

Пространства имен в XML и SXML

Д.А. Лизоркин, К.Ю. Лисовский

Московский Государственный Университет, Институт Системного Программирования РАН

Пространства имен XML обеспечивают простую методику получения имен элементов и атрибутов, используемых в документах Расширенного Языка Разметки, что достигается путем ассоциации имен с пространствами имен, идентифицируемых с помощью ссылок URI. SXML -- это представление XML Infoset в форме S-выражений.

Пространства имен, поддерживаемые SXML, могут рассматриваться как надмножество пространств имен в XML. В данной статье мы обсуждаем дизайн и реализацию пространств имен в SXML, и предоставляем сравнение с пространствами имен в XML. Обсуждение иллюстрируется Моделью Описания Ресурсов (Resource Description Framework) и Dublin Core, которые активно используются в приложениях электронных библиотек.

1 Введение

Хотя Рекомендация XML [1] была введена в употребление без идеи пространства имен, на пространствах имен основаны большинство технологий, имеющих отношение к XML, и Рекомендация о Пространствах Имен в XML [2] была опубликована Консорциумом Мировой Сети (World Wide Web Consortium) в январе 1999 года, вскоре после появления самой Рекомендации XML.

Пространства имен предоставляют механизм по различению имен, используемых в XML-документах, позволяя сохранять эти имена простыми и осмысленными, и, в то же время, уникальными.

Раздел [2](#) объясняет причину появления пространств имен в XML и дает обзор механизма организации пространств имен XML. Раздел [3](#) рассматривает представление пространств имен в SXML, реализации XML Information Set в виде S-выражений. В разделе [4](#) вводится Dublin Core, набор мета-элементов, предназначение которых -- облегчать поиск электронных ресурсов, включая электронные библиотеки (ЭБ). Раздел [5](#) описывает представление Dublin Core на XML/RDF, приводя, таким образом, иллюстративный пример практического применения пространств имен XML в контексте основанных на XML электронных библиотек.

[2](#) Пространства имен в XML

В модели данных XML подразумевается, что XML-документ содержит в себе дерево элементов. Каждый элемент имеет тип [\[2\]](#) (иногда называемый также именем тэга) и набор атрибутов; каждый атрибут состоит из имени и значения. Тип элемента в общем случае предназначен для того, чтобы отражать семантическое назначение элемента. Как обсуждается в [\[3\]](#), приложения обычно анализируют тип элемента и его атрибуты, чтобы определить, как обрабатывать этот элемент.

В XML версии 1.0 без пространств имен, типы элементов и имена атрибутов представляют собой неструктурированные строки, использующие ограниченный набор символов, и напоминают идентификаторы в языках программирования. Это вызывает проблемы в распределенных средах, таких как Веб, потому что не существует способа гарантировать уникальность типа элемента, т.е. один и тот же тип элемента может оказаться у нескольких семантически различных элементов. Например, один XML-документ может использовать элементы `part` для описания частей книг, другой документ может использовать элементы `part` для описания частей машин:

```
<book> <part>...</part> <part>...</part> </book> <car> <part>...</part> </car>
```

Приложение не имеет возможности узнать, как обрабатывать элемент `part`, если ему не предоставить некоторую дополнительную информацию, внешнюю по отношению к XML-документу. В нашем простом примере можно различить элементы `part`, анализируя типы их родительских элементов (`book` vs. `car`); однако в общем случае такой подход неудобен.

Аналогичные коллизии имен могут возникнуть и с XML-атрибутами. Рассмотрим первого разработчика, который использует атрибут `color`, чтобы выразить тот факт, что внешнее представление элемента `title` на экране должно иметь красный цвет:

```
<title color="red">...</title>
```

а второй разработчик использует атрибут `color`, чтобы определить цвет содержимого элемента при выводе на принтер:

```
<title color="red" color="gray16"/>...</title>
```

Мы не только не способны различить эти два атрибута. Элемент в последнем примере -- неправильный (not a well-formed) XML-элемент, поскольку Рекомендация XML не разрешает XML-элементу иметь несколько атрибутов с одинаковыми именами.

