
  </m:mrow>
</math>
</body>
</html>
```

Пример 2. Изображение формулы, описанной на языке MathML в примере 1:

$$\delta_\alpha F^{\alpha\beta} = \frac{4\pi}{c} J^\beta$$

Приложение Б. Пример формально-правильного XML-документа

```
<?xml version="1.0" ?>
<universities>
  <university>
    <name>UCSB</name>
    <location>Santa Barbara, CA</location>
  </university>
  <university>
    <name>University of Texas at Arlington</name>
    <location>Arlington, TX</location>
  </university>
  <university>
    <name>Baylor</name>
    <location>Waco, TX</location>
  </university>
</universities>
```

HTML-документ с вызовом XML-документа и JavaScript-конвертера:

```
<html>
<head>
 <title> Convertor </title>
</head>
<body>
  <xml id="xmlID" src="article.xml"></xml>
  <SCRIPT LANGUAGE="JavaScript1.2" SRC="convert.js">
  </SCRIPT>
</body>
</html>
```

Файл **article.xml** с разметкой текста публикации:

```
<?xml version="1.0"?>
<article>
<head>
   <title>XML, Java, and the future of the Web</title>
   <author> Jon Bosak </author>
   <org>Sun Microsystems</org>
<abstract>
     The extraordinary growth of the World Wide Web has been
     fueled by the ability it givesauthors to easily and cheaply
     distribute electronic documents to an international audience.
</abstract>
</head>
<body>
  <chapter> Introduction</chapter>
  <para>
     Most documents on the Web are stored and transmitted in HTML. HTML is a simple language
well suited for hypertext, multimedia, and  the display of small and reasonably simple documents.
HTML is based on SGML <bold>(Standard Generalized Markup Language, ISO 8879)</bold>, a
standard system for defining and using document formats.
   </para>
</body>
</article>
```

Фрагмент программы на языке JavaScript для обработки XML- объекта:

```
. . . . . .
nnodes = xmlID.childNodes.length
for(m=1; m<nnodes; m++)
  {
  root=xmlID.childNodes(m)

  document.write("<br> Node type:   " + root.nodeType);
  document.write("<br> Node name:   " + root.nodeName);

  }
  lenNode=root.childNodes.length;
```

```
   //---------- cycle for Root node -------------------
   for(i=0; i < lenNode; i++) {
        nodeName=root.childNodes(i).nodeName;
        nodeType=root.childNodes(i).nodeType;
        if(nodeType == 1)
              lenAttrib = root.childNodes(i).attributes.length
        else  lenAttrib = 0


      //-------- cycle for attributes for each nodes-----------------
      for(j=0; j<lenAttrib; j++) {
      attName=root.childNodes(i).attributes(j).nodeName
      attValue=root.childNodes(i).attributes.item(j).text

attVal=root.childNodes(i).attributes.getNamedItem(attribName).text

       checkTag(nodeName)
     }
}
```

Приложение Г. Объектная модель **Sun Java DOM**

Файл **table.xml** с XML-документом:

```
<?xml version="1.0" encoding='windows-1251'?>
<table>
   <element H="1">
        <name>Водород</name>
        <group>7</group>
        <period>1</period>
        <mass>1,0079</mass>
        <electron> 1 </electron>
   </element>
   <element He="2">
        <name>Гелий</name>
        <group>8</group>
        <period>1</period>
        <mass>4,0026</mass>
        <electron> 2 </electron>
   </element>
   <element Li="3">
        <name>Литий</name>
        <group>1</group>
        <period>2</period>
        <mass>6,941</mass>
        <electron> 1 2 </electron>
   </element>
</table>
```

Фрагмент Java-программы:

```
import org.w3c.dom.*;
import com.sun.xml.*;
```

```
import org.xml.sax.*;
import java.awt.*;
import java.io.*;

public class main
   public static void main (String argv []){
         InputSource      input;
         XmlDocument    doc;
         try {
            input = Resolver.createInputSource (new File (argv [0]));

            // turn it into an in-memory object
            doc = XmlDocument.createXmlDocument (input, false);
            doc.getDocumentElement ().normalize ();
            doc.write (System.out);
//-------------------------------------------------------
      int nodes=doc.getLength();
      System.out.println("Root number childes: " + nodes);

            ElementNode root = (ElementNode) doc.getDocumentElement ();
      String name = root.getLocalName ();
      System.out.println("Root name: " + name);

            NamedNodeMap         attributes = root.getAttributes ();
            int length = attributes.getLength ();
      System.out.println("Root attributes: " + length);

      childLevel chl = new childLevel("",0);
      chl.nextLevel(root,name);

