МЕТОДИКА ПРЕПОДАВАНИЯ

УДК 681.324

ПОДХОД К МЕТОДИКЕ ОБУЧЕНИЯ ПРОЕКТИРОВАНИЮ WEB-РЕСУРСОВ В ВУЗЕ: ОСНОВНЫЕ ПРОБЛЕМЫ И ВЫБОР СРЕДСТВ ПРОЕКТИРОВАНИЯ

Е.С. Перевезенцева

Рассматривается специфика Web-ресурсов как наиболее актуального класса современных информационных систем. Анализируются проблемы обучения проектированию Web-ресурсов в вузе и предлагается подход к разработке, предполагающий комплексное использование различных видов средств проектирования.

Specific features of Web-resources as the most important class of modern information systems are described in this paper. The problems of teaching to design Web resources in a higher educational establishment are analyzed, and the approach based on an integrated use of various design tools is proposed.

КЛЮЧЕВЫЕ СЛОВА
Web-ресурс, проектирование, средства проектирования, методология RUP, язык UML, диаграммы UML.

1. ВВЕДЕНИЕ

Web-ресурсы, представляющие собой наиболее актуальный класс современных информационных систем, одновременно являются одним из наиболее сложных в разработке классом, объединяющем в себе свойства целого ряда компьютерных продуктов. Это порождает специфические проблемы разработки и требует выбора адекватных этой специфике средств. В работе акцент делается на использовании методологии RUP (Rational Unified Process) – методологии, представляющейся на данном уровне развития информационных технологий наиболее адекватной задачам разработки сложных программных систем, ориентирующихся на интерактивное взаимодействие с пользователем и, следовательно, плохо поддающихся формальному описанию и требующих при проектировании достаточно гибких средств.

Процессу проектирования уделяется основное внимание. В ситуации, когда компьютерные продукты становятся неотъемлемым элементом во всех – весьма разнородных и в большинстве своем плохо поддающихся формальному описанию – сферах деятельности, это наиболее ответственный этап, базирующийся на анализе целей пользователей и классов решаемых задач. Удовлетворение целей пользователя является основным критерием успешности продукта. Вся разработка, и прежде всего – проектирование, ориентирована на это. Эти тезисы, порождаемые реальностью, присутствуют во всех серьезных источниках, посвященных

разработке компьютерных продуктов, причем характерно, что эти источники могут быть независимы и относиться к различным направлениям, подходам и этапам разработки.

Качество конечного продукта в первую очередь определяется качеством проекта. В то же время достаточно формализованные методы разработки, применимые для решения современных задач, и в первую очередь – методы проектирования, как нечто целостное просто отсутствуют. Формирование таких методов на основе анализа традиционно используемых и рождающихся в практике разработки подходов представляет непростую, но нуждающуюся в решении задачу. Автор предлагает некоторые средства ее решения, вписывающиеся в учебный процесс.

Излагаемые в статье положения используются автором в практике преподавания на факультете информационных технологий МГППУ ряда дисциплин, как напрямую преследующих цель обучения принципам создания Web-ресурсов, так и связанных с их созданием, а также при выполнении конкретных разработок.

2. ХАРАКТЕР СОВРЕМЕННЫХ WEB-РЕСУРСОВ

2.1. Специфика использования

С точки зрения пользователя Web-ресурсы представляются совокупностью Web-страниц, содержащих необходимую пользователю информацию и элементы управления, а процесс работы с ними – как переход между страницами, управляемый данными пользователя, задаваемыми в ответ на запрос. Традиционный термин «сайт» относится к этой – пользовательской – стороне Web-ресурсов. Однако современные Web-ресурсы как компьютерный продукт характеризуются колоссальным расхождением между тем, что предъявляется пользователю, и тем, как это формируется и предъявляется и какие средства при этом используются. При этом усложнение структуры Web-ресурсов и их эволюция от совокупности HTML-страниц, размещенных на сервере и выдаваемых пользователю, до огромных программных комплексов, когда подавляющая часть предъявляемых страниц формируется программным путем в процессе функционирования ресурса, диктуется прежде всего огромным диапазоном пользовательских задач.

Наиболее значимые черты современных Web-ресурсов с точки зрения их использования:

– охват самых разных проблемных областей;
– интерактивность – достижение результата путем взаимодействия с пользователем; направление процесса работы ресурса как программной системы определяется данными, задаваемыми пользователем в ответ на запрос системы;
– применение различных средств предъявления информации, или мультимедийный характер (от простого текста до трансляции видео в реальном режиме времени);
– применение различных средств хранения информации (от хранения данных в виде файлов до использования систем управления базами данных).

