гистерезисных потерь при циклической нагрузке. С этой точки зрения, наихудшими механическими характеристиками обладает второй образец» [16].

Использование представления об ОСУ изображений при распределении уровня серого показал, что второй образец – более упорядоченный, чем первый (ΔS=4,683) и третий (ΔS=4,752). С учетом результатов [16] видно, что с ростом упорядоченности структуры углеродных кластеров в резине ее механические свойства ухудшаются.

Заключение

Относительную степень упорядоченности можно рассматривать как интегральную количественную характеристику структуры физических тел, которая является альтернативой описательным понятиям типа «структурированность», «неоднородность», «сложность».

Очевидна возможность ввода в рассмотрение «абсолютной» упорядоченности. Для ее определения нужно выбрать нуль шкалы: это может быть упорядоченность изображения, созданного генератором случайных чисел с равномерным распределением, или, напротив, – полная упорядоченность «черного квадрата».

Литература

1. Герега А.Н. Об одном критерии относительной степени упорядоченности изображений. // Журнал технической физики. 2010. – Т. 80, вып. 5. – С. 149-150.
2. Хакен Г. Принципы работы головного мозга. – М.: Per Se, 2001. – 351 с.
3. Солсо Р. Когнитивная психология. – СПб.: Питер, 2006. – 589 с.
4. Прэтт У. Цифровая обработка изображений. – М.: Мир, 1982. – Кн. 1. – 310 с.
5. Методы компьютерной обработки изображений. / Под ред. В. Сойфера. – М.: Физматлит, 2001. – 784 с.
6. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: Техносфера, 2006. – 1072 с.
7. Гиббс Дж.В. Основные принципы статистической механики. / В кн. Дж. Гиббс Термодинамика. Статистическая физика. – М.: Наука, 1982. – 584 с.
8. Климонтович Ю.Л. Статистическая физика. – М.: Наука, 1982. – 608 с.
9. Герега А.Н. Моделирование кластерных структур в материале: силовые поля и дескрипторы. // Физическая мезомеханика. – 2013. – Т. 16, №5. – С. 87-93.
10. Kullback S., Leibler R.A. On information and sufficiency. // Annals of Mathematical Statistics. 1951. – V. 22, No. 1. – P. 79-86.
11. Кульбак С. Теория информации и статистика. – М.: Наука, 1967. – 480 с.
12. Климонтович Ю.Л. Уменьшение энтропии в процессе самоорганизации. S-теорема (на примере перехода через порог генерации). // Письма в Журнал технической физики. 1983. – Т. 9, вып. 23. – С. 1412-1416.
13. Климонтович Ю.Л. Критерии относительной степени упорядоченности открытых систем. // Успехи физических наук. 1996. – Т. 166, № 11. – С. 1231-1243.
14. Словарь иностранных слов. – М.: Русский язык, 1987. – 608 с.
15. Шестов С.С., Загребин Л.В., Яновский Ю.Г. и др. Упрочняющие оксидные покрытия алюминиевых сплавов «АСТРИМ», полученные на основе технологии микродугового оксидирования. Исследования структуры, шероховатости и микротвёрдости. // Механика композиционных материалов и конструкций. – 2010. – Т. 16, № 4/2. – С. 670-679.
16. Морозов И.А. Анализ микроструктуры наполненной резины при атомно-силовой микроскопии. // Механика композиционных материалов и конструкций. 2009. – Т. 15, – № 1. – С. 83-93.

УДК 004.02

ТЕХНОЛОГИИ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЙ

Лобода Ю.Г., канд. пед. наук, доцент, Орлова О.Ю., ст. преподаватель
Одесская национальна академия пищевых технологий, г. Одесса

Рассматриваются современные методы проектирования и разработки web-приложений. Поясняются понятия и основные стандарты сети Интернет, логика работы web-приложений, основные подходы и технологии их разработки. В качестве примера подхода к разработке web-приложений рассматривается технология Common Gateway Interface, а также подходы на основе объектных сред и архитектурного шаблона MVC.

