



A SAS Institute White Paper

Стандартные сценарии разработки Web-приложений

Содержание

Введение	3
Элементы среды Web	3
Среда Web	3
Web серверы	4
Web браузеры	5
Прокси-серверы	5
Технологии разработки Web приложений	
Языки	6
Компонентные модели	7
Протоколы передачи данных	8
Системы интегрированной разработки	9
Выбор существующих технологий под нужды Вашего приложения	
Сценарий №1 — Web-публикации	10
Сценарий №2 — Динамическая генерация содержимого	13
Сценарий №3 — Распространение приложений	19
Продукты, предлагаемые SAS Institute	23
Заключение	26

Patterns for Producing Web-enabled Applications was written by Chip Kelly.

Translated by Roman Volynets.

Copyright © 1999 by SAS Institute Inc.
All rights reserved. Limited copies may be made for internal use only.
Credit must be given to the publisher.
Otherwise, no part of this publication may be reproduced without prior permission by the publisher.

Введение

На рынке существует множество средств и технологий разработки Web-приложений. Вы можете самостоятельно разобраться в том, какие из них подходят Вам для решения Ваших задач и какие из них стоит использовать, однако, это может занять достаточно много времени. Мы бы хотели в течение нескольких минут показать Вам как использование программного обеспечения SAS может обеспечить эффективную разработку Web-приложений в кратчайшие сроки.

Неотразимая привлекательность сети Web заключается в ее общедоступности. Всякий, обладающий браузером, может получить доступ к Вашему приложению, если только Вы сделаете приложение Web-совместимым. Далее всю сложность и все заботы, связанные с распространением и эксплуатацией Вашего приложения, берет на себя архитектура Web. Данные для приложения, также как и выполняемые модули находятся на Web-сервере и Вы управляете ими централизованно, сводя к минимуму возможные последствия для пользователей при изменении программной части или содержимого Вашего приложения. Если Вы разработали новую, более привлекательную версию приложения, или если данные, с которыми работает приложение изменились, Ваши пользователи не будут испытывать никаких изменений ни в способе доступа к приложению, ни в способе работы с ним. Все это звучит здорово, но сразу же встают многочисленные вопросы. Например:

- Когда использование CGI лучше, чем Java?
- Дает ли Dynamic HTML ту же степень контроля над клиентским интерфейсом, что и JavaScript?
- В каких случаях предпочтительнее использовать Dynamic HTML вместо Java для Web-клиентов?
- В какой момент генерация динамических отчетов по SQL запросам перестает удовлетворять требованиям пользователей?

Все это примеры реальных вопросов, с которыми сталкивается всякий, заинтересованный в использовании возможностей Web для предоставления пользователям доступа к данным компании и системам поддержки принятия решений.

Ряд программных продуктов SAS напрямую связан с Web-технологиями. Если эти программные продукты являются частью Вашего решения, то у Вас появляется значительная гибкость при разработке Web-приложений. В данной статье содержатся ответы на поставленные выше и многие другие вопросы, связанные с разработкой и реализацией Web-приложений. Данная статья также содержит рекомендации по использованию тех или иных продуктов SAS System для решения различных задач в различных ситуациях.

Элементы среды Web

Какие же результаты Вы хотите получить и какие цели стараетесь достичь созданием Web-приложений? Вероятно, Вы хотите, чтобы Ваши пользователи могли получать доступ к информации через их браузеры, работать с этой информацией через браузеры и опять же через браузеры выполнять различные приложения.

Давайте приведем небольшой обзор тех элементов, с которыми Вам придется работать, чтобы сделать Web составляющей частью Вашей системы.

Среда Web

Есть ли в Вашей организации внутренняя сеть intranet? Intranet представляет собой корпоративную сеть TCP/IP, в которой есть Web-сервер и один или несколько компьютеров, оснащенных Web-браузерами. Internet представляет собой локальную сеть, объединяющую тысячи Web-серверов и миллионы пользователей Web-браузеров по всему миру. Как Вы уже наверное отметили, Web-серверы и Web-браузеры являются основными элементами среды Web. Мы упомянем еще один элемент — прокси-сервер (firewall) — который является несущественным с точки зрения простых коммуникаций, но приобретает важное значение с точки зрения вопросов безопасности.

Web-серверы

На рынке существует множество **Web-серверов**. Большинство из них предлагают сходную функциональность по отношению к ключевым составляющим:

- **HTTP** (Hyper Text Transport Protocol) — протокол передачи гипертекстовой информации, в более общем смысле — стандартный протокол передачи данных между элементами сети Web.
- **CGI** (Common Gateway Interface) — стандартный шлюзовой интерфейс — позволяет Web-клиентам инициировать выполнение программ на Web-сервере. CGI определяет для Web-сервера стандартные методы вызова других исполняемых программ. Программы CGI могут быть созданы с использованием различных языков программирования и поддерживаются практически всеми Web-серверами.
- **Сервлеты Java** (Java servlets) — представляют собой альтернативный метод вызова программ на серверной стороне. Можно сказать, что сервлеты Java — это CGI программы, написанные на Java и имеющие лучшие характеристики в отношении производительности и возможности сохранения состояния (sessions). Сервлеты также автоматически поддерживаются большинством Web-серверов. Для тех же серверов, которые не имеют встроенной поддержки сервлетов, доступны различные дополнительные модули (plug-ins), которые эту поддержку обеспечивают.

- **Active Server Pages (ASP)** — являются еще одним методом вызова выполняемых программ на сервере. Этой возможностью обладают Web-серверы компании Microsoft, хотя наблюдаются некоторые попытки переноса этой технологии и на другие платформы.

Web-браузеры

На рынке существует также много **Web-браузеров**. Однако, большинство организаций предпочитают выбирать между двумя браузерами — Microsoft Internet Explorer и Netscape Navigator. Несмотря на то, что между браузерами много различий, мы остановим внимание на общих чертах, присущих всем браузерам.

- **HTML (Hyper Text Markup Language)** — язык гипертекстовой разметки — общий язык, на котором «говорят» все Web-браузеры. HTML файлы скачиваются с Web-сервера Web-браузером, который форматирует страницу Web согласно предписаниям HTML. HTML обеспечивает простые возможности форматирования и обработки форм, управление шрифтами, отображение информации в табличном виде, гипертекстовые связи и поддержку для загрузки и выполнения Java-апплетов. Текущая версия HTML — 3.2.
- **Мультимедийная информация** — может отображаться в браузерах либо в форматах **GIF** или **JPEG**, либо с помощью специальных встраиваемых приложений, которые устанавливаются на клиентскую машину и вызываются на выполнение браузером (helpers, plug-ins). Другие виды мультимедийной информации также могут воспроизводиться в браузере, например, аудио- и видео-файлы, прямое воспроизведение звука и видео, файлы VRML (Virtual Reality Markup Language).
- **JavaScript** — поддерживается, но существуют различия в реализациях. JavaScript не имеет отношения к языку программирования Java — они связаны между собой разве только начальными буквами в названиях. JavaScript впервые был предложен для использования в браузере Netscape Navigator в качестве средства, обеспечивающего большую интерактивность и больший контроль над Web-страницами, которого нельзя достичь с помощью одного только HTML. Затем JavaScript был стандартизован как ECMAScript и принят на вооружение компанией Microsoft для использования в Internet Explorer (JScript).
- **Java-апплеты** — поддерживаются встроенной виртуальной машиной Java (Java Virtual Machine, JVM), но существуют различия в реализациях. Java-апплет представляет собой выполняемую программу, написанную на языке Java которая выполняется в среде браузера, используя JVM как интерпретатор. Мы не будем углубляться в детали устройства апплетов или среды JVM. Достаточно сказать, что использование Java-апплетов позволяет создавать высоко интерактивные пользовательские приложения для Web-браузеров.

- **Dynamic HTML (DHTML)** — также поддерживается в современных браузерах. Однако, существуют различия в реализациях. DHTML представляет собой усовершенствованную версию HTML которая поддерживает объектную модель документа (Document Object Model), предложенную организацией Internet Engineering Task Force (IETF) в стандарте HTML 4.0. Наряду с превосходными возможностями форматирования, DHTML поддерживает условную логику и динамическое выполнение, которые отсутствовали в HTML 3.2.