Рекомендация о Пространствах Имен в XML пытается улучшить эту ситуацию за счет расширения модели данных, которая позволила бы квалифицировать типы элементов и имена атрибутов с помощью Унифицированного Идентификатора Ресурса (Uniform Resource Identifier, URI). Таким образом, документ, описывающий части машин, может использовать тип `part`, квалифицированный одним Унифицированным Идентификатором Ресурса; а документ, описывающий части книг, может использовать тип `part`, квалифицированный другим Унифицированным Идентификатором Ресурса. Как отмечается в [3], роль Унифицированного Идентификатора Ресурса в имени -- просто позволить приложению распознать имя. Не дается никаких гарантий относительно реального существования какого-либо ресурса по данному URI. Рекомендация о Пространствах Имен в XML не требует, чтобы типы элементов и имена атрибутов были *квалифицированными именами* (еще называемые *полными именами*); они также могут оставаться *неквалифицированными именами* (которые также называют *локальными именами*).

Рекомендация о Пространствах Имен в XML квалифицирует имена с помощью Унифицированных Идентификаторов Ресурсов косвенным способом, который основан на идее префикса. Если тип элемента или имя атрибута содержит двоеточие, то часть имени, стоящая перед двоеточием, рассматривается как префикс, а часть имени после двоеточия -- как локальное имя. Префикс `foo` ссылается на Унифицированный Идентификатор Ресурса, определяемый значением атрибута `xmlns:foo`.

Например, типы всех XML-элементов, описывающих книги, можно квалифицировать каким-то общим Унифицированным Идентификатором Ресурса -- `"http://www.books.com/xml"`. Тогда элемент, который описывает части книг (т.е. локальное имя которого -- `part`), будет представлен на XML как:

```
<books:part xmlns:books="http://www.books.com/xml">...</books:part>
```

Ситуация с именами атрибутов -- точно такая же, как и с типами элементов. Например:

```
<title xmlns:display="http://www.computer-displays.org"
xmlns:printer="http://www.bwprinters.net" display:color="red"
printer:color="gray16"/>...</title>
```

Все эти атрибуты `xmlns` достаточно громоздки, поэтому Рекомендация о Пространствах Имен в XML позволяет их наследовать [3]: если префикс `foo` используется в имени тэга, но элемент не содержит атрибута `xmlns:foo`, то будет использоваться значение атрибута `xmlns:foo` из родительского элемента; если родительский элемент также не содержит атрибута `xmlns:foo`, то будет использоваться значение атрибута `xmlns:foo` из его родительского элемента, и т.д. Например,

```
<books:book xmlns:books="http://www.books.com/xml"> <books:part>...</books:part>
<books:part>...</books:part> </books:book>
```

префикс `books:` может быть описан только один раз на верхнем уровне, и затем использоваться несколько раз в дочерних элементах, без дополнительных описаний.

Часто возникает ситуация, когда большинство элементов в документе имеют квалифицированные типы элементов с одним и тем же Унифицированным Идентификатором Ресурса. Рекомендация о Пространствах Имен для XML имеет специальный синтаксис, чтобы сделать этот случай еще более удобным для записи. Атрибут `xmlns` определяет Унифицированный Идентификатор Ресурса, который будет использоваться для квалификации всех типов элементов, не содержащих префикса. Атрибут `xmlns` наследуется точно так же, как и атрибуты с префиксами `xmlns:`. Таким образом, предыдущий пример можно переписать как:

```
<book xmlns="http://www.books.com/xml"> <part>...</part> <part>...</part>
</book>
```

Необходимо отметить, что атрибут `xmlns` не влияет на имена атрибутов, не содержащих префиксов.

[3](#) Пространства имен в SXML

Поскольку SXML уже был введен в нашей предыдущей статье [4], в данном разделе мы обсудим дизайн и реализацию пространств имен в SXML.

XML квалифицирует имена с помощью Унифицированных Идентификаторов Ресурсов косвенным способом (основанном на идее префикса), поскольку Унифицированный Идентификатор Ресурса в общем случае может содержать буквы, которые не допускаются в корректном XML-имени. Однако любая буква,

которая может встречаться в Унифицированном Идентификаторе Ресурса, разрешена и в SXML-имени, т.к. SXML-имя -- это *символ* языка программирования Схема. Из данного факта следует очень важное свойство SXML: в отличие от XML, SXML обладает способностью квалифицировать имена с помощью Унифицированных Идентификаторов Ресурсов непосредственным образом, используя Унифицированный Идентификатор Ресурса (вместо префикса) в полном имени. Например:

(<http://www.books.com/xml:book> (<http://www.books.com/xml:part> ...)
(<http://www.books.com/xml:part> ...))

Самое правое двоеточие в SXML-имени отделяет локальное имя от Унифицированного Идентификатора Ресурса, определяющего пространство имен.