Considered modern methods of designing and developing web applications. Explains the basic concepts and standards of the Internet, logic web applications, basic under-methods and technologies of their development. As an example of the approach to web application development technology is considered Common Gateway Interface, and approaches based on the object environments and architecture the MVC pattern.

Ключевые слова: web-приложения, скрипт, сервер, сервлет, технология.

Несколько лет назад большинство Web-сайтов представляло собой набор статических HTML-страниц. Однако, в процессе превращения Интернета из набора информационных ресурсов в инструмент ведения бизнеса технологии создания сайтов существенно изменились, большинство Web-сайтов крупных компаний представляет собой набор приложений, обладающих интерактивностью, средствами персонализации, средствами взаимодействия с клиентами (вплоть до приема заказов и платежей) и партнерами, а нередко, и средствами интеграции с «внутренними» корпоративными приложениями компании.

Цель статьи заключается в рассмотрении современных методов проектирования и способов разработки web-приложений. В качестве примера подхода к разработке web-приложений рассматривается технология Common Gateway Interface (CGI, общий шлюзовой интерфейс).

Материалы и методика исследований. Способы разработки web-приложений могут быть разделены на три большие категории: подходы, основанные на программировании или скриптах: внешние программы или скрипты; расширения web-сервера; подходы, основанные на использовании шаблонов web-страниц, включающих вставки кода скриптов и специальных серверных тэгов; объектные среды (каркасы, фреймверки, frameworks).

Хотя между этими категориями и имеются пересечения (а также различные мнения о том, к какой категории относится конкретная технология разработки), большинство широко известных подходов связаны с одной конкретной категорией.

Рассмотрим *программные подходы*. В данном подходе web-приложением (динамическим ресурсом, связанным с URL адресом) является внешняя программа, составленная на некотором универсальном языке программирования высокого уровня (например, таком, как Java или C++) или скрипт, составленный с помощью скриптового языка (scripting language), выполнение которого производится также с помощью внешней программы – интерпретатора скриптов (script engine). Основной проблемой с программным подходом к разработке web-приложений является их ориентация на написание кода. Разметка HTML и другие конструкции форматирования встраиваются в логику работы программы с помощью операторов вывода.

Это ограничивает возможности web-дизайнеров вносить свой вклад в оформление создаваемой приложением страницы. Web-дизайнер может разрабатывать макет страницы, а программист должен затем преобразовывать его в код и связать со скриптом или программой. Для изменения практически любого элемента формируемой страницы требуется вмешательство программиста, касается ли это изменения логики работы программы, либо изменения оформления и расположения элементов страницы.

Внешние программы. Простейший способ динамически формировать web-страницы в ответ на HTTP запрос заключается в том, чтобы передать работу по решению требуемой задачи и формированию HTML страницы внешней программе, которая должна получать переданные в HTTP запросе входные параметры и сформировать выходную страницу на языке HTML.

Первой широко используемой, независимой от типа web-сервера, программной технологией создания и выполнения web-приложений была технология Common Gateway Interface (CGI, общий шлюзовой интерфейс). Она определяла набор правил, которым должна следовать программа, чтобы она могла выполняться на разных HTTP серверах и операционных системах.

В соответствии с CGI технологией при поступлении в web-сервер HTTP запроса, который включает ссылку не на статическую страницу, а на CGI программу (например: prog.exe), создается новый процесс, в котором запускается требуемая прикладная программа. Технология CGI задает способ передачи такой программе параметров, входящих в состав HTTP запроса. Передача входных данных может выполняться либо с помощью фиксированного набора переменных среды (environment va-riables), которые могут создаваться одной программой и использоваться другими программами), либо через входные данные функции, с которой начинается работа программы (функция main()), а результаты работы программы (HTML страница) возвращаются с помощью стандартного потока вывода STDOUT. Приведем пример простой CGI программы, написанной на языке C, которая формирует HTML страницу с перечнем переданных ей параметров.

```
// Пример простой CGI программы:
#include "stdafx.h"
#include "cgi1.h"
#include <stdlib.h>
```

```
int main(int argc, TCHAR* argv[ ], TCHAR* envp[ ]) {
printf("Content-Type: text/html\n\n"); printf("<HTML>\n");
printf("<HEAD> <TITLE>Пример CGI-программы </TITLE></HEAD>\n");
printf("<BODY>"\n); printf("Пример CGI-программы на C\n");
int i=0;
while(envp[i]) {
// перечать параметров переданных программе в массиве envp
printf("<p>значение параметра %s </p>",envp[i]);
i++;
}
printf("</BODY>\n"); printf("</HTML>\n);
return 0; }
```

