СЕКЦИЯ 2. КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ

БИБЛИОТЕКА ФУНКЦИЙ ДИНАМИЧЕСКОГО ФОРМИРОВАНИЯ САЙТОВ НА ОСНОВЕ ТЕХНОЛОГИЙ XML-XSL

В.И. Гущинская, Д.И. Самаль
Беларусь, г. Минск

Введение

Классическая модель «клиент-сервер» подразумевает полностью синхронное взаимодействие между браузером и сервером, т.е. до тех пор, пока пользователь не выполнит какое либо действие, запросов от браузера к серверу не поступает и, соответственно, сервер для данного пользователя не выполняет никаких полезных действий [1]. После поступления запроса сервер начинает его обработку, а пользователь соответственно ждёт результатов. Несмотря на повсеместное использование, данная модель не лишена недостатков. Так как большинство страниц Интернета формируются динамически, то при использовании данной модели взаимодействия даже из-за малейшего видоизменения представления информации (например, иная сортировка данных таблицы) браузеру приходится обращаться к серверу, серверу – перевёрстывать всю страницу, а пользователю – просто ждать результата. Такая модель приводит к увеличенному объёму трафика, повышенной нагрузке на сервер и, как общий результат, - к неудобству в работе.

В то же время, мощностей современных компьютеров, с помощью которых пользователи выходят в Интернет, с лихвой хватает для выполнения большинства операций по изменению вида страницы, представления информации (т.е. динамической вёрстке страницы). Делегирование части подобных функций клиенту позволит избежать лишней загрузки сервера и снизить объём пересылаемой между клиентом и сервером информации (трафика).

1. Предлагаемый вариант «активный клиент-сервер»

Исходя из обозначенных выше предположений, была разработана библиотека функций для реализации «активного клиента», опирающаяся на известные технологии, и которая способна обеспечить принципиально новую функциональность web-приложений. Использовались следующие технологии:

1. DHTML и CSS для обеспечения дизайна;

2. DOM для обеспечения интерактивности;
3. XML и XSLT для работы с данными;
4. JavaScript, объединяющий всё вышеперечисленное.

Уже после разработки библиотеки стало известно об аналогичной разработке американских коллег – так называемой парадигмы "Ajax" [2,3], что означает "асинхронный JavaScript плюс XML", или, если быть более точным, "асинхронный JavaScript+CSS+DOM+XMLHttpRequest". На данный момент она уже воплощена в нескольких проектах, например Google Suggest [4] и Google Maps [5].

Таким образом, «активный клиент» - это принципиально новый подход к процессу формирования страницы, использование которого не представляет трудности для разработчиков сайтов.

2. Подробности реализации

В отличие от упомянутой парадигмы "Ajax" предлагаемая библиотека была ориентирована на взаимодействие с «пассивным» сервером, т.е. не производящим никакой обработки данных в принципе. Таким образом, роль «пассивного» сервера фактически сводится к удалённому хранилищу информации. «Активный» клиент, взаимодействующий с таким сервером должен, в данном случае, выполнять те функции, которые не доступны «пассивному» серверу: помимо получения информации с сервера в исходном – не преобразованном виде, производить XSL-трансформации над XML данными, формировать HTML-страницу и отображать конечный вариант в окне. Таким образом, функции сервера не выходят за рамки функций, выполняемых бесплатными хостинг-серверами: при подключении клиента сервер должен лишь отправить клиенту запрашиваемую HTML-страницу.

Далее в работу включается клиент. Получив единственный файл от сервера, он запрашивает остальные файлы: с данными (XML, HTML), инструкциями (XSL, JavaScript). Функцию тривиальной передачи данных сервер осуществить может. Клиент получает все необходимые файлы и преступает к выполнению функций динамического формирования данных: начинает производить трансформации, выполнять инструкции и формировать страницу, которую увидит пользователь. Если необходимо некоторое изменение отображённой информации, запрос не отправляется на сервер. Имея все необходимые данные, клиент и сам может осуществить обработку и сформировать страницу с обновлённой в соответствии с запросом информацией.