Хотя такие длинные имена SXML-элементов в написанном виде выглядят громоздкими, они эффективны как структуры данных с точки зрения занимаемой памяти, поскольку являются *символами* языка Схема. Неважно, насколько длинным будет имя *символа*, -- это длинное имя представлено только один раз, в таблице символов. Все дальнейшие вхождения *символа* -- это лишь ссылки на соответствующую ячейку в таблице символов [5].

Подобное представление также согласуется с Рекомендацией о Пространствах Имен, которая говорит: "Заметим, что префикс используется только как контейнер для хранения названия пространства имен. При построении имен, область действия которых выходит за пределы первоначального документа, приложения должны использовать название пространства имен, а не префикс."

В SXML мы можем использовать либо контейнеры, либо сами названия пространств имен.

Помимо непосредственного способа расширения имен Унифицированными Идентификаторами Ресурсов, SXML поддерживает концепцию *идентификаторов пространств имен*, которые определенным образом напоминают префиксы пространств имен в XML. Аналогично префиксу, *идентификатор пространства имен* соответствует Унифицированному Идентификатору Ресурса, определяющему пространство имен. Отличительной чертой *идентификаторов пространства имен* является то, что существует взаимно однозначное соответствие между ними и Унифицированными Идентификаторами Ресурсов, определяющими пространства имен. Это в общем случае не так для префиксов пространства имен в XML. Например, различные префиксы могут определять один и тот же Унифицированный Идентификатор Ресурса; префиксы могут переопределяться в дочерних элементах.

Идентификатор пространства имен, таким образом, служит в качестве сокращения для Унифицированного Идентификатора Ресурса в SXML-именах. Соответствие между идентификаторами пространства имен и Унифицированными Идентификаторами Ресурсов задается в служебной вершине *NAMESPACES*,

которая располагается перед элементом документа. Пример описания книги с помощью Модели Описания Ресурсов (Resource Description Framework, RDF), выраженный в виде законченного SXML-документа с пространствами имен, показан в таблице [1](#).

```
(*TOP* (@@ (*NAMESPACES* (rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#") (dc "http://purl.org/dc/elements/1.1/")) (*PI* xml "version=\"1.0\"" (rdf:RDF (rdf:Description (dc:creator "Karl Mustermann") (dc:title "Algebra") (dc:subject "mathematics") (dc:date "2000-01-23") (dc:language "EN") (dc:description "An introduction to algebra")))))
```

Таблица 1: Описание на RDF, представленное в виде SXML-документа с пространствами имен

Таблица [2](#) иллюстрирует соответствие между префиксами в XML, квалифицированными явным образом именами в SXML, и идентификаторами пространства имен в SXML. Приведенный документ содержит некоторое описание ресурса, выраженное с помощью Модели Описания Ресурсов (Resource Description Framework), которая будет подробно рассмотрена в разделе [5](#). Модель Описания Ресурсов имеет собственное пространство имен, и для него обычно используется префикс `rdf:`. Предположим, что еще одно пространство имен, имеющее Унифицированный Идентификатор Ресурса "`http://www.resources-of-different-family.com`", встречается в этом документе. Поскольку "`rdf`" является наглядным сокращением для этого Унифицированного Идентификатора Ресурса, возможно, что `rdf:` будет использоваться для него в качестве префикса. В XML подобная ситуация вызывает определенную путаницу, поскольку один и тот же префикс будет использоваться для двух различных Унифицированных Идентификаторов Ресурсов, и потребуются переопределения значений префикса.

В SXML с квалифицированными явным образом именами, документ выглядит понятно, хотя это достигается ценой длинных имен.

SXML с идентификаторами пространства имен должен будет использовать два таких идентификатора, -- из-за взаимно однозначного соответствия между идентификаторами пространств имен и Унифицированными Идентификаторами Ресурсов.

```

<?xml version="1.0"?> <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"> <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-grammar"> <rdf:editor xmlns:rdf="http://www.resources-of-different-family.com"> <rdf:Description xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"> <rdf:fullName xmlns:rdf="http://www.resources-of-different-family.com"> Dave Beckett</rdf:fullName> </rdf:Description> </rdf:editor> </rdf:Description> </rdf:RDF> (*TOP* (*PI* xml "version=\"1.0\""
(http://www.w3.org/1999/02/22-rdf-syntax-ns#:RDF
(http://www.w3.org/1999/02/22-rdf-syntax-ns#:Description (@
(http://www.w3.org/1999/02/22-rdf-syntax-ns#:about "http://www.w3.org/TR/rdf-syntax-grammar")) (http://www.resources-of-different-family.com:editor
(http://www.w3.org/1999/02/22-rdf-syntax-ns#:Description (http://www.resources-of-different-family.com:fullName "Dave Beckett"))))) (*TOP* (@@
(*NAMESPACES* (rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#") (rodf "http://www.resources-of-different-family.com"))) (*PI* xml "version=\"1.0\""
(rdf:RDF (rdf:Description (@ (rdf:about "http://www.w3.org/TR/rdf-syntax-grammar")) (rodf:editor (rdf:Description (rodf:fullName "Dave Beckett"))))))