Этот диапазон возможностей и областей практического применения определяет синергетический характер – взаимоусиление воздействий различных факторов применения, взаимопроникновение требований – как самих ресурсов, так и идей, принципов и методов их разработки и дает толчок не просто к поиску методов разработки в конкретной области, но к их анализу и интеграции на качественно ином уровне, способном обеспечить некоторый общий подход к разработке. Острая необходимость в такого уровня методах особенно сильна в случае, когда речь идет о преподавании основ интернет-технологий, где Web-ресурсы выступают как объект изучения и разработки, т.е. при профессиональной подготовке лиц, которым, в соответствии с полученной специальностью, предстоит участвовать в разработке в той или иной роли (от постановщика задачи до создателя кода). Наличие достаточно точных и адекватных средств описания объекта и процесса разработки становится ключевым фактором обучения;

при этом упомянутые средства должны быть достаточно универсальными и сохранять свою актуальность и по завершении процесса обучения.

Следует также учесть новизну самой области, вступающую в противоречие с консервативностью используемых методов разработки компьютерных продуктов – при том, что недостаточно разработаны даже методы создания программного обеспечения в традиционном плане, не говоря об интернет-программировании.

2.2. Специфика разработки

Диапазон возможностей Web-ресурсов с точки зрения использования определяет их главную характеристику с точки зрения внутренней структуры и технологии разработки – двойственный характер [2]. Это приводит к необходимости вести разработку в двух очень разных, но одновременно взаимосвязанных аспектах.

С одной стороны, основным содержанием Web-ресурсов являются данные, ради представления которых ресурс создается, или контент. Отсюда – задачи проектирования контента и организации взаимодействия с пользователем, появившиеся одновременно с рождением Интернета. Это основной (и сложившийся первым) класс задач проектирования в случае, если Web-ресурс представлен набором статических, заранее созданных и хранимых на сервере Web-страниц, связанных ссылками. Первоначально этот класс задач был единственным.

С другой стороны, современные Web-ресурсы являют собой развитые программные комплексы, использующие многообразные программные средства и инструменты для создания кода и системы управления базами данных (СУБД) для хранения информации и управления ею. Информация, предъявляемая пользователю, плюс возможности интерактивного взаимодействия с ресурсом (сайтом) есть не что иное как результат работы соответствующих программ (точнее, совокупности программных сценариев), основная часть которых функционирует на стороне сервера. Отсюда – задачи проектирования Web-ресурсов как программных комплексов.

Приведенные положения иллюстрируются на рис.1.

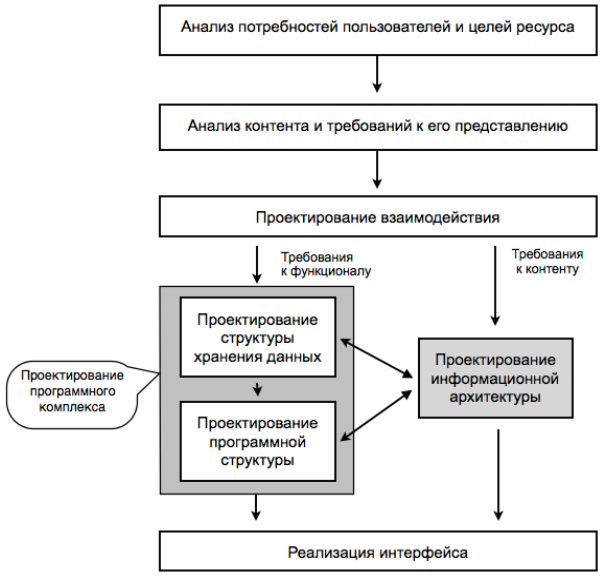


Рис. 1. Два аспекта проектирования Web-ресурса.

Схема достаточно условна; связи, идущие сверху вниз, означают прежде всего не временную последовательность разработки, а факт реализации вышележащих блоков в нижележащих. Тем не менее она в общих чертах позволяет отобразить двойственность объекта рассмотрения. Кратко остановимся на основных элементах этой схемы.

В действительности проектирование взаимодействия, носителем которого является интерфейс компьютерного продукта как совокупность средств, большая часть которых представлена визуальными элементами управления, пронизывает (но, к сожалению, далеко не всегда должным образом реализуется) практически все смысловые моменты проектирования. Однако для пользователя продукт представлен только своим интерфейсом.