Прокси-серверы

Если среда Web простирается за пределы компании в Internet, то для защиты от проникновения в корпоративную сеть извне и исключения возможности доступа к конфиденциальной информации используются прокси-сервера (proxy servers), или брандмауэры (firewalls). Прокси-сервер представляет собой специальный тип Web-сервера, который принимает запросы от пользователей Web-браузеров, избирательно перенаправляет эти запросы на обработку Web-серверами внутренней сети и возвращает результаты выполнения запроса пользователю в окно браузера. Избирательность прокси-сервера, его «разборчивость в связях» позволяет не допустить неизвестных пользователей во внутреннюю сеть, а также преградить доступ по неразрешенным коммуникационным методам.

Технологии разработки Web-приложений

В данном разделе описываются существующие средства и технологии, необходимые для разработки Web-приложений. Языки программирования и языки описания содержимого, компонентные модели, протоколы передачи данных, системы интегрированной разработки — все это Вам в той или иной мере потребуется при создании Web-приложений. Каждый из элементов перечисленных категорий имеет свои уникальные особенности, позволяющие его использовать в различных ситуациях при реализации различных аспектов Вашего приложения.

Языки

Языки, которые Вы используете при разработке Web-приложений, оказывают прямое влияние на спектр возможностей создаваемого Web-приложения. Скорее всего, Вы будете использовать не один, а сразу несколько языков из тех, что перечислены ниже, так как они предназначаются для различных целей и ситуаций.

- **HTML/DHTML** — интерпретируемый язык, позволяющий управлять визуальными атрибутами отображаемой информации.
- **JavaScript** — также интерпретируемый язык который можно использовать для придания клиентскому пользовательскому интерфейсу большей интерактивности по сравнению с HTML.

- **Java** — объектно-ориентированный язык программирования, который используется для создания приложений, работающих в рамках Web-браузера (на базе JVM) и характеризующихся наиболее высоким уровнем интерактивности.
- **Visual Basic (VB)** — разработанный Microsoft язык программирования, который легок в использовании и позволяет быстро создавать пользовательские приложения. Visual Basic существует только на платформах Microsoft Windows, и может рассматриваться в контексте Web-приложений, которые следуют спецификации ActiveX.
- **SAS** — язык программирования в SAS System, который обеспечивает поддержку как стандартных конструкций процедурного языка, так и возможность вызова аналитических подпрограмм. Программное обеспечение SAS включает также макроязык, позволяющий текстуально генерировать код в процессе предварительной компиляции.
- **SCL (SAS Component Language)** — язык, разработанный SAS Institute и обеспечивающий доступ к вычислительным возможностям серверов SAS с помощью компонентной архитектуры. Может использоваться для разработки приложений в распределенных средах, включая Web.
- **SQL (Structured Query Language)** — стандартный язык запросов для извлечения информации из реляционных баз данных. Знание языка SQL позволяет легко и быстро разрабатывать Web-приложения, которые генерируют динамические отчеты на основе как данных SAS, так и данных других реляционных СУБД таких, как Microsoft Access, DB2®, Oracle®, Sybase и Informix.
- **htmSQL** — язык разметки, являющийся частью программного обеспечения SAS/IntrNet™. htmSQL позволяет встраивать в HTML страницы конструкции языка SQL. Текст htmSQL обрабатывается CGI программой, которая взаимодействует с сервером SAS. Эта программа передает SQL запросы серверу SAS, форматирует результаты выполнения запроса и передает сформированную страницу с данными в браузер.
- **XML (eXtensible Markup Language)** — недавнее добавление к списку языков, используемых в области Web-приложений. XML дополняет большинство других языков, так как его основное предназначение — структурирование и передача информации в Web. HTML, к примеру, обеспечивает управление визуальными атрибутами Web-страниц, в то время как XML, напротив, обеспечивает управление содержимым в отношении данных, как только Web-страница передана в браузер. XML включает в себя спецификацию HTML, и, также как и HTML, построен на основе языка SGML (Standard Generalized Markup Language) — этом прародителе всех языков разметки.
- **PERL** — отличный язык программирования для разработки CGI скриптов. PERL не является коммерческим продуктом, он свободно доступен для использования. Создание программ не вызывает особых сложностей.
- **C/C++** — эти языки могут использоваться для решения наиболее трудных и сложных программистских задач, однако здесь мы упоминаем их только в контексте написания CGI программ. Если эффективность выполнения программ является первостепенным критерием при разработке, то использование C/C++ является наилучшим вариантом. Однако, необходимо отметить, что использование этих языков требует наличия более полных знаний и большего опыта программирования, чем при работе с другими языками, упомянутыми в данном списке.

Компонентные модели

Языки, выбранные Вами для разработки приложений, определяют подход к возможной компонентной архитектуре. Использование компонент при разработке приложений является удобным способом выделения в модули различных функций, которые требуются пользователям для получения результатов. Кроме того, модули, разработанные с учетом стандартных прикладных программных интерфейсов (API) могут быть повторно использованы в приложениях, требующих сходной функциональности.

На сегодняшний день существует три распространенные компонентные архитектуры для Web-приложений.

- **JavaBeans** — если Вы разрабатываете программы на языке Java, то следование спецификации JavaBeans позволит использовать Вашу программу в качестве компоненты, способной взаимодействовать с другими компонентами. Подобная организация приложений является с точки зрения разработчика нежестко связанной, а с точки зрения пользователя приложение является высоко интегрированным. Усилия, направленные на то, чтобы привести спецификацию JavaBeans к масштабу предприятия, стали первым шагом к созданию Enterprise JavaBeans (EJB). Спецификации EJB обещает стать значительным достижением в процессе унификации объектно-ориентированных протоколов, особенно по причине того, что Object Management Group (OMG) выбрала EJB в качестве шаблона для спецификации CORBA 3.0. Так что совместимость со стандартом CORBA означает совместимость со стандартом EJB, и наоборот, что является очень привлекательным для разработчиков, создающих многоплатформные приложения.
- **COM/DCOM/ActiveX** — если Вы разрабатываете приложения либо на C/C++, либо на VB (Microsoft), Вы можете сохранять разработанные модули как компоненты, которые могут быть использованы в среде браузера Microsoft Internet Explorer. Эти компоненты называются элементами управления ActiveX, и позволяют реализовать дополнительную функциональности и интерактивность в рамках браузера (только на платформе Windows).

- **Remote Object Class Factory (ROCF)** — технология, встроенная в продукт компании SAS Institute — AppDev Studio™, интегрированную среду разработки на Java, которая позволяет создавать динамические связи Java-апплетов и приложений для доставки информации с объектами SCL, выполняющимися на сервере SAS. Использование ROCF в качестве компонентной архитектуры облегчает миграцию некоторых SCL приложений для использования в Web.

Протоколы передачи данных

Web-приложения нередко используют данные, хранящиеся в различных СУБД, в частности, для динамического формирования отчетов. На сегодняшний день существует несколько основных протоколов, использование которых покрывает нужды большинства Web-приложений, связанные с необходимостью организации доступа, извлечения, изменения и пополнения новой информацией баз данных.

- **Протокол ODBC (Open Data Base Connectivity)** — разработанный Microsoft стандартный протокол, позволяющий приложениям взаимодействовать с реляционными СУБД. Чтобы понять, что такое ODBC, представьте себе провод, соединяющий Ваше приложение с базой данных, по которому в одном направлении передаются SQL запросы, а в обратном — данные, являющиеся результатом обработки указанных запросов базой данных.
- **Протокол JDBC (Java Data Base Connectivity)** — стандартный протокол, позволяющий апплетам и приложениям, написанным на Java, взаимодействовать с реляционными СУБД. Мост JDBC-ODBC, являющийся частью спецификации Java, позволяет сервлету Java использовать для доступа к данным протокол ODBC. Кроме того, поставщики некоторых СУБД могут предоставлять средства для прямого взаимодействия с базой данных по протоколу JDBC. Функционально JDBC аналогичен протоколу ODBC, но поскольку JDBC основан на Java, он является платформенно-независимым.
- **Протокол SAS/SHARE®** — протокол многопользовательского сервера данных SAS/SHARE для работы с данными SAS System. Запросы клиентов обрабатываются синхронно, в порядке поступления. В комбинации с продуктом SAS/IntrNet, сервер SAS/SHARE обеспечивает доступ к данным SAS со стороны клиентов ODBC, JDBC и htmSQL. Для приложений, создаваемых на базе программного обеспечения SAS, SAS/SHARE является наиболее эффективным средством обработки многочисленных SQL запросов.
- **Протокол SAS/CONNECT®** — протокол для синхронного или асинхронного взаимодействия с сервером SAS, предусматривающего доступ как к данным, так и к вычислительным возможностям SAS System. В комбинации с продуктом SAS/IntrNet, SAS/CONNECT позволяет Web-приложениям на базе HTML и клиентам Java устанавливать двустороннюю связь с сервером SAS.