```

Таблица 2: XML документ, содержащий переопределения префиксов (наверху), и его представление в SXML с использованием квалифицированных явным образом имен (в центре), и с использованием идентификаторов пространств имен (внизу).

В общем случае, один префикс пространства имен в пределах XML-документа может быть ассоциирован с несколькими разными Унифицированными Идентификаторами Ресурсов; однако XML-документы без подобных множественных определений (т.е. такие, где соответствие между префиксами и Унифицированными Идентификаторами Ресурсов взаимно однозначно), составляют существенное подмножество "практических" XML-документов. Поскольку правила хорошего стиля предлагают производить все объявления префиксов в элементе документа, концепция префиксов в XML фактически очень близка к концепции идентификаторов пространства имен в SXML.

[4](#) **Dublin Core**

Нахождение требуемой информации в Мировой Паутине становится все более проблематичным из-за взрывного роста сетевых ресурсов.

Существующее в настоящее время Веб-индексирование быстро эволюционировало, чтобы удовлетворить потребностям к инструментам по поиску ресурсов. Однако подобное индексирование, будучи полезным, является плохой заменой более разнообразным возможностям, которые доступны при описании ресурсов.

Библиотекари, исследователи в области электронных библиотек и специалисты по разметке текстов собрались на приглательном семинаре, проведенном в марте 1995 года, чтобы обратиться к проблеме нахождения ресурсов в сети. Это деятельность переросла в серию родственных семинаров и вспомогательных мероприятий, которые все вместе стали известны как Серия Семинаров по Метаданным Dublin Core.

Одним из основных результатов этих усилий стал набор элементов, которые коллективным мнением участников данных семинаров выбраны как основные элементы в области междисциплинарного поиска ресурсов. Термин "Dublin Core" применяется к этому основному ядру описательных элементов [6].

Элементы разделяются на три группы, которые приблизительно обозначают класс или область хранящейся в них информации:

- 1. элементы, относящиеся, главным образом, к Содержимому ресурса,
- 2. элементы, относящиеся, главным образом, к ресурсу с точки зрения Интеллектуальной Собственности, и
- 3. элементы, относящиеся, главным образом, к Представлению ресурса.

Таблица 3 демонстрирует эти 3 группы из 15 элементов Dublin Core.

Содержимое	Интеллектуальная Собственность	Представление
Заголовок (Title)	Создатель (Creator)	Дата (Date)
Предмет (Subject)	Издатель (Publisher)	Формат (Format)
Описание (Description)	Помощник (Contributor)	Идентификатор (Identifier)
Тип (Type)	Права (Rights)	Язык (Language)
Источник (Source)		
Отношение (Relation)		
Освещение (Coverage)		

Таблица 3: Мета-элементы Dublin Core

Любой элемент необязателен, или может встречаться несколько раз. Мета-элементы могут появляться в любом порядке. Упорядоченность нескольких вхождений одного и того же элемента (например, ?????????) может иметь значение, предполагаемое автором мета-описания; однако сохранение порядка не гарантируется в каждой пользовательской среде. В частности, RDF/XML поддерживает упорядочивание, в то время как HTML -- нет.

Мы дадим короткое описание нескольких элементов из Dublin Core, которые имеют особый интерес в контексте электронных библиотек:

Заголовок (Title)

Имя, данное ресурсу.

Автор или Создатель (Creator)

Человек или организация, которые в первую очередь ответственны за интеллектуальное содержимое ресурса, например, авторы в случае печатных документов.

Предмет и Ключевые Слова (Subject)

Тема ресурса. Обычно, тема выражается как ключевые слова или фразы, описывающие предмет или содержимое ресурса.

Описание (Description)

Текстовое описание содержимого ресурса, например, аннотация в случае объектов, представляющих собой документы.

Тип Ресурса (Type)

Категория ресурса, такие как домашняя страница, роман, поэма, деловая бумага, технический отчет, эссе, словарь.