Главное, что должно быть заложено в проектирование взаимодействия – цели пользователей, для которых разрабатывается продукт, и классы решаемых ими задач [1, 3]. Теоретической базой здесь служат основные принципы когнитивной психологии, преломляемые под углом взаимодействия человека с компьютером [5]. Хорошо спроектированный и реализованный интерфейс прежде всего должен быть состоятельным, т.е. при решении задач, для которых предназначен продукт, все элементы интерфейса должны поддерживать решение этих задач, в комплексе представлять собой инструмент их решения.

Информационная архитектура определяется как сочетание схем организации, предметизации и навигации, реализованных в информационной системе. Цель создания этих схем – помощь пользователю в поиске нужных данных [4, 12]. Эта сторона разработки исторически начала складываться раньше, изначально была единственной и воплощалась в структуре размещаемых на сервере страниц и навигации по ним.

Как было сказано выше, современные Web-ресурсы в большинстве своем – сложные программные комплексы. Поэтому к их разработке применимы все требования, предъявляемые к разработке программного обеспечения (ПО), и могут быть использованы соответствующие средства проектирования. Страницы ресурса, видимые пользователем, формируются программным путем; на сервере размещается не готовый сайт, а программные сценарии, его формирующие.

В Web-ресурсах традиционной формы (статических сайтах) информационная архитектура воплощается в распределении информации по страницам и схеме навигации по ним. В современных Web-ресурсах информационная архитектура программно формируемой части ресурса воплощается и в структуре баз данных, и в программной реализации последовательности подачи пользователю материала, управляющих элементов и способов передачи данных.

В конечном продукте оба аспекта интегрируются в организацию взаимодействия с пользователем и по отдельности могут рассматриваться только в процессе проектирования как аспекты взаимодействия – с точки зрения предъявления информации пользователю (с позиций информационной архитектуры) и с точки зрения способа формирования этой информации (с программных позиций). Проще говоря, информационная архитектура определяет то, что будет видеть пользователь, а программная часть ресурса – как это формируется.

Следует отметить, что строго формализованных средств, позволяющих описать хотя бы каждый аспект по отдельности, нет. Дисциплины, посвященные информационной архитектуре, располагают целым рядом положений [4], однако не являются единой сложившейся дисциплиной. То же можно сказать о технологии разработки программного обеспечения (ПО), состояние которой не расценивается как удовлетворительное. Поэтому единственно приемлемым выходом из ситуации представляется использование средств из той и другой области с предварительным анализом возможности и целесообразности их применения и возможным внесением модификаций. Необходимость сочетать методы этих двух областей усугубляет проблему, но именно такой подход применяется на практике при достаточно глубокой проработке проектов.

Интерфейс (точнее, визуальный интерфейс) Web-ресурса с пользователем являет собой материальное воплощение организации взаимодействия.

Следует отметить, что каждый из выделенных на вышеприведенной (приблизительной и неполной) схеме располагает собственной – в той или иной мере разработанной и формализованной – технологией, складывавшейся достаточно самостоятельно и в итоге не стыкующийся автоматически с остальными. При этом на схеме никак не отображена специфика собственно сети – обмен данными между компьютерами, клиент-серверное взаимодействие, где также существуют свои принципы, технологии и инструменты.

Учитывая описанную многоаспектность рассмотрения Web-ресурсов, в первую очередь необходимо определить, что же все-таки должно быть объектом изучения в курсах, суть которых можно определить как «Web-технологии». Очевидна низкая эффективность изучения одного конкретного аспекта (хотя бы и нескольких!) и невозможность охвата в учебном курсе всех аспектов. Так, ни владение тонкостями языка разметки страниц, ни умение программировать отдельные алгоритмы на сетевых языках без знания принципов и протоколов клиент-серверного взаимодействия и соответствующих умений не дадут возможности создать даже простейший реальный Web-ресурс; точно так же его невозможно создать, не умея создать HTML-страницу, но теоретически познав тонкости протоколов и организации запросов.

Выход видится в отборе базовых элементов, необходимых для построения того, что можно было бы считать Web-ресурсом в минимальном объеме, и освоения технологии построения каркаса Web-ресурса, который может лечь в основу дальнейших разработок уже в реальной, не учебной деятельности. Одновременно необходимо расставить акценты на важности тех или иных элементов и основное внимание сосредоточить на ключевых моментах, оставив остальное для самостоятельной проработки.

В данном случае применительно к схеме рис. 2 такими ключевыми моментами являются основные принципы выполнения отдельных этапов и организация их взаимосвязи, т.е. проектирование ресурса в целом. Применительно к реальности необходимо знание принципов работы в сети и соответствующие умения.

Сведения о языке разметки страниц, языках программирования на клиентской и серверной стороне, создании баз данных как правило либо рассматриваются либо непосредственно в отдельных курсах, либо экстраполируются на основе этих курсов (например, программирование), либо в значительной степени могут быть почерпнуты из справочников.