Системы интегрированной разработки

И наконец последнее, что Вам потребуется для разработки Web-приложений — это интегрированная среда разработки. До сих пор, правда, находятся «настоящие» программисты, которые пишут и отлаживают программы, используя обыкновенные текстовые редакторы. Но таких людей становится все меньше и меньше, потому как гораздо удобнее использовать специальные интегрированные системы разработки (Integrated Development Environment, IDE), в частности, для языков, протоколов и компонентных моделей, перечисленных выше.

Данная статья не предполагает подробный обзор существующих систем разработки. Ниже приводится список наиболее популярных систем с описанием их возможностей для создания Web-приложений.

- **Java IDE** — это целая группа систем разработки для создания апплетов и приложений на Java. Некоторые из них написаны на Java, некоторые являются версиями изначально существующих систем IDE, переработанными для поддержки Java. Некоторые сами написаны на языке Java, некоторые написаны на C или C++ и просто генерируют код на Java как результат работы. Большинство поставщиков систем этой категории предоставляют визуальные среды разработки, которые позволяют создавать сложные Java-программы как опытным разработчикам, так и новичкам. Лидерами на рынке подобных систем являются:
 - Visual Cafe — Symantec
 - Jbuilder — Inprise (ранее Borland)
 - VisualAge™ — IBM
 - Supercede — Supercede
 - AppDev Studio™ — SAS Institute
 - Visual J++ — Microsoft
- **Visual Basic (VB)** — Компания Microsoft является единственным поставщиком, обеспечивающим поддержку VB, однако, это один из наиболее популярных языков программирования для персональных компьютеров. Подобно системам разработки на Java, упомянутым в предыдущей секции, Visual Basic является визуальной интегрированной средой разработки и в значительной степени защищает программиста от необходимости знать синтаксис языка.
- **SAS/AF®** — среда разработки от SAS Institute, используемая для разработки программного обеспечения в рамках SAS System. SAS/AF использует язык программирования SCL и позволяет создавать объекты и приложения, которые способны выполняться в распределенных средах и обеспечивать доступ к серверам SAS.
- **SAS/EIS®** — визуальная среда разработки от SAS Institute, которая содержит ряд готовых классов, написанных на SCL, и позволяет собирать на их основе приложения для OLAP анализа.

¹ *Design Patterns*, Ehrich Gamma, Richard Helm, Ralph John Vlissides: Addison-Wesley, 1995

- **C/C++** — На рынке существует несколько поставщиков сред разработки на C/C++, некоторые из них упомянуты в секции, описывающей системы разработки на Java. Если Ваши сотрудники обладают достаточными знаниями и опытом, C/C++ может быть приемлемой альтернативой Java, Visual Basic и SCL в области разработки Web-приложений. На C/C++ могут быть написаны элементы управления ActiveX. На C/C++ также могут быть написаны программные модули, взаимодействующие с SAS System и обеспечивающие дополнительную функциональность (в этом случае используется продукт SAS/TOOLKIT™). Программы на C/C++ могут также взаимодействовать с компонентами Java.

Выбор существующих технологий под нужды Вашего приложения

Теперь Вы знаете какие продукты, компоненты и технологии требуются для создания Web-приложений. Следующим шагом является выработка сценариев использования всех этих средств, что позволило бы Вам значительно эффективней применять Ваши знания и обеспечить наибольшую отдачу от предполагаемых разработок. Для описания сценариев мы будем использовать подход, предложенный Эрихом Гамма (Erich Gamma) и его соавторами в их замечательной книге *Design Patterns*¹.

Под сценарием понимается описание характерных, часто встречающихся проблем и изложение основных способов решения этих проблем в различных ситуациях. В контексте данной статьи все сценарии предусматривают использование различных средств для создания Web-приложений на базе программного обеспечения SAS. При описании сценариев мы будем следовать схеме, предложенной в упомянутой книге, и рассматривать следующие аспекты:

- **Цель (Intent)** — описание проблемы, решаемой данным сценарием
- **Обоснование (Motivation)** — краткое описание того, как данный сценарий может решить проблему.
- **Применимость (Applicability)** — описание ситуаций, в которых данный сценарий может быть задействован.
- **Компоненты (Participants)** — список элементов, составляющих основу сценария.
- **Способы взаимодействия (Collaborations)** — описание способов взаимодействия компонент.
- **Реализация (Implementation)** — перечень потенциально возможных проблем, с которыми можно столкнуться при реализации сценария.
- **Примеры (Samples)** — фрагменты реальных приложений, демонстрирующих использование сценария.

Сценарий №1 — Web-публикации

Цель

Вы хотите разместить на Web-сервере статическую информацию, которая будет доступна пользователям через Web-браузер. Отметим, что эта информация является статической только с точки зрения пользователя Web-браузера. На самом деле, она может регулярно обновляться или изменяться в соответствии с намеченным графиком. Например, каждую ночь или еженедельно в рамках пакетного производственного задания могут запускаться программы SAS, которые создают ряд таблиц или отчетов. Использование данного сценария позволяет передать сгенерированные отчеты любому пользователю, работающему с Web-браузером.

Обоснование

Если у Вас есть информация, которая изменяется не слишком часто и которую требуется распространять среди широкого круга пользователей, то Вам подходит именно этот сценарий. В соответствии с данным сценарием Вы создаете на основе имеющейся информации статические Web-страницы и публикуете их на Web-сервере, в результате чего любой пользователь, у которого есть Web-браузер и который имеет доступ к Вашему Web-серверу, сможет просматривать или распечатывать эти страницы.

Применимость

Пользователи получают доступ к Вашей информации через Web-браузер. Никакое специализированное программное обеспечение не должно дополнительно устанавливаться на персональные компьютеры пользователей.

Компоненты

- **Web-браузер** — программа для просмотра Web-страниц, которые она скачивает с Web-сервера.
- **Web-сервер** — программный процесс, который принимает запросы на доступ к информационным ресурсам от пользователей Web-браузеров и пересылает им необходимые файлы для просмотра.
- **Генератор содержимого Web-сервера (content generator)** — это ключевой элемент в данном сценарии, поскольку именно содержимое Web-сервера, или его информационное наполнение, определяет ценность приложения. Простейшим примером генератора содержимого Web-сервера является человек, который создает HTML файлы и размещает их в директории Web-сервера. С другой стороны, генератором содержимого может быть программное обеспечение SAS с его мощными возможностями в отношении систем поддержки решений и хранилищ данных. Результаты работы любой программы SAS могут быть преобразованы в формат HTML и сохранены в виде файлов в директории Web-сервера. Таким образом, генератором содержимого может быть пакетный или фоновый процесс выполнения программ SAS, создающих необходимые файлы для Web-сервера. После того, как будут созданы все файлы, пользователи смогут обращаться к ним из окна Web-браузера.

• **Механизм передачи файлов** (file transport mechanism) — взаимодействие между Web-браузером и Web-сервером осуществляется с помощью стандартного протокола HTTP, предусматривающего также возможности передачи файлов, поэтому нет необходимости подробно на этом останавливаться. То, что заслуживает рассмотрения, — это способы доставки файлов от генератора содержимого на Web-сервер. Довольно часто для этих целей применяется протокол FTP, но Вы можете использовать и ряд других способов.

Способы взаимодействия

Генератор содержимого создает файлы в формате, пригодном для использования в контексте Web, и, используя механизм передачи файлов, размещает их в директориях, которые доступны Web-серверу. Пользователь, работающий с Web-браузером, может обратиться с запросом на эти файлы к Web-серверу и просмотреть их в окне браузера. Наиболее общей и удобной формой распространения информации в Web являются файлы в формате HTML, которые чаще всего и имеют в виду, когда говорят о Web-страницах. HTML файлы могут содержать текстовую информацию, таблицы, графические изображения в форматах GIF и JPEG и гиперссылки. Графические изображения могут использоваться в качестве либо фоновых рисунков, либо декоративных элементов, либо элементов для навигации. Кроме того, графика может использоваться в качестве собственно содержимого, позволяя передавать информацию в виде картинки, которая, согласно пословице, «стоит тысячи слов». С помощью гипертекстовых ссылок пользователи могут перемещаться по различным частям одной страницы или переходить на другие страницы в поисках нужной им информации. Удачная схема гипертекстовых связей позволяет избавить посетителей Вашего Web-сервера от необходимости просматривать лишнюю информацию.