Идентификатор Ресурса (Identifier)

Строка или число, уникальным образом идентифицирующие ресурс. Примеры для сетевых ресурсов включают
Унифицированный
Идентификатор Ресурса (Uniform Resource Identifier, URI) и
Унифицированное Имя Ресурса (Uniform Resource Name, URN).
Другие уникальные в глобальном масштабе идентификаторы, такие как Международный Стандартный Номер Книги (International Standard Book

Number, ISBN) или другие формальные имена -- также являются кандидатами для этого элемента.

Мета-элементы, как правило, имеют некоторое отношение к ресурсу, который они описывают, и, если формат ресурса позволяет, они могут быть включены внутрь тела ресурса. Два таких формата -- это Гипертекстовый Язык Разметки (Hypertext Markup Language, HTML) [7] и Расширяемый Язык Разметки (Extensible Markup Language, XML). В настоящее время активно используется HTML, но, с появлением стандарта XML, XML совместно с Моделью Описания Ресурсов (Resource Description Framework, RDF) обещают значительно более выразительные средства для представления мета-данных.

В разделе 5 данной статьи мы рассмотрим представление Dublin Core на RDF/XML.

[5](#) Представление Dublin Core в Модели Описания Ресурсов (RDF)

Модель Описания Ресурсов (Resource Description Framework, RDF), разработанная под эгидой Консорциума Всемирной сети (World Wide Web Consortium, W3C) -- это инфраструктура, позволяющая осуществлять представление, передачу и повторное использование структурированных метаданных [8]. Модель Описания Ресурсов использует XML в качестве общего синтаксиса для передачи и обработки метаданных.

RDF -- результат совместных

потребностей нескольких сообществ в выразительной и гибкой архитектуре для поддержки метаданных в сети. RDF -- это коллективная разработка, в которую вкладывают свои интеллектуальные ресурсы несколько компаний, являющихся членами Консорциума Всемирной сети. RDF исходит из дизайна XML, а также из предложений, поданных компаниями Microsoft и Netscape. Усилия других групп в области метаданных, включая Dublin Core и Warwick Framework, также повлияли на дизайн RDF.

RDF предоставляет модель для описания *ресурсов*. RDF определяет ресурс как любой объект, который может быть уникальным образом идентифицирован с помощью Унифицированного Идентификатора Ресурса (Uniform Resource Identifier, URI). Ресурсы имеют *свойства* (атрибуты или характеристики). Набор свойств, которые относятся к одному и тому же ресурсу, называется *описанием*. В основе RDF находится независимая от синтаксиса модель для представления ресурсов и соответствующих им описаний.

Структура, лежащая в основе любого выражения в RDF, может рассматриваться как ориентированный помеченный граф, который состоит из вершин и помеченных дуг [9]. Граф RDF -- это набор троек: *Вершина-предмет*, *Дуга-Свойство* и *Вершина-объект*. Тройка показана на Рисунке 1.



Рисунок 1: Тройка в RDF

Каждая дуга обозначает утверждение о некотором отношении между предметами, обозначаемыми вершинами, которые соединяет эта дуга. Данное утверждение имеет три части:

1. *свойство*, описывающее некоторое отношение (которое также называется *предикатом*),
2. значение, являющееся *предметом* утверждения, и
3. значение, являющееся *объектом* утверждения.

Направление дуги существенно: она всегда указывает в сторону объекта утверждения.

Например, модель данных, соответствующая утверждению "*автор Документа 1 -- Джон Смит*", имеет объект *Документ 1*, дугу с пометкой *автор* и соответствующий предмет *Джон Смит*. Модель данных, отвечающая данному утверждению, графически выражается на рисунке [2](#).



Рисунок 2: Тройка, выражающая соотношение между документом и его автором

Общий смысл графа RDF -- это соединение (т.е. логическое "И") всех утверждений, которые он содержит.

Чтобы закодировать граф на XML, вершины и дуги представляются с помощью имен XML-элементов, имен атрибутов, содержимого элементов и значений атрибутов [10].

Граф может рассматриваться как набор всевозможных путей вида ???????, ?????, ???????, ?????, ..., ???????, которые покрывают граф целиком. В RDF/XML эти пути превращаются в последовательности вложенных друг в друга элементов, в которых чередуются элементы, кодирующие Вершины, и элементы, кодирующие Дуги. Такая последовательность элементов называется серией полос Вершина/Дуга. Вершина в начале пути становится самым внешним элементом, следующая Дуга становится дочерним для нее элементом, и так далее. Полосы обычно располагаются около корня XML-документа и всегда начинаются с Вершин.