3. СРЕДСТВА ПРОЕКТИРОВАНИЯ И ДОКУМЕНТИРОВАНИЯ

3.1. Анализ основных средств

Рассмотрим некоторые основные средства проектирования и документирования компьютерных продуктов, в том числе Web-разработок, представленные в доступных источниках, включая Интернет, и используемые на практике. В целом средства этого перечня различаются назначением, областями применения, известностью и широтой распространения, и определение степени их применимости к Web-проектированию и особенно – комплексного использования представляет отдельную задачу, решаемую здесь лишь в ограниченном объеме.

Кратко проанализируем следующие направления:

– методы и средства разработки информационной архитектуры Л. Розенфельда [4];
– методология проектирования взаимодействия с использованием персонажей А. Купера [1, 3];
– средства проектирования взаимодействия, описанные в книге Дж. Гарретта [2];
– средства и методы проектирования и документирования сложных программных проектов (методология RUP – Rational Unified Process, язык UML с расширением WAE – Web Application Extension для Web – приложений) [7-9];

– структурные методы и средства проектирования программного обеспечения (ПО) и баз данных, изначально ориентированные на описание бизнес-процессов (BpWin, ErWin) [11].

Анализ перечисленных средств и реальная практика разработки Web-ресурсов позволяют сделать некоторые выводы, касающиеся роли и места тех или иных средств. Приведем основные из них.

На начальных этапах разработки наиболее адекватна методология целеориентированного проектирования (Goal Directed Design, GDD) Купера [1,3], позволяющая проанализировать цели пользователей и определить базовые функции будущего изделия. Основные положения этой методологии стали классикой Web-разработки. Эти положения подтверждаются в книге Дж. Гарретта [2].

Положения методологии целеориентированного проектирования прекрасно согласуются с положениями по проектированию методологии RUP, хотя они различны по назначению, авторству и времени создания. Первая из них преследует цель только проектирования взаимодействия, но реализует эту цель во всей возможной полноте и с учетом неформального характера предметной области. Вторая позиционируется как методология разработки программного обеспечения на всех его этапах, начиная с проектирования, содержит массу средств документирования процесса разработки ПО (в форме диаграмм и сценариев), но предполагает, что неформальная содержательная часть работы по определению целей и т.д. проделана. Однако благодаря наличию мощных и гибких средств описания различных аспектов процесса разработки эта методология в первую очередь является методологией проектирования. В итоге методология RUP, не преследуя исходно такой цели, предоставляет средства для отображения целей и основных функций компьютерного продукта, а именно, формализм диаграмм вариантов использования (Use Case диаграмм).

Структурные методы (например, [11]) позволяют построить модель внутренней организации системы и потоков данных, позволяющую перейти к программной реализации и построению баз данных. Эти методы созданы значительно раньше RUP и предназначены именно для упомянутых задач моделирования, но решают эти задачи хорошо и очень распространены, в том числе в качестве предметов учебных курсов вузов. Несмотря на то, что структурные методы и RUP относятся к разным подходам (к структурно- и объектно-ориентированному соответственно), противоречий между ними нет; возможно использование диаграмм как одного, так и другого направления.

И, наконец, специализированные средства реализующего методологию RUP языка UML, а именно, расширение Web Application Extension (WAE), позволяют отобразить специфику программирования Web-ресурса, суть которой состоит в обмене данными между клиентом и удаленным сервером, запуске серверных сценариев посредством передачи данных клиента и формировании этими сценариями предъявляемых на клиентской стороне Web-страниц.

В качестве вывода из данного раздела следует отметить, что из всех современных средств проектирования компьютерных продуктов методология RUP наиболее полно охватывает весь процесс разработки и – главное – делает акцент на начальных его этапах: анализе целей и задач и проектировании. В итоге сам методологический подход и комплекс средств проектирования таковы, что область их применения может выходить далеко за рамки разработки программного обеспечения. Использование этой методологии является не только тенденцией, но и практикой Web-проектов и представляется весьма плодотворным.

3.2. Методология RUP и язык UML

Как уже было сказано, методология RUP (Rational Unified Process), предназначенная для выполнения всех этапов разработки проекта по нисходящей схеме и предлагающая систему диаграмм,

описывающих различные аспекты проекта на различных его стадиях, и одно из главных инструментальных воплощений RUP – язык UML (Unified Modeling Language, Унифицированный язык моделирования) являются лидерами в области разработки программных средств.