Технология CGI позволяет использовать любой язык программирования, который может работать со стандартными устройствами ввода/вывода. Кроме этого, CGI программы можно писать с использованием скриптовых языков, которые называются “CGI скриптами”. Примерами скриптовых CGI языков являются, например, Perl, Python или Tcl. При использовании скрипта web-сервер вызывает на выполнение внешнюю программу – интерпретатор скриптов (script engine), которой передаются данные HTTP запроса и имя файла, в котором содержится запрашиваемый пользователем скрипт. А затем данная программа выполняет указанный скрипт и возвращает серверу сформированную HTML страницу.

Недостатки технологии CGI. Технология CGI является достаточно простым способом динамически формировать информацию в web-сети, но она имеет существенные недостатки, которые делают ее не практичной в большинстве случаев:

— основной проблемой является производительность: для каждого HTTP запроса к CGI программе web-сервер запускает новый процесс, который заканчивает работу только после завершения программы. Работа по созданию и завершению процессов является достаточно трудоемкой, что может очень быстро понизить производительность системы, кроме этого различные активные процессы начинают конкурировать за системные ресурсы, такие как оперативная память.

— для составления и отладки CGI программ разработчик должен обладать достаточно большим опытом программирования на одном из языков, на которых можно программировать CGI программы.

— в CGI программах программный код и код разметки полностью перемешаны. Дизайнер должен знать программирование, чтобы менять структуру web-страниц.

Попыткой объединить переносимость CGI приложений с эффективностью является технология FastCGI. Данная технология основывается на простой идеи: вместо необходимости каждый раз запускать новый процесс для обработки CGI скрипта, FastCGI позволяет не закрывать процессы, связанные с CGI скриптами, после окончания обработки, а использовать их для обработки новых запросов к CGI программам. А это означает, что не требуется постоянно запускать и удалять новые процессы, так как один и тот же процесс может использоваться многократно для обработки запросов. Такие процессы могут инициализироваться только один раз при их создании.

Модули сервера, которые выполняют функциональность FastCGI, взаимодействуют с HTTP сервером с помощью своих собственных API. Эти API стараются скрыть детали реализации и конфигурирования от FastCGI приложений, но разработчики все равно должны знать особенности реализации технологии FastCGI, так как модули различных типов серверов не совместимы между собой.

Расширения web-серверов. Недостатки технологий CGI можно также преодолеть путем расширения возможностей web-серверов с помощью специальных компонентов. Используя такие расширения, программы, формирующие HTTP ответы, могут выполняться более эффективно, без необходимости их завершения после обработки каждого запроса и за счет использования общих ресурсов несколькими приложениями. Такие технологии обычно предоставляют возможность хранить в основной памяти данные сеансов работы пользователей, которые взаимодействуют с приложением в течение большого числа HTTP запросов.

Интерфейс Java Servlet API. Другой широко используемой технологией расширения архитектуры web-сервера является прикладной интерфейс Java Servlet API, который связывает web-server с виртуальной машиной Java Virtual Machine (JVM). Виртуальная машина JVM поддерживает выполнение специальной Java программы (контейнер сервлетов), которая отвечает за управление данными сеанса работы и выполнение Java-сервлетов.

Сервлеты это специальные классы на языке Java (программы), которые имеют доступ к информации из HTTP запросов. Они формируют HTTP ответы, которые возвращаются браузерам. Контейнер сервлетов (среда выполнения) отвечает за получение от web-сервера HTTP запросов на выполнение сервлетов; создание сеанса работы пользователя, если это требуется; вызов сервлета связанного с HTTP запросом;

передачу сервлету параметров, которые содержатся в HTTP запросе, представленных в виде Java объектов. В отличие от ISAPI расширений, технология Servlet API является переносимой между разными web-серверами, операционными системами и компьютерными платформами. Сервлеты выполняются одинаково в любой среде, которая предоставляет совместимый с ними контейнер сервлетов. Технология Servlet API используется большим количеством разработчиков и поддерживается многими известными web серверами.