      Frame ft = new setFrame();
         } catch (SAXParseException err) {
            System.out.println ("** Parsing error" );
         } catch (SAXException e) {
         } catch (Throwable t) {
            t.printStackTrace ();
         }  System.exit (0);
   }
}
```

Приложение Д. Конвертирование (XML + XSL) -> HTML

Файл **test.xml**:

```
<?xml version="1.0"?>
<PERIODIC_TABLE>
 <ATOM STATE="GAS">
  <NAME>Hydrogen</NAME>
  <SYMBOL>H</SYMBOL>
  <ATOMIC_NUMBER>1</ATOMIC_NUMBER>
  <ATOMIC_WEIGHT>1.00794</ATOMIC_WEIGHT>
  <BOILING_POINT UNITS="Kelvin">20.28</BOILING_POINT>
  <MELTING_POINT UNITS="Kelvin">13.81</MELTING_POINT>
  <DENSITY UNITS="grams/cubic centimeter"><!-- At 300K -->
   0.0899
```

```
  </DENSITY>
 </ATOM>
 <ATOM STATE="GAS">
  <NAME>Helium</NAME>
  <SYMBOL>He</SYMBOL>
  <ATOMIC_NUMBER>2</ATOMIC_NUMBER>
  <ATOMIC_WEIGHT>4.0026</ATOMIC_WEIGHT>
  <BOILING_POINT UNITS="Kelvin">4.216</BOILING_POINT>
  <MELTING_POINT UNITS="Kelvin">0.95</MELTING_POINT>
  <DENSITY UNITS="grams/cubic centimeter"><!-- At 300K -->
   0.1785
  </DENSITY>
 </ATOM>
 </PERIODIC_TABLE>
```

файл **test.xsl**:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
   <html>
    <head>
     <title>Atomic Number vs. Atomic Weight</title>
    </head>
    <body>
     <xsl:apply-templates select="PERIODIC_TABLE"/>
    </body>
   </html>
  </xsl:template>

  <xsl:template match="PERIODIC_TABLE">
   <h1>Atomic Number vs. Atomic Weight</h1>
   <table>
    <th>Element</th>
    <th>Atomic Number</th>
    <th>Atomic Weight</th>
     <xsl:apply-templates select="ATOM"/>
   </table>
  </xsl:template>

  <xsl:template match="ATOM">
   <tr>
    <td><xsl:value-of select="NAME"/></td>
    <td><xsl:value-of select="ATOMIC_NUMBER"/></td>
    <td><xsl:value-of select="ATOMIC_WEIGHT"/></td>
   </tr>
  </xsl:template>
</xsl:stylesheet>
```

Результирующий файл **test.html**:

```
<html>
<head>
```

```
<title>Atomic Number vs. Atomic Weight</title>
</head>
<body>
<h1>Atomic Number vs. Atomic Weight</h1>
<table>
<th>Element</th><th>Atomic Number</th><th>Atomic Weight</th>
<tr>
<td>Hydrogen</td><td>1</td><td>1.00794</td>
</tr>
<tr>
<td>Helium</td><td>2</td><td>4.0026</td>
</tr>
</table>
</body>
</html>
```

Результат выдачи на экран HTML-документа:

Atomic Number vs. Atomic Weight

Element	Atomic Number	Atomic Weight
Hydrogen	1	1.00794
Helium	2	4.0026

Рукопись поступила в издательский отдел
6 марта 2000 года.

Галактионов В.В.
Р10-2000-44

Расширяемый язык разметки XML
(eXtensible Mark-up Language): промышленный стандарт,
определяющий архитектуру программных средств
Интернет следующего поколения

Прошедший 1999 год стал периодом становления новой Интернет-технологии — XML (eXtensible Mark-up Language) — расширяемого языка разметки, установленного консорциумом WWW (http://www.w3.org) в качестве нового промышленного стандарта, определяющего архитектуру программных средств Интернет следующего поколения. В данной работе приведены результаты исследования этой технологии, основные возможности XML, правила и рекомендации по его применению.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна, 2000

Перевод автора

Galaktionov V.V.
P10-2000-44

XML (eXtensible Mark-up Language): Industrial Standard,
Determining Architecture of the Next Generation
of the Internet Software

The past 1999 became the period of standing of the new Internet technology — XML (eXtensible Mark-up Language), the language of sectoring established by a Consortium WWW (http://www.w3.org) as a new industrial standard, determining architecture of the next generation Internet software. In this message the results of a research of this technology, basic opportunities XML, rules and recommendations for its application are given.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna, 2000

Редактор Е.Ю.Шаталова. Макет Н.А.Киселевой

Подписано в печать 28.03.2000
Формат 60 × 90/16. Офсетная печать. Уч.-изд. листов 2,43
Тираж 300. Заказ 51941. Цена 2 р. 92 к.

Издательский отдел Объединенного института ядерных исследований
Дубна Московской области