Московский государственный технический университет имени Н.Э. Баумана

Домашнее задание по курсу: “Технология XML”

Тема: “Технология XSLT”

Выполнила: студентка Заварзина В.В.

Проверил: Гапанюк Ю.Е.

**Москва, 2010**

**Содержание**

Введение

XSLT как язык

Преобразования

Области применения XSLT

XSLT в архитектуре клиент-сервер

XSLT на стороне сервера

XSLT на стороне клиента

Пространство имен в XSLT

XSLT и другие языки

**Введение**

XSLT означает eXtensible StyleSheet Language for Transformation, что на русский язык традиционно переводится как “расширяемый язык стилей для преобразований”. Работа над XSLT была инициирована проектом XSL - eXtensible StyleSheet Language (расширяемым языком стилей).

Спецификация XSLT гласит, что это язык для преобразования одних XML – документов в другие XML – документы. Но по мере развития язык перерос ее и теперь уместнее сказать, что XSLT – это язык для преобразования структуры документов.

**XSLT как язык**

Любое преобразование можно условно поделить на 3 составляющие:

1. Обращение к преобразуемому объекту;
2. Создание результата преобразования;
3. Логика, связывающая первые два действия и направляющая процесс преобразования.

Применительно к преобразованию XML – документов первая подзадача означает получение информации, которую этот документ содержит – в том числе и информацию о структуре, которая является неотъемлемой его частью. Обращение в данном случае имеет несколько смыслов, в том числе – опрашивать, делать запросы, вычислять, выбирать; в общем смысле – задавать о документе вопросы и получать на них ответы. Для этой цели в **XSLT** служитязык, называемый XPath – язык путей в XML – документах. XPath является лаконичным, но при этом очень мощным средством обращения к XML – документам. Роль XPath в XSLT так велика, что их можно было бы считать единым целым, если бы только XPath не использовался также и в других языках, предназначенных для работы с XML.

Вторая и третья условные части преобразования являются прерогативой самого XSLT. XSLT – это XML – язык в полном смысле этого слова: программы на XSLT являются хорошо оформленными XML – документами. XSLT также использует пространство имен; практически все имена, которые встречаются в XSLT рассматриваются как расширенные имена, характеризуемые локальной частью вкупе с URI – уникальным идентификатором пространства имен.

В отличие от традиционных императивных языков программирования, преобразование в XSLT не является последовательностью действий, которую необходимо выполнить для достижения результата. Преобразование – это набор шаблонных правил, каждое из которых определяет процедуру обработки определенной части документа.

XSLT – интерпретация “Hello? World!”, когда мы преобразуем документ:



Синтаксис XSLT, являющийся чистым XML, может показаться для языка программирования не совсем обычным, однако, вряд ли какой другой синтаксис был бы более удобным.

**Преобразования**

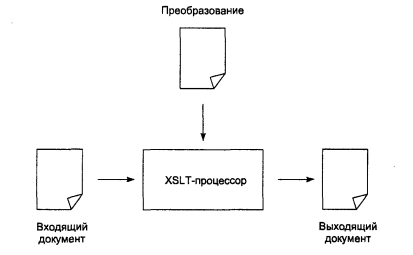
В преобразовании в основном участвуют три документа:

- входящий документ, который подвергается преобразованию;

- документ, который описывает само преобразование;

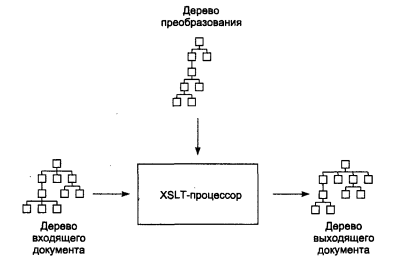
- выходящий документ, который является результатом преобразования.

Схема XSLT – преобразования



Процессор получает входящий документ и преобразование, и, применяя правила преобразования, генерирует выходящий документ – такова в общем внешняя схема картины. На самом деле процессор оперирует не самими документами, а древовидными моделями их структур – именно структурными преобразованиями занимается XSLT, оставляя за кадром синтаксис, который эти структуры выражает.