Подходы на основе шаблонов. Подходы, основанные на *шаблонах* (template approaches, шаблонные подходы) используют в качестве адресуемых объектов (имеющий URL-адрес) не программы или скрипты, а «шаблоны». По существу шаблоны являются HTML файлами с дополнительными “тэгами”, которые задают методы включения динамически формируемого контента. Таким образом, файл шаблона содержит HTML код, который описывает общую структуру страницы, и дополнительные *серверные тэги*, размещенные таким образом, чтобы формируемой с их помощью содержание странице имело требуемый вид.

В настоящее время к наиболее распространенным технологиям разработки web-приложений на основе шаблонов, относятся следующие: Server-Side Includes (SSI), Cold Fusion, PHP, Active Server Pages (ASP) и Java Server Pages (JSP).

Технология Cold Fusion. Другой достаточно популярной технологией, основанной на шаблонах, является технология Cold Fusion, разработанная компанией Adobe. Преимущество данного подхода заключается в том, что данный шаблон может создаваться и поддерживаться дизайнером страницы, который имеет базовые знания языка HTML и web-графики, но не имеет опыта программирования. Специальные тэги, которые являются “расширением” HTML .

Технология PHP Hypertext Preprocessor. Технология «PHP Hypertext Preprocessor» или просто PHP позволяет разработчикам встраивать программный код в шаблоны, с помощью языка, сходного с языком скриптов Perl.

Технология Active Server Pages. Компания Microsoft разработала технологию ASP (Active Server Pages), которая объединила возможности создания шаблонов, включающих скрипты, с доступом к набору OLE и COM объектов, имеющихся с операционной системе Windows, в том числе и к ODBC источникам данных. Данная технология, объединенная с бесплатным web сервером Internet Information Server (IIS), быстро стала популярной среди программистов, использующих Visual Basic, которые оценили возможность использования в шаблонах языка VBScript. Как и PHP шаблоны, ASP страницы могут включать блоки скриптов (использующими ссылки на COM объекты), вперемешку с HTML форматированием. В отличие от таких объектно-ориентированных языков, как Java или C++, язык, используемый в ASP страницах, был плоским, линейным и строго процедурным.

В отличии от технологии PHP, ASP не связан с одним конкретным скриптовым языком. В ASP в качестве стандартного языка используется язык Visual Basic Scripting Edition (VBScript), но может использоваться и язык JavaScript.

В ASP шаблоны (также, как и в PHP шаблоны) могут включаться блоки, выделенные с помощью тэгов <% … %>, которые содержат код скрипта, выполняемый интерпретатором ASP шаблонов, при формировании ответа. HTML разметка, который находится вне таких блоков, рассматривается как исходный HTML код и просто переписывается в формируемую HTML страницу. Кроме этого в начало шаблона могут добавляться директивы страницы, такие, как например, <% @LANGUAGE = VBScript %>, которая информирует систему обработки об используемом скриптовом языке.

Подходы на основе объектных сред. Обычные скриптовые технологии на стороне сервера используют различные объекты, но не позволяют разрабатывать и использовать собственные классы и создавать на их основе объекты. В связи с этим дальнейшее развитие web-технологий было связано с созданием специальных объектно-ориентированных технологий разработки web-приложений. Использование данных технологий позволяет сделать разработку web-приложений более сходной с разработкой обычных объектно-ориентированного программного обеспечения.

Объектные среды (фреймверки, frameworks) представляют собой следующий уровень совершенствования разработки web-приложений. Вместо объединения разметки и логики в единый модуль, объектные среды (frameworks) поддерживают принцип отделения содержания от представления. Модули ответственные за создание контента отделяется от модулей, которые показывают это содержание в конкретном формате.