Реализация

При реализации сценария Web-публикации следует обратить внимание на следующие моменты:

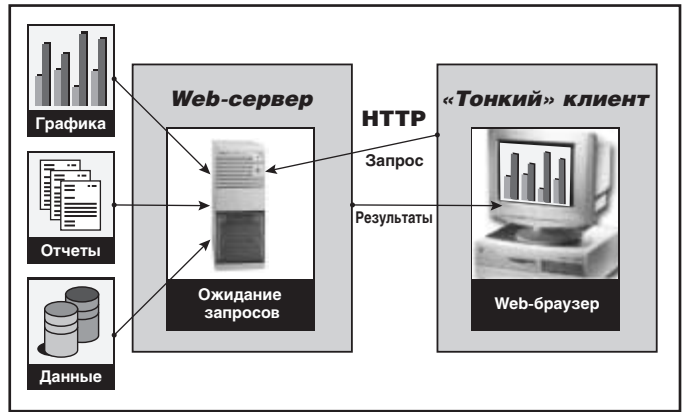


Рис. 1. Содержимое Web-сервера.

1. Частота обновления информации должна отвечать ожиданиям пользователей. Если пользователям необходимо, чтобы информация обновлялась часто (например, чаще чем раз в сутки), то Вам возможно следует использовать следующий сценарий — сценарий динамической генерации содержимого.
2. Если Вы не знаете какой браузер будут использовать посетители Вашего Web-сервера, то при проектировании структуры и оформления Web-страниц Вам следует ориентироваться на браузер с наиболее скромными возможностями. Весьма полезно также разместить на страницах ссылку «MAILTO:» с адресом Вашей электронной почты, чтобы иметь обратную связь с пользователями, которые при необходимости сообщат Вам, что им нравится или не нравится в оформлении Ваших Web-страниц.

Примеры

Начиная с версии 7.0 система SAS содержит механизм унифицированного вывода результатов ODS (Output Delivery System). ODS позволяет легко и быстро сохранять результаты работы программ SAS в форматах, пригодных для использования в Web. Данный пример показывает как можно опубликовать простой отчет.

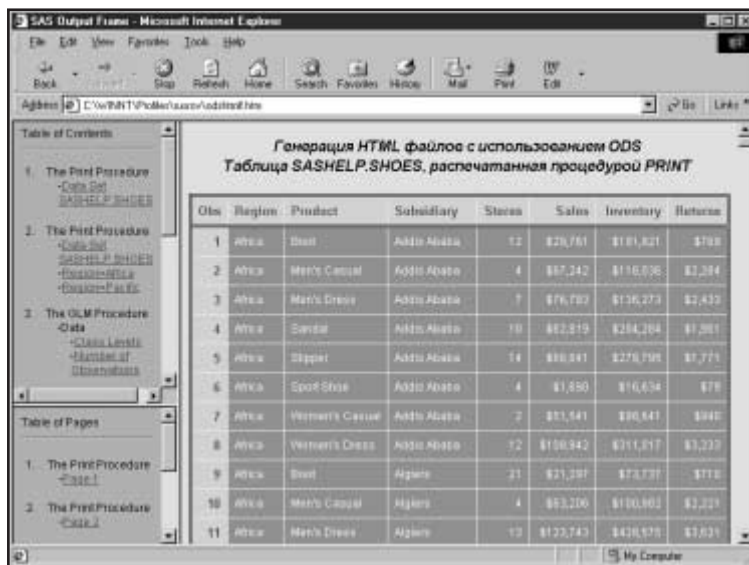


Рис. 2. Исходный код и полученные результаты.

Сценарий №2 — Динамическая генерация содержимого

Цель

Данный сценарий предусматривает динамическую генерацию содержимого в ответ на запрос пользователя и передачу результатов в браузер для просмотра.

Отличие от предыдущего сценария Web-публикаций состоит лишь в том, что момент генерации содержимого соотнесен с моментом поступления запроса. В сценарии Web-публикаций содержимое создается асинхронным процессом, то есть процессом, который создает необходимые файлы заранее, до поступления запросов пользователей. В сценарии динамических приложений процессы создания содержимого и передачи результатов для просмотра пользователем жестко связаны друг с другом, или можно сказать, выполняются синхронно.

Обоснование

Возможность генерации отчетов по запросам значительно расширяет диапазон ожиданий пользователей в отношении Web-приложений. Описываемый подход позволяет пользователям производить произвольные, нерегламентированные запросы и осуществлять эвристический поиск информации. Это расширяет пользовательское восприятие Web-приложения и возможности его применения и обеспечивает большую гибкость в отношении приложений по извлечению данных и формированию отчетности, что исключает необходимость напрямую обращаться к администраторам данных или разработчикам приложений.

Применимость

Вы хотите дать пользователям возможность манипулирования информацией из окна Web-браузера. Одной из основных причин этого является желание исключить ресурсоемкий этап предварительной подготовки данных, то есть процесс пересоздания на основании данных из репозитория всевозможных комбинаций статических Web-страниц, которые потенциально могут потребоваться пользователям.

Данный сценарий (называемый также сценарием распространения отчетов) подходит в тех случаях, когда Вы хотите предоставлять пользователям именно ту информацию, которая им нужна и которая извлекается из свежих, актуальных, доступных данных.

Компоненты

- **Компоненты сценария Web-публикаций** — все компоненты из первого сценария, включая Web-браузер, Web-сервер, механизм передачи файлов и генератор содержимого Web-сервера. В данном сценарии Web-сервер получает дополнительную функциональность — наряду с передачей файлов, Web-сервер может инициировать программные процессы через интерфейс CGI или через специализированные интерфейсы для того, чтобы обслуживать более сложные запросы. В этом случае генератор содержимого динамически взаимодействует с браузером, выступая в роли либо машины обработки запросов, либо машины выполнения вычислений.

- **Машина обработки запросов** (query processor) — генератор содержимого, который принимает SQL запросы как входные параметры и выдает форматированные результаты выполнения этих запросов в качестве выходных параметров. В составе SAS System средством обработки SQL запросов является продукт SAS/SHARE. Компонента htmSQL и SAS/SHARE драйвер для Java, входящие в состав продукта SAS/IntrNet, позволяют передавать SQL запросы с Web-сервера серверу SAS/SHARE.

- **Машина выполнения вычислений** (compute processor) — генератор содержимого, который способен принимать на вход произвольные программы и команды и выдавать на выход форматированные результаты. Основное различие между машиной обработки запросов и машиной выполнения вычислений состоит в том, что в одном случае мы ограничены рамками языка SQL при описании действий, в то время как, в другом случае мы можем использовать произвольные программы на том языке, с которым работает машина выполнения вычислений. Компонента Application Dispatcher, входящая в состав SAS/IntrNet, является примером машины выполнения вычислений для Web-приложений, работающих в технологии CGI. Application Dispatcher позволяет пользователям Web-браузеров обращаться с запросами к серверу SAS, который способен выполнять произвольные программы, написанные на языке программирования SAS, и обращаться ко всему спектру возможностей SAS System.

- **Репозиторий данных** — сетевой ресурс для хранения данных, доступ к которому имеет машина обработки запросов или машина выполнения вычислений. Данные могут храниться в SAS System как наборы данных (data set) или как многомерные базы данных (Multi-Dimensional Data Base, MDDB), а также могут храниться, к примеру, в таблицах баз данных DB2® или Oracle®, или в любом другом формате.

Способы взаимодействия

Пользователь обращается за информацией к Web-серверу, указывая адрес URL, который содержит встроенный запрос или обращение к программе. Указанный запрос или программа используются для извлечения данных из репозитория данных с помощью либо машины обработки запросов, либо машины выполнения вычислений. Результаты выборки или работы программы форматируются и передаются назад в браузер пользователя через механизм передачи файлов.

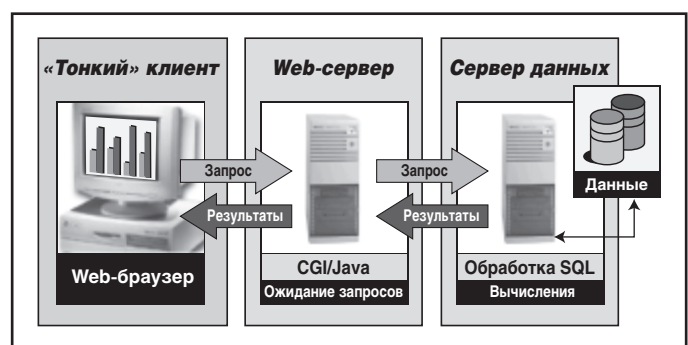


Рис. 3. Схема взаимодействия компонент при динамической генерации содержимого.