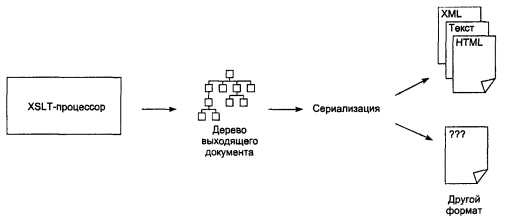
Древовидные структуры XSLT



Хотя практически все процессоры выполняют каждый из трех этапов, рабочей областью XSLT является только второй этап, этап преобразования. XSLT практически не контролирует парсинг входящего документа, как правило этим занимается встроенный или внешний SAX- или DOM- парсер.

Сериализация является процессом создания физической интерпретации результирующего дерева, а если эта задача делегируется XSLT – процессору, то преобразованию под силу контролировать физический вывод генерируемого документа.

Сериализация в XSLT



**Области применения XSLT**

В отличие от языка XML, предметную область XSLT задать очень легко. XSLT следует применять там, где необходимо преобразование одного документа в другой.

Естественно, XSLT имеет также и некоторые ограничения:

- XSLT не подходит для описания преобразований с очень сложной логикой;

- XSLT не подходит для преобразований, которые требуют сложных вычислений.

Первое ограничение связано с тем, что преобразование XSLT – это всего лишь набор элементарных правил. В подавляющем большинстве случаев этого достаточно для описания преобразования, однако встречаются также и такие задачи, для которых данного набора правил будет недостаточно. Например, древовидные структуры могут описываться математическими выражениями, но при этом преобразование для упрощения или вычисления этого дерева выражений может быть чересчур сложным для XSLT.

Второе ограничение является следствием простоты языка XPath, который используется в XSLT для вычислений. XPath предоставляет только самые простейшие вычислительные конструкции, которых явно недостаточно для сложных задач. Кроме того, функциональный стиль XSLT и отсутствие изменяемых переменных делают очень затруднительными многошаговые и циклические вычисления.

**XSLT в архитектуре клиент-сервер**

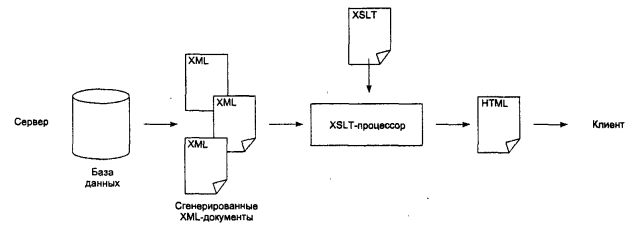
Многие из систем, применяющих XSLT, так или иначе, сводятся к клиент-серверной архитектуре, в которой клиент делает запрос, а сервер в качестве ответа возвращает некоторые данные. XSLT в таких решениях может использоваться для приведения структуры данных из внутреннего формата сервера к некоторому внешнему формату, понятному клиенту. Примером подобной системы может быть Web-сервер, предоставляющий клиентам информацию, которая динамически генерируется из базы данных.

Классическим и широко применяемым решением для такого рода задачи являются серверные компоненты, сервлеты и различные скриптовые языки, которые преобразуют запросы клиента в запросы к базе данных, а затем оформляют результаты выполнения в виде HTML и отсылают клиенту.

Минус этого решения в том, что оно слишком сильно зависит от презентации данных. Новая презентация или сравнительно серьезное исправление старой заставляют, чуть ли не полностью переписывать Web-приложение.

В подобных системах достаточно эффективно используется связка XML+XSLT. Вместо того чтобы генерировать по данным HTML-презентацию, можно создать XML-документ, и, применяя преобразования, возвращать клиенту именно тот результат, которого он ожидает.

Схема взаимодействия XML и XSLT в архитектуре клиент-сервер



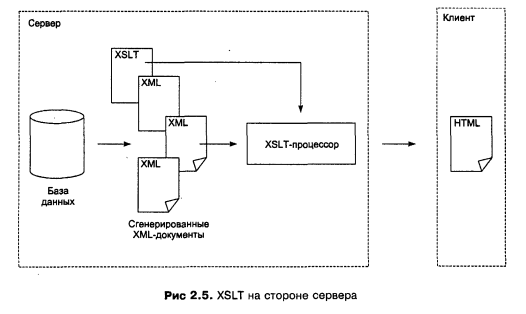
На этой схеме нет четкой границы, которая отделяла бы клиент от сервера. Дело в том, что существует два принципиально различных способа использования XSLT в подобной архитектуре: преобразования могут выполняться как на стороне сервера, так и на стороне клиента.