Отображение информации происходит с помощью технологий XML-XSL. В примере реализации сайта с помощью разработанной библиотеки способ отображения информации задаётся пользователем. Ин-

формация может быть структурирована в виде табличных данных, может быть представлена в виде всевозможных графиков, может быть удобна в виде гистограммы, функциональной зависимости, трёхмерного изображения. Все вышеперечисленные варианты отображения легко реализуемы и не требуют ни большой мощности сервера для генерации запрошенного изображения, ни высокоскоростных каналов передачи, ни длительного времени отклика для формирования результата. Это становится возможным благодаря тому, что клиенту передаются только непосредственно данные в виде набора XML-файлов и набор XSL-файлов, определяющих варианты визуального представления полученных данных, причём передача осуществляется единожды.

Далее, когда пользователь определится с выбором предложенных вариантов отображения (или их комбинацией), запрос, отправленный на сервер, перехватывается библиотекой функций и выполняется непосредственно браузером, затрачивая его собственную мощность, не загружая канал передачи и, как следствие, уменьшая временные задержки для отображения требуемого результата.

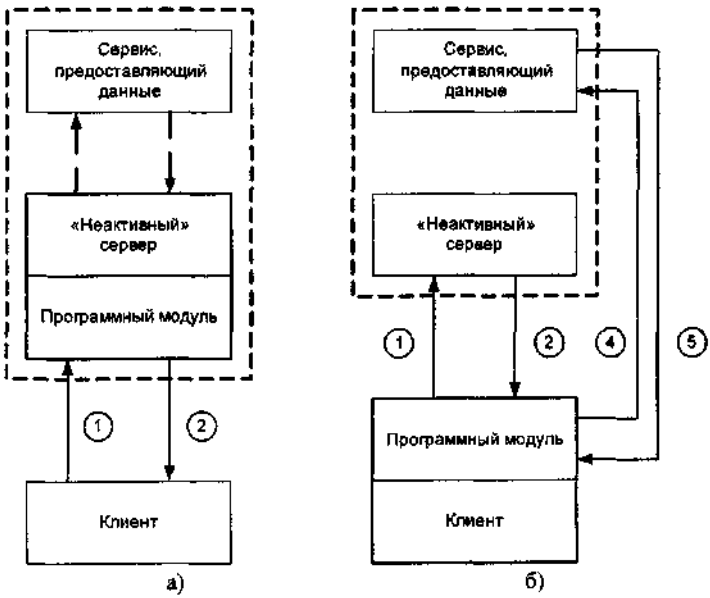

Рис. 1 Схема работы активного сайта: а) при первой загрузке; б) при последующих загрузках

3. Функционирование сайта с применением «активного клиента»

В качестве примера для демонстрации преимуществ данной технологии была выбрана реализация сайта для отображения биржевых сводок. Сайт должен получать информацию от сервисов – баз данных бирж и отображать её в удобном для пользователя виде. Информация отображается в виде графиков либо таблиц статистики, возможна сортировка таблиц и отображение заданного графика отдельно или нескольких одновременно, а также предсказание поведения кривых графика в следующий момент времени с заданной вероятностью.

Схема работы подобного сайта продемонстрирована на рисунке 1. Последовательность действий работы сайта определяется цифрами.

Пользователь загружает первую страничку сайта. В процесс загрузки подключается библиотека функций. Она запрашивает с сервера дополнительные информационные файлы, необходимые для его работы. Таким образом, выполняются этапы 1-2. Все необходимые данные фиксируются внутри библиотеки (3). Библиотека определяет, какие данные необходимо отображать в соответствии с запросом пользователя. Далее она формирует запросы к сервисам, предоставляющим эти данные (4) и получает их (5). Преобразует полученные данные к виду, необходимому для их отображения, и предоставляет пользователю (6). Это полный цикл работы библиотеки функций. Однако, например, для перехода на следующую страницу библиотека уже не должна обращаться к серверу (пункты 1 и 2), эти этапы производятся один раз при подключении клиента и не повторяются до тех пор, пока не будет исчерпан лимит времени обновления сайта. Т.е. цикл работы сокращается до этапов 3-6.

Таким образом, функции обработки и отображения информации (построение страничек, переход по ссылкам, формирование графиков и статистики, формирование запросов к базам данных и преобразование ответов на них в необходимый формат) выполняются на клиентской стороне библиотекой функций, в то время как функции сервера ограничиваются лишь хранением исходных данных.