Как и в случае Web-публикаций, динамическая генерация содержимого основана на создании файлов тех форматов, которые подходят для использования в Web-браузере, так что основными форматами для этого сценария являются HTML и графические форматы. Кроме того, при использовании динамических процессов для создания содержимого появляются две новые дополнительные компоненты:

- *Механизм запуска процессов* (Process spawners) — механизм, позволяющий Web-серверу запустить новый процесс для обработки пользовательского запроса. Подобные механизмы предусмотрены в различных технологиях — CGI, специализированных прикладных интерфейсах (API), таких как ISAPI и NSAPI, технологиях Java сервлетов и Active Server Pages.
- *Распределенные средства просмотра* (distributed content viewers) — апплеты Java и элементы управления ActiveX.

Механизмы запуска процессов

Использование механизмов запуска процессов является в настоящее время наиболее широко распространенным и удобным способом генерации динамических отчетов и доставки их через Web. Это механизмы просты, надежны и достаточно эффективны в смысле накладных расходов процессорного времени и требований к полосе пропускания.

Все четыре упомянутые технологии обладают способностью инициировать программные процессы на сервере, правда для этого используются различные методы. CGI является наиболее предпочтительной технологией для реализации Web-приложений, которые могут выполняться на компьютерах различных производителей. Например, компонента htmSQL продукта SAS/IntrNet представляет собой небольшую CGI программу, которая может работать практически на любом Web-сервере, так как практически любой Web-сервер поддерживает стандарт CGI.

Два специализированных прикладных интерфейса для создания Web-приложений — ISAPI и NSAPI — преимущественно используются в приложениях, ориентированных на большой поток запросов транзакционного типа, оперирующих небольшими объемами данных и имеющих тенденцию выполняться помногу раз. Использование этих API может несколько повысить эффективность транзакционного приложения, однако это приложение будет переносимым и масштабируемым только для Web-серверов того же производителя (реализующих тот же самый API). SAS/IntrNet не обеспечивает поддержки для таких специализированных прикладных интерфейсов, так как их использование приводит к потере переносимости, которая не может быть компенсирована тем незначительным выигрышем в производительности по сравнению с оптимизированным решением на базе CGI.

Java сервлеты похожи на приложения CGI, за исключением того, что они создаются с использованием Java и требуют для выполнения виртуальную машину Java (JVM). Если Web-сервер поддерживает тех-

нологию Java сервлетов, то JVM обычно уже включена в состав Web-сервер, что облегчает процесс разработки. Учитывая то, что в спецификацию CORBA 3.0 были добавлены стандарты модели компонент JavaBeans совместно со стандартами CORBA, Java сервлеты будут становиться все более популярным средством создания платформно-независимых приложений для работы на стороне сервера. Поддержка сервлетов в программном обеспечении SAS будет добавлена в рамках SAS System версии 8.

Active Server Pages (ASP) — это стандарт Microsoft для разработки серверных приложений для Web. Файлы ASP являются выполняемыми модулями на Windows и могут обращаться к другим службам, которые доступны на сервере, например, к Microsoft Transaction Server, если требуется поддержка транзакций, или к MSMQ, серверу очередей сообщений Microsoft для создания содержимого с использованием асинхронных процессов. Поддержка ASP в программном обеспечении SAS будет добавлена в рамках SAS System версии 8.

Распределенные средства просмотра

Java-апплеты представляют собой пакеты с выполняемым кодом, которые могут быть доставлены на Web-браузер. Для этого, HTML страница должна содержать специальные теги для встраивания Java-апплетов. Когда браузер загружает подобную страницу, то он сразу же инициирует процесс перекачки с Web-сервера файлов с кодом апплета. Затем загруженный апплет начинает выполняться в контексте Web-браузера, используя в качестве среды выполнения виртуальную машину Java (Java Virtual Machine, JVM), которой оснащаются современные браузеры.

Элементы управления ActiveX по функциональности похожи на Java-апплеты, но они существуют только для клиентов на базе Windows. Элементы ActiveX могут также резидентно сохраняться на клиентской машине. Эта возможность весьма привлекательна для реализации процесса распространения программного обеспечения в корпоративных сетях, однако подобные схемы распространения выполняемого кода следует использовать с осторожностью и придерживаться строгих ограничений безопасности. Элементы ActiveX, выполняясь на клиентской машине, могут обращаться к системным службам, ресурсам и периферийным устройствам. Так что это палка о двух концах, так как тот же самый элемент управления, который может выводить информацию на принтер, может и удалять файлы с локального жесткого диска.

Реализация

При реализации сценария динамической генерации содержимого следует обратить внимание на следующие моменты:

1. Важно найти правильный баланс между возможностями генератора содержимого и возможностями средств просмотра. К примеру, отчет в тысячи строк, скорее всего, будет удобнее просматривать по частям или распечатать его на бумаге, чтобы изучить во время полета на встречу с Вашими заокеанскими партнерами.

2. Следует минимизировать количество шагов, которые приводят пользователя к желаемому отчету. Использование слишком большого числа параметров может снизить эффективность распространения информации вследствие неизбежных временных задержек.
3. Машина обработки запросов, использующая язык SQL, представляет собой прекрасный обобщенный механизм для обработки множества поступающих запросов в отчеты. Программный код SQL может генерироваться приложением, изолируя пользователя от необходимости знать конструкции SQL. Компонента htmSQL и драйвер SAS/-SHARE для Java, входящие в состав SAS/IntrNet, в сочетании с сервером SAS/SHARE обеспечивают великолепные возможности по обработке запросов для приложений SAS, работающих через Web.
4. Машины выполнения вычислений, такие например как SAS System, работающая в режиме сервера, имеют широкий спектр средств для организации доступа практически к любым данным, для выполнения любых преобразований над данными, для создания произвольных отчетов и для сохранения этих отчетов в любом формате, используемом в Internet, например, HTML, GIF, JPEG, CSV, и т.д.
5. Для повышения гибкости и эффективности использования Web-приложений разрабатывайте и используйте те генераторы содержимого, которые обеспечивают достаточно высокий уровень надежности и позволяют динамически создавать отчеты со встроенными гиперссылками (также создаваемыми динамически), которые позволяют вызывать из созданного отчета другие динамические отчеты.

Примеры

В качестве примера мы рассмотрим приложение Xplore, одно из приложений в составе пакета продуктов SAS/IntrNet. Xplore представляет собой Web-приложение для просмотра содержимого библиотек и каталогов, существующих в сессии SAS, и выполнения над элементами библиотек и каталогов действий, зависящих от типа элемента.

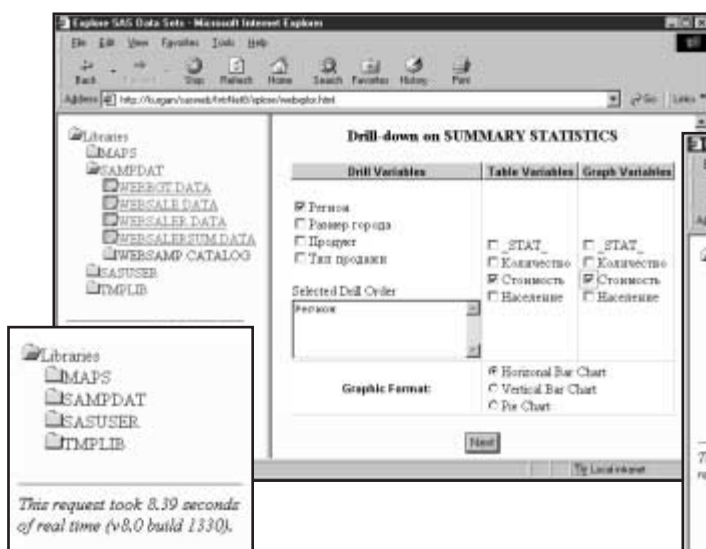


Рис. 4. Список библиотек.

Интерфейс приложения организован в виде двух фреймов: в левом фрейме отображается структура библиотек и каталогов SAS, а в правом — информация о выбранном элементе. На рис. 4 показан список библиотек, которые были назначены в текущей сессии SAS. Пользователь может произвести двойной щелчок правой кнопкой мыши по пиктограмме папки и просмотреть содержимое библиотеки или каталога. (Пиктограмма папки содержит в нижнем правом углу красный кружок, по которому можно понять, что папка представляет библиотеку или каталог SAS, а не директорию операционной системы).