В настоящее время есть два подхода к созданию объектно-ориентированных web-приложений:

— подходы, основанные на наборе специальных web-страниц (web-форм), связанных с описаниями классов, объекты которых будут создаваться, и использоваться при их вызове (например: технология ASP.Net Web Forms**;** технология JavaServer Faces);

— подходы, основанные на использовании наборов классов, соответствующих шаблону Model-View-

Controller (MVC) (например: технологии на основе языка Java – Tapestry, Struts, Spring и технология компании Microsoft – ASP.Net MVC).

Подход на основе архитектурного шаблона MVC. В соответствии с архитектурным шаблоном MVC все классы, составляющие приложение (в том числе и web-приложение) делятся на три основные группы (компоненты): Модель (Model), Представление (View) и Контроллер (Controller). Каждый из этих компонентов отвечает за свои задачи:

— Модель (Model) – это набор классов, реализующих всю бизнес-логику web-приложения. Эти классы отвечают за обработку данных, размещение их в БД, чтение из БД, а так же за взаимодействие между самими объектами, составляющими такие данные.

— Представление (View) – набор классов, отвечающих за интерфейс взаимодействия с пользователями (User Interface, UI). С их помощью формируются HTML страницы, показывающие пользователям данные. Представление использует данные из Модели и предоставляет пользователям возможность выполнять их редактирование.

— Контроллер (Controller) – это связующее звено между первыми двумя компонентами. Классы данного компонента получают данные о запросе к серверу (например, значения, полученные из отправленной формы), и передает их в Модель для обработки и сохранения. После обработки полученных данных Контроллер выбирает, каким способом показать их клиенту с помощью использования некоторого класса из Представления.

В результате такого разделения web-приложения на компоненты, разработчик получает полный контроль над формируемым HTML документов; упрощается структура приложения; облегчается задача выполнения тестирования приложения; достигается полное отделение логики работы приложения от представления данных.

Примерами технологий разработки на основе MVC являются:

— технология Struts (основанная на языке Java);

— технология ASP.Net MVC, входящая в состав набора технологий ASP.Net платформы .Net Framework;

— технология Ruby on Rails (Ruby – язык программирования, а Rails – фреймворк, использующий данный язык) особенно популярная в последнее время.

Общие рекомендации по разработке web-приложений. Основной целью архитектора программного обеспечения при проектировании web-приложений является максимальное упрощение их структуры путем разделение задач на функциональные области, обеспечивая при этом безопасность и высокую производительность. Для эффективной работы web-приложений в обычных для них сценариях, необходимо:

— выполнить логическое разделение функциональности приложения, используя многослойную структуру для логического разделения приложения на слой представления, бизнес-слой и слой доступа к данным. Это помогает создать удобный в обслуживании код и позволит отслеживать и оптимизировать производительность каждого слоя в отдельности. Четкое логическое разделение также обеспечивает более широкие возможности масштабирования приложения.

— использовать абстракцию для реализации слабого связывания между слоями. Этот подход можно реализовать с помощью интерфейсных типов или абстрактных базовых классов можно определить совместно используемую абстракцию, которая должна быть реализована интерфейсными компонентами.

— определиться, как будет реализовано взаимодействие компонентов друг с другом. Для этого необходимо понимать сценарии развертывания, которые должно поддерживать приложение.

— использовать кэширование для сокращения количества сетевых вызовов и обращений к базе данных.

— использовать протоколирование и инструментирование. Необходимо выполнять аудит и протоколирование действий в слоях и уровнях приложения. Журналы регистрации событий могут использоваться для выявления подозрительных действий, что часто обеспечивает раннее обнаружение атак на систему.

— продумать аспекты аутентификации пользователей на границах доверия. При проектировании приложения необходимо предусмотреть аутентификацию пользователей при пересечении границ доверия, например, при доступе к удаленному бизнес-слою из слоя представления.

— не передавать важные конфиденциальные данные по сети в виде открытого текста. Если требуется передавать по сети конфиденциальные данные, такие как пароль или куки аутентификации, использовать для этого шифрование и подписи данных либо шифрование с использованием протокола Secure Sockets Layer (SSL).