4. Построение сайтов с использованием библиотеки

Для написания собственного сайта с использованием предложенной библиотеки функций необходимо разработать ряд файлов:

1. Информационная часть сайта, содержащая текст, картинки, информацию для построения таблиц, графиков и любые другие данные, необходимые для функционирования сайта, разбивается на отдельные файлы – каждый файл содержит определённую часть материала. Деление на файлы может быть осуществлено

либо постранично, либо более мелкими блоками в зависимости от сложности дальнейшей обработки данных, осуществляемой в процессе выполнения поставленных задач. Информация записывается в соответствии с форматом представления данных XML и сохраняется в файлы с одноименным расширением.

2. Для отображения информации разрабатываются XSL-трансформации. Допускается разработка стилей CSS и компонентов поведения HTC. Иногда для создания более гибкого поведения страницы требуется написать несколько функций на JavaScript.
3. Структура сайта оформляется в формате XML и сохраняется в файле SiteConfig.xml. Это обязательный файл.
4. Сайт дополняется модулями библиотеки.

Таким образом, создание сайта на основе технологий XML-XSL с использованием библиотеки или без её использования отличается лишь созданием дополнительного конфигурационного файла siteConfig.xml в первом случае. Однако структура этого файла примитивна и не требует никаких дополнительных знаний или умений. Применение же библиотеки при разработке сайтов определённых типов даёт некоторые преимущества, оговоренные выше.

К тому же предусматривается возможность неограниченного расширения функциональности библиотеки. Это достигается путём написания дополнительных функций. Включение вновь созданной функции осуществляется записью в конфигурационном файле. Вызов необходимой функции происходит автоматически без участия создателя сайта на этапе выполнения библиотеки в процессе чтения и разбора конфигурационного файла.

Заключение

Разработанная библиотека является альтернативным вариантом динамического формирования web-страниц на основе технологий XML, XSL. К её преимуществам можно отнести: минимизацию использования мощностей сервера, бóльшую гибкость (обновление алгоритмов работы происходит автоматически при начале работы с сайтом), меньший трафик. Единственным недостатком предлагаемого подхода является увеличение размера отсылаемых данных при самом первом обращении к серверу. Данный недостаток не существенен, т.к. впоследствии файлы библиотеки могут храниться в кэше браузера и обновляться только при модификации сайта разработчиком.

Сайты, реализуемые с помощью библиотеки: информационные ка-

талоги, библиотеки документации, архивы программ, биржевых новостей и т.д.

Принципиально не реализуемые сайты: электронные магазины (возможны только каталоги, без заказов), почтовые сервера, и т.п., т.е. сайты, со сложными базами данных, а так же требующие регистрации пользователя.

Использование разработанной библиотеки целесообразно, например, при хранении на сервере большого объёма информации, доступной всем пользователям, для обеспечения возможности её сортировки, отбора по заданным критериям, отображении в удобном для пользователя виде.

Однако все перечисленные выше ограничения исчезают, если сервер так же, как и клиент, активен и берёт на себя те функции, которые не может выполнить клиент. А именно – обеспечиваст интерактивность (т.е. обратную связь) и способен оперировать с базой данных, то есть реализовывать модель взаимодействия «активный клиент»-«активный сервер».

Таким образом, разработанная технология взаимодействия «активный клиент»- «сервер», в общем, и предлагаемая библиотека, в частности, могут применяться для реализации различных целей в зависимости от потребностей конкретного пользователя и обеспечивать максимальную функциональность при минимальных потребностях в мощности сервера, а также предоставлять возможность создания благоприятных условий для отображения информации в максимально удобной пользователю форме.

Литература

1. Хефлин Д., Ней Т. Разработка Web-скриптов. Библиотека программиста. – СПб.: Питер, 2001. – 496 с.
2. А.Ализар Ajax:новый подход к разработке веб-приложений. - Компьютерные вести №19, 3 мая 2005 - с.21.
3. Adaptive Path <http://www.adaptivepath.com/publications/essays/archives/000385.php>
4. Google Suggest, <http://www.google.com/webhp?complete=1&hl=en>
5. Google Maps, <www.maps.google.com>