Чтобы создать законченный RDF/XML-документ, сериализованный в вид XML граф должен быть помещен внутрь элемента `rdf:RDF`, который становится элементом верхнего уровня (элементом документа, *document element*). Префикс `rdf:` связан с пространством имен RDF -- "`http://www.w3.org/1999/02/22-rdf-syntax-ns#`" -- чтобы приложения могли распознать данный XML-документ как документ RDF/XML.

RDF дает возможность разным сообществам определять собственную семантику описаний. Важно, однако, избегать семантической

двусмысленности описаний между различными сообществами. Свойство типа "автор", например, может иметь более широкое или более узкое значение в зависимости от нужд конкретного сообщества. Будет проблематичным, если множество сообществ будут использовать свойство одного типа, чтобы подразумевать под ним совершенно разные вещи. Чтобы предотвратить это, RDF уникальным образом идентифицирует свойства с использованием механизма пространства имен в XML. Пространства имен в XML предоставляют возможность однозначно идентифицировать семантику и соглашения, управляющие использованием конкретного свойства, за счет уникальной идентификации руководящей организации. В качестве примера можно привести свойство типа "автор", определенное Инициативой Dublin Core как "лицо или организация, ответственные за создание интеллектуального содержимого ресурса" и специфицированное элементом CREATOR Dublin Core. Пространство имен в XML используется для однозначной идентификации схемы для словаря Dublin Core, путем указания на ресурс Dublin Core, который определяет соответствующую семантику.

Пятнадцать базовых элементов из Набора Элементов Dublin Core могут рассматриваться как составляющие единое пространство имен [11], на которое можно ссылаться во Всемирной сети как "http://purl.org/dc/elements/1.1/".

Каждый элемент Dublin Core представляется в RDF как XML-элемент:

- идентификатор пространства имен которого -- "http://purl.org/dc/elements/1.1/", и

- локальное имя которого совпадает с именем элемента Dublin Core.

XML-элемент `rdf:RDF` является элементом верхнего уровня (элементом документа) в документе RDF/XML; и каждый описываемый ресурс включается в элемент `rdf:Description`. Таблица 4 дает пример описания Dublin Core, выраженного на RDF/XML.

```
<?xml version="1.0"?> <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description>
    <dc:creator>Karl
    Mustermann</dc:creator>
    <dc:title>Algebra</dc:title> <dc:subject>mathematics</dc:subject>
    <dc:date>2000-01-23</dc:date>
    <dc:language>EN</dc:language>
    <dc:description>An introduction to algebra</dc:description>
  </rdf:Description> </rdf:RDF>
```

Таблица 4: Описание Dublin Core, выраженное в RDF/XML

[6](#) Заключение

Пространства имен в XML расширяет язык XML концепцией полных имен, область действия которых выходит за рамки того документа, где они были определены. Это свойство жизненно

важно для документов, содержащих несколько словарей разметки, для разрешения проблем распознавания и коллизий; и это свойство играет ключевую роль для многих XML-технологий и приложений.

Пространства имен, поддерживаемые SXML, могут рассматриваться как надмножество пространств имен в XML. Помимо поддержки префиксов пространства имен, SXML обеспечивает возможность непосредственного использования Унифицированных Идентификаторов Ресурсов в полных именах, а также механизм идентификаторов пространств имен.

Литература

[1]

Extensible Markup Language (XML) 1.0 (Second Edition). W3C Recommendation 6 October 2000.
<http://www.w3.org/TR/REC-xml>

[2]

Namespaces in XML. World Wide Web Consortium
14-January-1999.

<http://www.w3.org/TR/REC-xml-names/>

[3]

James Clark. XML Namespaces.
February 4, 1999.

<http://www.jclark.com/xml/xmlns.htm>

[4]

Д.А.
Лизоркин,
К.Ю.
Лисовский.
SXML: XML-
документ
как S-выраж
ение. Элект
ронные
библиотеки,
2003, Том 6,
Выпуск 2.

[http://www.
elbib.ru/jo
urnal/2003/
200302/LK/L
K.ru.html](http://www.elbib.ru/journal/2003/200302/LK/LK.ru.html)

[5]

Oleg
Kisel
yov.
SXML
, Revi
sion
2.5.
Augu
st 9,
2002.

[http
://o
kmij
.org
/ftp
/Sch
eme/
SXML
.htm
l](http://okmij.org/ftp/Scheme/SXML.html)

[6]