— проектировать web-приложение для выполнения с менее привилегированной учетной записью. Процесс должен иметь ограниченный доступ к файловой системе и другим ресурсам системы. Это позволит максимально сократить возможные негативные последствия на случай, если злоумышленник по-

пытается взять процесс под свой контроль.

Существует ряд общих вопросов, на которые следует обратить внимание при проектировании. Эти вопросы можно сгруппировать по определенным областям проектирования: обработка запросов приложения; аутентификация; авторизация; кэширование; управление исключениями; протоколирование и инструментирование; навигация; компоновка страницы; формирование визуального отображения страницы; управление сеансами; проверка введенных данных (валидация).

Литература

1. Комагоров В.П. Технологии сети Интернет: протоколы и сервисы. [Учебное пособие] / В.П. Комагоров. – Томск, изд-во ТПУ, 2008. – 112 с.
2. Руководство компании Microsoft по проектированию архитектуры приложений (второе издание). 2009. – 560 с. [Электронный ресурс]. – Режим доступа: http://download.microsoft.com/documents/rus/msdn/ры_приложений_полная_книга.pdf
3. Столбовский Д.Н. Основы разработки Интернет приложений и Web сервисов на основе ASP.Net: Учебный курс / Д.Н. Столбовский. – Владикавказ: Северо-Кавказский ГМИ, 2008. – 256 с. [Электронный ресурс]. – Режим доступа: http://window.edu.ru/window/library?p_rid=57408)
4. Тузовский А.Ф. Высокоуровневые методы информатики и программирования. Учебное пособие / Тузовский А.Ф. – Томск, изд-во ТПУ, 2009. – 200 с.
5. Эспозито Д. Microsoft ASP.Net 2.0. Базовый курс / Пер. с англ. – М.: Издательство «Русская редакция», 2007. – 688 с.

УДК 004.02

ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ МОНІТОРИНГУ ТА ЗАХИСТУ ДОВКІЛЛЯ

Лобода Ю.Г., канд. пед. наук, доцент, Орлова О.Ю., ст. викладач
Одеська національна академія харчових технологій, м. Одеса

У статті розглянуто питання використання інформаційних технологій які допомагають людині у моніторингу довкілля. Представлено загальну характеристику та дано визначення географічних інформаційних систем, розглянуто загальну характеристику їхніх компонентів і визначальні функції.

In the article the questions of using information technologies, which help a person in environmental monitoring. Presents a General overview and definition of geographical information systems, considers General characteristics of their components and distinctive features.

Ключові слова: моніторинг, географічна інформаційна система, дигітайзер.

Вступ. Сучасний розвиток інформаційних технологій та просторовий характер більшості екологічних аспектів природно-антропогенних систем, їхня багатофакторність та значні обсяги даних, що обробляються, зумовили необхідність автоматизації еколого-географічного картографування із застосуванням сучасних комп'ютерних технологій, що дістало назву – географічні інформаційні системи (ГІС). Вважається, що саме просторовий аналіз є головним напрямом розвитку ГІС. Світовий досвід показав надзвичайну ефективність і перспективність використання ГІС у багатьох сферах життєдіяльності суспільства.

Мета статті полягає у розгляді питань систем екологічного управління, використання комп'ютерних технологій для картографування та аналізу об'єктів навколишнього природного середовища, а також подій, що відбуваються в ньому. Для досягнення мети були поставлені такі завдання: визначити архітектуру ГІС (географічна інформаційна система), з'ясувати основні етапи технологічного процесу ГІС

Матеріали і методика досліджень. Потужною силою в розвитку сучасного суспільства є інтенсивне глобальне поширення інформаційно-комунікативних технологій, які допомагають збирати, зберігати, аналізувати та розповсюджувати інформацію. Слід зазначити, що найбільшого розвитку інформаційні технології досягли в США. Мали створені комп'ютерами та розміщені на веб-сайті КТСД (Коаліція щодо токсикантів Силіконової Долини), які містять дані про забруднення, – це лише один із прикладів того, як інформаційні технології допомагають людині у моніторингу довкілля. Активно в цьому напрямі працює і Європейське космічне агентство (ЄКА). Прикладом цього є проект «Глобальний моніторинг навколиш-