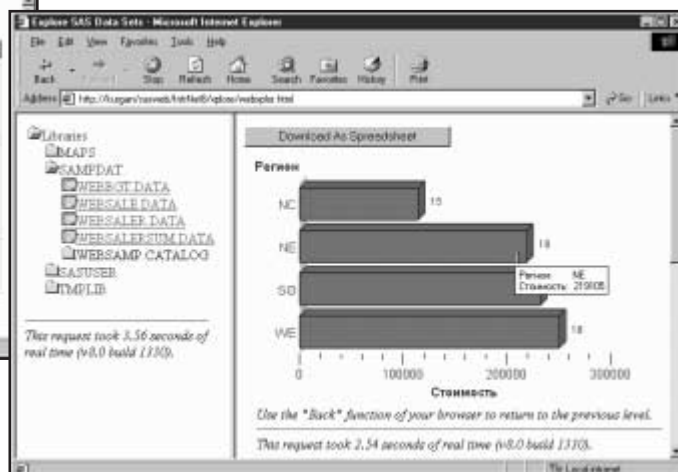
Различные типы элементов в библиотеках и каталогах SAS имеют различные пиктограммы, и с каждым типом элемента связано некоторое действие, определяющее поведение Xplore при выборе конкретного элемента.

Например, пользователь может щелкнуть мышкой по пиктограмме набора данных SAS, тогда в правой части экрана будут показаны записи этого набора данных. Для разделов каталога с графической информацией, в правой части экрана отображается рисунок, для текстовых разделов — текст, и т.д.

Xplore имеет специальное действие для наборов данных, являющихся результатом работы процедуры SUMMARY — сначала определяются классифицирующие и анализируемые переменные, затем отображается Web-страница, в которой пользователь может указать в каком виде (табличном и/или графическом) он хочет просматривать информацию и задать иерархию сверления, выбрав несколько классифицирующих переменных, и наконец, получить страницу с интерактивным отчетом, соответствующим заданным параметрам. Рис. 5 показывает сгенерированную Web-страницу, в которой пользователь задает параметры отчета. Нажатие кнопки SUBMIT, присутствующей на этой странице, приводит к генерации страницы с отчетом, показанной на рис. 6. Для просмотра графики может использоваться Java-апплет, который поддерживает операции сверления в соответствии с иерархией, определенной пользователем, для просмотра более детальных данных.

Рис. 5. Сгенерированная Web-страница.

Рис. 6. Сгенерированная страница с отчетом.



Сценарий №3 — Распространение приложений

Цель

Вы должны распространять выполняемые приложения по требованию пользователей, при этом необходимо обеспечить централизованное управление процессом разработки и поддержки приложений.

Обоснование

Пользователи приложений становятся все более разбросанными и удаленными от центральных ИТ ресурсов. Чтобы облегчить поддержку распределенных приложений, ИТ переходит от развертывания посредством инсталляций программного обеспечения, к развертыванию посредством перекачки файлов по методу «выталкивания» («push»), либо по методу «вытягивания» («pull»). Хотя это не входит в рассматриваемый сценарий, многие из вопросов, связанных с общей стоимостью владения (total cost of ownership, TCO), решаются с использованием методологий конфигурационного управления, которые так или иначе сгруппированы вокруг комбинаций методов «выталкивания» или «вытягивания» при инсталляции и поддержке программного обеспечения. Рассматриваемый сценарий является отражением этих методологий, в том смысле что его использование позволяет удовлетворять зачастую временные нужды пользователей персональных компьютеров вместо создания группы администраторов. Метод «выталкивания» является активным и помещает данные и выполняемые приложения на клиентские машины заранее, в то время как метод «вытягивания» является пассивным и помещает данные и приложения на клиентские машины только по требованию пользователя.

Данный сценарий предусматривает использование обоих методов, однако ради краткости изложения в данной статье рассматривается только метод «вытягивания».

Применимость

Вы хотите дать пользователям возможность выполнять приложения, запуская их из окна браузера.

Компоненты

- **Компоненты сценария Web-публикации** — все компоненты сценария №1, включая Web-браузер, Web-сервер, механизм передачи файлов и генератор содержимого. В данном сценарии, генератор содержимого, или некоторая его часть, может быть перенесена в браузер или на клиентский компьютер, где и происходит процесс генерации содержимого.
- **Механизм «выталкивания»/«вытягивания» (Push/Pull mechanism)** — Механизм выталкивания представляет собой вещательный элемент в сети, который определяет компьютеры по их IP адресам, собирает информацию о конфигурации приложений и самостоятельно обновляет программное обеспечение, если имеется более новая версия. И все это происходит без взаимодействия с пользователем. Механизм вытягивания является более пассивным и только извещает пользователя о наличии новых данных или нового прикладного обеспечения. Комбинация механизмов выталкивания и вытягивания часто называют циклом публикации/подписки, в котором генератор содержимого посылает широковещательное сообщение о наличии новой информации или приложений, и любой подписчик, настроенный на нужную волну, автоматически получит новую информацию. В этом случае процесс генерации содержимого по-прежнему пассивен, однако пользователи могут активировать его автоматически, подписавшись на этот процесс.
- **Репозиторий приложений (Application repository)** — Репозиторий приложений является логическим дополнением репозитория данных. Он предоставляет возможности хранения и извлечения приложений. Управление подобным репозиторием может варьироваться от простейшего варианта, в котором используются каталоги и наборы данных SAS, до гораздо более сложных. Более сложные режимы управления обычно используют либо системы управления базами данных для хранения и извлечения информации о месторасположении программ, либо если приложение использует объектно-ориентированный подход, либо брокер объектов запросов (object request broker), возможно в сочетании и сервером директорий (directory server) для поиска объектов, которые могут быть разбросаны по всей сети Web.

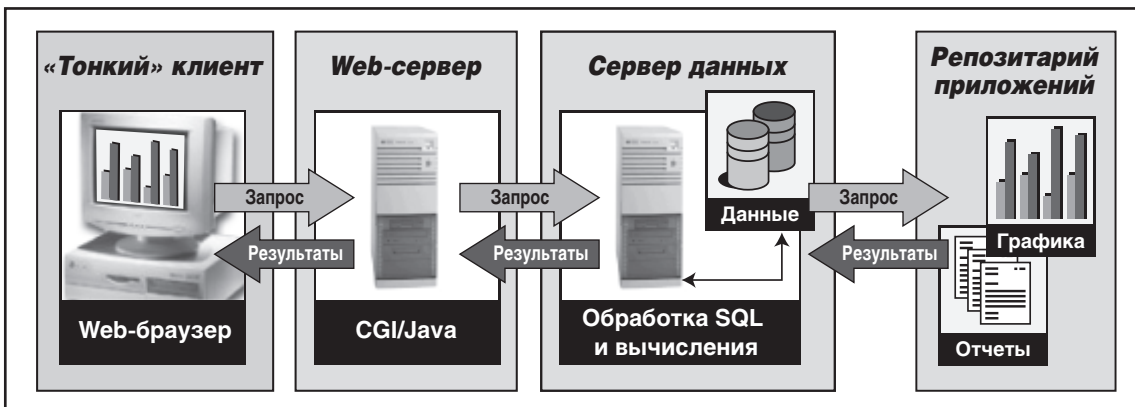


Рис. 7. Схема взаимодействия компонент в сценарии распространения приложений.

Способы взаимодействия

Пользователь делает выбор в окне браузера, что заставляет Web-сервер инициировать запрос на содержимое. Обработка этого запроса может происходить на самой клиентской машине, для чего Web-сервер обеспечивает доставку на машину клиента выполняемых приложений и на этом прекращает обработку запроса. Обработка этого запроса может также потребовать выполнения некоторой программы на Web-сервере или на другой машине, доступной по сети. Репозиторий приложений, к которому обращается Web-сервер, находит запрашиваемую программу и подготавливает ее к выполнению. Доставка содержимого в браузер осуществляется через Web-сервер либо самим репозиторием приложений, либо выполнением приложения в зависимости от организации системы. Как и в двух предыдущих сценариях HTML является основным транспортным механизмом для распространения приложений. Интерактивные приложения могут потребовать использования более сложных инструментов которые могут встраиваться в HTML файл, передаваемый в браузер. Динамический HTML (Dynamic HTML, DHTML), JavaScript, VBScript, Java и элементы ActiveX могут использоваться, для того, чтобы сделать Web-страницу более динамичной и интерактивной.