**XSLT на стороне сервера**

Применение XSLT на стороне сервера совершенно незаметно для клиента – он, как и раньше, в ответ на свой вопрос получает HTML или документ в другом требуемом формате. В данном случае связка XML+XSLT является дополнительным звеном, дополнительным уровнем абстракции, который позволяет отделять данные от презентации, добиваясь простоты и универсальности. Создание преобразований для генерации HTML по имеющимся XML-документам – задача совершенно иного плана, чем написание серверных приложений и программ, которые непосредственно работают с результатами выполнения запросов к базе данных.

Минусом этого способа является то, что мы все равно возвращаем клиенту представление данных, а не сами данные. Естественно, используя XSLT, множество доступных представлений расширить очень легко, но это множество в любом случае будет ограничено. Для большинства Web- систем этого более чем достаточно, но существующие Web-технологии больше ориентированы на представление данных, чем на сами данные – они стараются предвосхитить вопросы на заранее процедуры ответов.

XSLT на стороне сервера



**XSLT на стороне клиента**

Идея использования XSLT на стороне клиента заключается в том, чтобы отдавать клиенту отдельно нужные ему данные и отдельно преобразование, которое будет создавать для этих данных требуемое представление. Четкое разделение данных и их представления предоставит клиенту полную свободу распоряжаться полученной информацией. Преобразование в этом случае всего лишь предлагает возможную трактовку этой информации, ни к чему не обязывая.

Еще одним достоинством выполнения преобразований на стороне клиента является разгрузка сервера, ведь такой подход освобождает его от необходимости выполнять процедуру преобразования.

Основным ограничением этого способа является предложение, что программное обеспечение на стороне клиента сможет выполняться преобразования. К сожалению, решение такого вида может применяться только в очень ограниченном числе случаев – когда достоверно известно, что целевой клиент поддерживает XSLT.

преобразование документ сервер стиль

**Пространство имен в XSLT**

Для того, чтобы выделить элементы и атрибуты, которые принадлежат логической схеме XSLT, в этом языке применяется механизм пространств имен. Это означает, что в документе, преобразования элементы, относящиеся к XSLT, должны принадлежать его пространству имен.

Уникальный идентификатор ресурса пространства имен XSLT имеет вид: http://www.w3.org/199/XSL/Transform

Пример простого преобразования, в котором объявлено пространство имен:



**XSLT и другие языки**

Несмотря на то, что XSLT является вполне самостоятельным языком, его очень часто используют как составную часть в проектах, которые пишутся на других языках программирования. Тому существует множество причин. Попытаемся выделить главные из них:

* Традиционные императивные языки программирования очень плохо подходят для обработки древовидно структурированных данных. Программы, действия в которых непременно выполняются последовательно одно за другим, в общем случае не могут эффективно (с точки зрения компактности и понятности кода) обработать сложные иерархические структуры.
* В некоторых случаях XSLT-преобразования документов оказываются, наоборот, настолько сложны, что из соображений эффективности и простоты бывает намного легче использовать традиционные языки.
* Во многих проектах использование XSLT может обеспечить лёгкую и гибкую интеграцию. Например, если одним из этапов процедуры обмена XML-данными будет XSLT-преобразование, расширение количества форматов, известных системе, будет производиться не дописыванием исходного кода, а добавлением преобразований. А поскольку XSLT обеспечивает не только синтаксические, но и семантические преобразования, то есть преобразования на структурном уровне, роль этого языка в проектах интеграции, основанных на использовании XML, может быть очень велика.
* Использование XSLT-преобразований может коренным образом упростить создание web-ориентированных приложений. Надо сказать, что во многих случаях XSLT-преобразования просто избавляют от необходимости программировать что-либо на других языках; однако даже тогда, когда без традиционных подходов не обойдёшься, XSLT служит хорошую службу, обеспечивая простой, удобный и легко настраиваемый вывод фрагментов HTML.