Нет необходимости передавать на клиентский компьютер приложение в полном объеме, напротив, приложение может быть разделено на части, некоторые из которых выполняются на клиенте, а некоторые на сервере. Эта техника полезна для того, чтобы минимизировать время перекачки файлов на клиентскую машину, и как следствие, минимизировать время необходимое для запуска приложения.

Репозиторий приложений является новой структурой, введенной в данном сценарии. Репозиторий приложений можно представить как библиотеку, из которой клиенты могут выписывать и использовать приложения, а генераторы содержимого регистрировать новые приложения и поддерживать их в актуальном состоянии.

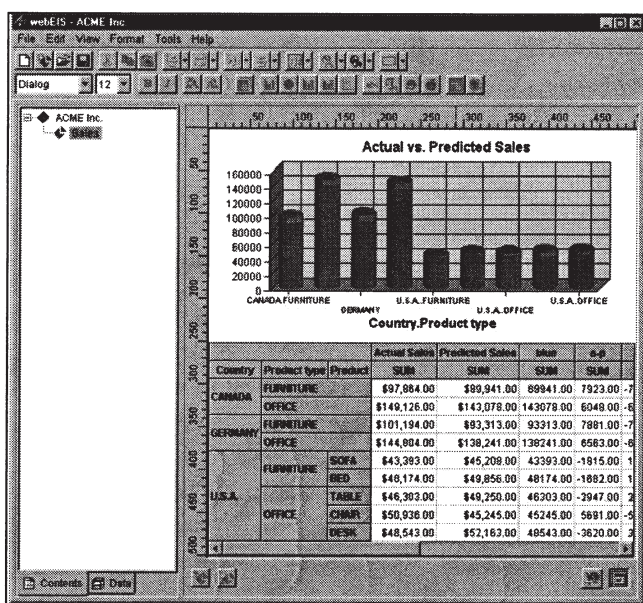


Рис. 8. Распространение программного кода по требованию.

нии. Сложность репозитория зависит от манеры его использования, подходящей для пользователей. В простейшем случае репозитория является директорией на сервере, открытой только на чтение (read-only). Более сложные формы могут включать дополнительные услуги по авторизации пользователей или поиску информации в этой директории. Другие реализации могут использовать очереди сообщений для извещения вычислительных серверов об ожидающих обработки запросах. Вообще говоря, более сложное устройство системы оправдывается необходимостью масштабирования и доступности. Однако, все равно некоторая форма репозитория приложений должна присутствовать для реализации сценария распространения приложений.

Реализация

При реализации сценария распространения приложений следует обратить внимание на следующие моменты:

1. Пропускная способность сети является ограничивающим фактором в любом сценарии, в котором предусмотрено перемещение данных с одного компьютера на другой по сети. В сценарии распространения приложений подразумевается, что задержка, вызываемая перемещением файлов на клиентский компьютер, должна быть минимальной, — и этим нельзя пренебрегать. Если размер файла достаточно велик, скажем, свыше 1 Mb, то мобильные пользователи, имеющие сетевое соединение на 28.8 Kb/s, будут вынуждены ждать несколько минут. Пользователям локальных или глобальных сетей для перекачки приложения того же объема может потребоваться несколько секунд в зависимости от загрузки сети.
2. Важно использовать правильное соотношение между временем скачивания и частотой и продолжительностью использования приложения. Например, время скачивания в 2 минуты для приложения, которое Вы будете использовать в течение продолжительного периода времени (несколько часов или дней) является вполне удовлетворительным. Однако, то же самое время перекачивания файлов для приложения, которое Вам нужно только сейчас и на короткий промежуток времени, может быть неприемлемым.
3. Возможности клиентского компьютера или требования к переносимости приложений могут исключать использование элементов ActiveX или ограничивать применение JavaScript/Jscript. В этой ситуации стоит рассмотреть возможность использования Java, как наиболее жизнеспособный вариант реализации распространения логики приложений.

Примеры

webEIS™ один из продуктов, входящих в состав AppDev Studio, демонстрирует использование рассматриваемого сценария для создания динамического содержимого. Эксплуатация приложений осуществляется с использованием принципов данного сценария. Ради краткости мы пропустим описание шагов, необходимых для создания приложения, и перейдем непосредственно к рассмотрению ситуации, когда кто-либо хочет использовать созданное приложение.

Как и в случае со всеми Web-приложениями мы начинаем с того, что указываем адрес URL в окне браузера. Браузер обращается к Web-серверу и начинает загружать на клиентский компьютер (используя механизм передачи файлов) файлы, содержащие программную логику. В случае апплетов webEIS программная логика реализована в классах Java, собранных и сжатых в файл архива (Java Archive, JAR). После загрузки и инициализации апплета, пользователь получает возможность просматривать информацию через Java-апплет. В нашем примере (рис. 8) апплет использует иерархическую структуру для управления отчетами, которая представлена в виде дерева в левой части экрана, а также таблицы и диаграммы для просмотра информации в правой части экрана. Будучи загруженным в окно браузера, апплет становится как бы независимым и позволяет, к примеру, просматривать другие отчеты в виде таблиц, или менять тип диаграммы или графика.

Основное различие между данным сценарием и сценарием динамического создания содержимого состоит в том, что в сценарии №2 содержимое, отображаемое в окне браузера, создается на сервере программным процессом и передается в браузер для просмотра. В сценарии распространения приложений

Приложение, реализующее процесс генерации содержимого, перекачивается в браузер, и создание содержимого происходит на клиентской машине в самом браузере (или в операционной системе, если генератор содержимого не является апплетом или элементом ActiveX).

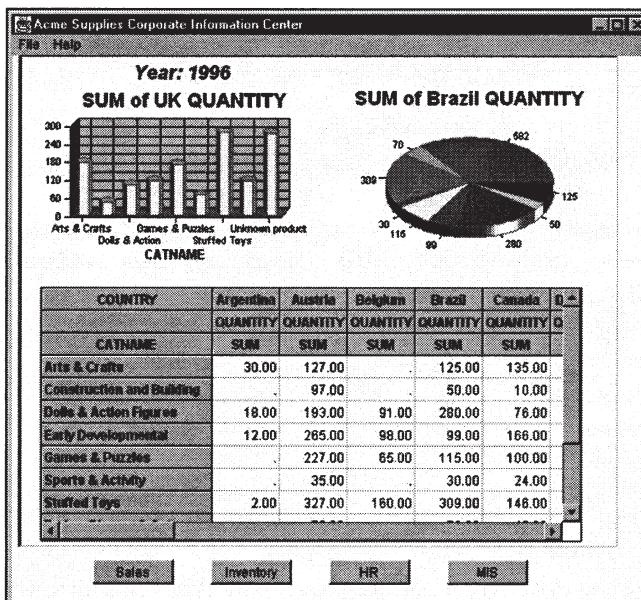


Рис. 9. Пример экрана распределенного приложения.

Продукты, предлагаемые SAS Institute

Подводя итог, перечислим продукты SAS Institute, которые могут использоваться при разработке Web-приложений. В конце списка приведена таблица, которая показывает какие продукты могут использоваться при реализации определенного сценария.

• **Base SAS®** — центральный продукт SAS System. В отношении рассматриваемых в статье вопросов Base SAS включает:

- Репозиторий данных в виде наборов данных SAS
- Репозиторий приложений в виде каталогов SAS
- *Web Publishing Tools* — макросы для использования в сценарии Web-публикаций (версия 6 и выше)
- *Output Delivery System (ODS)* — система унифицированного вывода результатов для использования в сценарии Web-публикаций (версия 7 и выше)

• **SAS/MDDB® Server** — специализированный репозиторий для хранения агрегированных многомерных данных.

• **Семейство продуктов SAS/ACCESS®** — доступ к репозиториям данных других производителей.

• **SAS/SHARE®** — многопользовательский репозиторий данных, работающий с наборами данных SAS, многомерными базами данных SAS (MDDB) или репозиториями других производителей.

• **SAS/CONNECT®** — средство организации клиент/серверного взаимодействия между сессиями SAS, в частности может использоваться как механизм передачи файлов.

• **SAS/IntrNet™** — продукт для создания Web-приложений, работающих на базе SAS System. SAS/Intrnet включает:

- *Application Dispatcher* — средство создания Web-приложений на базе технологии CGI.
- *htmlSQL* — средство обработки запросов, которое позволяет вернуть в окно браузера в виде HTML отформатированные результаты выполнения запроса
- *Классы Java* — для включения в пользовательские приложения:
 - *SAS/SHARE Driver for Java* — для распространения отчетов
 - *SAS/CONNECT Driver for Java* — для распространения приложений
- *Java Tunnel* — CGI-программа, которая позволяет Java-апплету взаимодействовать с сервером, отличным от Web-сервера, в частности, работать через прокси-сервер.
- *AppDev Studio™* — интегрированная среда разработки Web-приложений для платформы Windows, включающая рабочие версии следующих продуктов SAS:

- BASE SAS
- SAS/GRAPH®
- SAS/AF®
- SAS/FSP®
- SAS/EIS®
- SAS/CONNECT
- SAS/SHARE
- SAS/IntrNet

AppDev Studio также содержит следующие компоненты:

- **webAF™** — интегрированная среда разработки Java-приложений, взаимодействующих с сервером SAS. Объекты в подобных приложениях являются распределенными: одна часть объекта (*viewer*) выполняется на клиентской стороне, другая — на стороне сервера (*model*). Создание распределенных объектов осуществляется с использованием Remote Object Class Factory (ROCF). ROCF дает возможность автоматически инкапсулировать модели SAS/AF в приложения Java, без необходимости создавать дополнительный специализированный код. webAF может использоваться в сценарии распространения приложений.
- **webEIS** — Java-приложение, позволяющее создавать Java-апплеты для работы с многомерными базами данных SAS (распространение отчетов)
- **SAS® Integration Technologies** — новый продукт, введенный в SAS System версии 8. SAS Integration Technologies представляет собой совокупность модулей, обеспечивающих открытость SAS System для программного обеспечения других производителей и возможность интеграции на основе ряда прогрессивных технологий. Все эти технологии позволяют задействовать богатые вычислительные возможности серверов SAS и способы представления информации вне собственно среды SAS как при интерактивной, так и при пакетной обработке. На момент публикации данной статьи в состав SAS Integration Technologies были включены следующие модули:
 - **Interface Object Model (IOM)** — шлюз для обмена информацией с серверами SAS на основе стандартов CORBA, COM, и DCOM.
 - **MQInterface** — интерфейс поддержки серверов сообщений MQSeries (IBM) и MSMQ (Microsoft). MQInterface позволяет приложениям SAS, написанным на языке SCL или на языке шага обработки данных (DATA step), помещать сообщения в очередь или извлекать сообщения из очереди. Сообщения могут быть использованы для доставки данных, команд по обработке данных или выполняемых бинарных файлов.

- **LDAP Interface** — интерфейс поддержки серверов директорий, работающих в стандарте Lightweight Directory Access Protocol (LDAP). Позволяет серверу SAS осуществлять поиск и извлечение информации, хранящейся в директории LDAP, а также размещать информацию в директории LDAP. Данный подход является ключевым для обеспечения масштабируемости при использовании репозитариев приложений.

- **Publish/Subscribe** — технологии «публикации/подписки» в рамках версии 8, позволяющие программно проводить сбор информационных объектов и их упаковку в пакеты для распространения клиентам. Обеспечивается поддержка на уровне языка для выполнения действий по публикации и подписке в рамках SAS System.

- **Enterprise Reporter™ (ER)** — продукт для конечного пользователя, позволяющий создавать и распространять отчеты, в том числе и в виде Web-публикаций. ER Enterprise Reporter существует на платформе Windows для версий SAS 6 и 7. В версии 8 Enterprise Reporter имеет сходную функциональность, однако, использует продукт SAS Integration Technologies для доступа к серверу SAS.

SAS Product or Component	Pattern		
	Web Publish	Dynamic Content	Application Distribution
Base SAS®	✓	✓	✓
– ODS	✓		
– Web Publishing Macros	✓		
SAS/MDDB Server®		✓	✓
SAS/ACCESS® products		✓	✓
SAS/CONNECT®			✓
SAS/SHARE®		✓	✓
SAS/IntrNet®		✓	✓
– Application Dispatcher		✓	✓
– htmSQL		✓	
– SAS/CONNECT Java driver		✓	✓
– SAS/SHARE Java Driver		✓	✓
– AppDev Studio™		✓	✓
webAF™			✓
webEIS™		✓	
SAS Integration Technologies		✓	✓
IOM		✓	✓
MQInterface		✓	✓
Directory Support		✓	✓
Pub/Sub		✓	✓
Enterprise Reporter™	✓	✓	

Рис. 10. Продукты SAS для создания Web-приложений.

Заключение

Сначала давайте дадим ответы на вопросы, заданные в начале статьи.

- Когда использование CGI лучше, чем Java?**
 Когда у Вас уже есть работающее CGI-приложение, которое отвечает Вашим требованиям как в отношении функциональности, так и в отношении производительности. Технологию CGI следует рассматривать в качестве первоначального варианта при использовании сценариев №2 и №3, так как эта технология является достаточно простой, понятной и легко реализуемой. В этом смысле CGI лучше, чем Java, просто потому, что Вы закончите Ваш проект в более короткое время. Вполне возможно, что после того, как Вы и Ваши пользователи поработаете с первым вариантом Web-приложения, использование Java станет единственным вариантом, который обеспечивает для пользователей необходимый уровень интерактивности приложения, или единственным вариантом, позволяющим реализовать в рамках архитектуры клиент-сервер, используя те же самые программные ресурсы.
- Дает ли Dynamic HTML ту же степень контроля над клиентским интерфейсом, что и JavaScript?**
 И Dynamic HTML, и JavaScript интерпретируются Web-браузером и обеспечивают возможность выполнения сценариев. Оба языка дают примерно равную степень контроля над клиентским интерфейсом и поведением Web-страницы, представленной в окне Web-браузера. Стоит отметить, что многое из того, что возможно в DHTML, требует использования JavaScript.
- В каких случаях предпочтительнее использовать Dynamic HTML вместо Java для Web-клиентов?**
 В отличие от JavaScript язык Java значительно отличается от Dynamic HTML по ряду характеристик. Java является последовательной реализацией объектно-ориентированного языка программирования и дает разработчику приложений полную свободу при кодировании логики программы и разработке интерфейса. С точки зрения возможностей программирования Dynamic HTML не может даже сравниться с языком Java.

- В какой момент генерация динамических отчетов по SQL запросам перестает удовлетворять требованиям пользователей?**

На самом деле, это сравнение сценариев №2 и №3, так что ответ зависит от того, насколько мощными должны быть возможности Вашего аналитического приложения, чтобы составлять отчеты на Web в соответствии с требованиями пользователей. Если Вы в состоянии дать им то, что они хотят, задействуя только SQL запросы, — прекрасно. Однако, если нужды пользователей простираются до нерегламентированных запросов и процедур специального анализа, формирования сложной графики или чего-либо такого, чего SQL запросы обеспечить не в состоянии, то Ваш генератор содержимого на базе SQL уже «выдохся». Если это происходит — переходите к использованию компоненты Application Dispatcher продукта SAS/IntrNet, чтобы предоставлять пользователям именно то содержимое, в котором они нуждаются. Application Dispatcher способен обратиться ко всему спектру возможностей сервера SAS в отношении управления данными и их анализа, чтобы генерировать содержимое на основе информации из хранилища данных.

Повсеместная распространенность Web значительно упростила процесс потребления информации пользователями. Но как Вы возможно уже заметили, те самые неотразимые возможности Web не упростили процесс доставки информации пользователям. Чтобы решить все Ваши задачи, Вам, возможно, потребуется использование комбинации сразу нескольких продуктов. Некоторые из этих продуктов вполне могут быть от SAS Institute. Наше стремление к открытым технологиям — что демонстрируется наличием продуктов SAS/IntrNet и SAS Integration Technologies, обеспечивающих возможность тесной интеграции в масштабе предприятия, — позволит Вам успешно создавать комплексные решения на базе Web, которые помогают решать задачи в области хранилищ данных, систем OLAP анализа и систем поддержки принятия решений.



Московское
представительство
109240, Москва,
Николаямская ул., 13
Тел.: +7 095 937 4151
Факс: +7 095 937 4155
<http://www.sas.com/russia>

SAS Institute Inc.
Европейская штаб-квартира
Neuenheimer Landstr. 28-30
P.O. Box 10 53 40
D-69043 Heidelberg, Germany
Тел.: +49 6221 4160
Факс: +49 6221 474850

SAS Institute Inc.
Мировая штаб-квартира
SAS Campus Drive,
Cary, NC 27513 USA
Тел.: +1 919 677 8000
Факс: +1 919 677 4444
<http://www.sas.com>