UML – язык для спецификации, визуализации, конструирования и документирования сложных систем. Включает средства для описания модели предметной области в различных аспектах: от уровня постановки целей до уровня программной реализации, давая возможность отображения как структуры, так и функционирования проектируемой системы [7-9]. Модель (точнее, совокупность моделей) в UML описывается с помощью различных видов диаграмм.

Диаграмма вариантов использования (Use Case diagram) являет собой наиболее общую концептуальную модель сложной системы, которая является исходной для построения всех остальных диаграмм и описывает общие требования к функциональному поведению проектируемой системы.

Диаграмма классов (class diagram) является по своей сути логической моделью, отражающей статические аспекты сложной системы.

Диаграммы, описывающие поведение (behavior diagrams), отражают динамические аспекты функционирования сложной системы.

Сюда входят: диаграмма состояний (statechart diagram) диаграмма деятельности (activity diagram); диаграммы взаимодействия (interaction diagrams); диаграмма последовательности (sequence diagram); диаграмма кооперации (collaboration diagram).

Диаграммы, отображающие программную реализацию (implementation diagrams), служат для представления физических компонентов системы и относятся к физической модели.

Сюда входят диаграммы компонентов (component diagram) и развертывания (deployment diagram).

Де-факто UML стал стандартом в области разработки программного обеспечения в целом.

Методология RUP рекомендует, но жестко не предписывает ни выбора множества диаграмм, ни порядка их использования. Более того, набор диаграмм избыточен, функции ряда диаграмм перекрываются и отдельные элементы проекта могут быть выполнены разными способами. Поэтому при разработке как правило используется только часть диаграмм, выбираемая проектировщиками исходя из их видения задачи и владения методологией. Это – огромное достоинство методологии и языка UML, позволяющее не только строить широкий набор диаграмм, отображающих объект исследования с многих сторон, но и гибко подходить к освоению самой методологии. Это особенно важно для учебного процесса, где заведомо объект автоматизации не может быть слишком сложным, а изучение всех описательных стредств невозможно.

3.3. UML и Web

При разработке Web-ресурсов выбираемые средства одновременно должны отображать и специфику программирования в Интернете. Последняя предполагает обмен данными между клиентом и удаленным сервером, причем:

– на серверной стороне хранятся и выполняются программные сценарии;
– на клиентской стороне пользователю предъявляются Web-страницы, последовательность появления и содержание которых составляют решение задачи пользователя;
– эти страницы формируются серверными сценариями, т.е. представляют собой результат работы программ;
– запуск серверных сценариев инициируется клиентской стороной посредством передачи данных клиента на сервер.

Для разработки Web-приложений UML располагает специальным расширением Web Application Extension (WAE) [9], которое дополняет систему обозначений новыми элементами, позволяющими разработчикам моделировать специфические архитектурные элементы Web-приложений. Это – подмножество диаграмм классов для Web, дающее возможность адекватно и наглядно отображать описанную выше специфику обработки данных в сети. Некоторые элементы WAE приведены на рис.2.

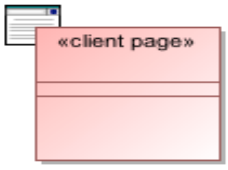
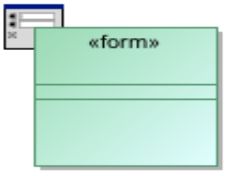
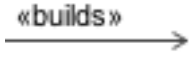
	Серверная страница – Web-страница, содержащая сценарии, выполняемые на сервере.
	Клиентская страница – Web-страница в формате HTML
	Форма – часть клиентской страницы, содержащая поля для ввода данных. При нажатии кнопки передачи формы данные передаются на сервер.
	Ссылка на клиентскую страницу
	Передача данных из полей формы на серверную страницу
	Построение клиентской страницы посредством серверного сценария

Рис. 2. Некоторые элементы расширения Web Application Extension (WAE).

Однако в отличие от сферы разработки традиционных приложений, где UML стал стандартом де-факто, в сфере разработки Web-приложений соответствующие средства практически не упоминаются, не говоря уже о широком использовании (хотя они включены в большинство инструментальных средств, таких как Rational Rose, Magic Draw и др.). Основная масса опубликованных материалов (включая интернет-материалы) посвящена публикации данных и отдельным инструментам разработки, содержащим собственные элементы проектирования. При этом в области практических разработок отсутствие сколько-нибудь универсальных средств проектирования сказывается не так болезненно, как в сфере образования, и может компенсироваться наличием каких-либо корпоративных технологий и инструментов

разработки; личным опытом и способностями конкретных разработчиков; определенностью решаемых задач. Между тем средства WAE адекватны задачам разработки программных компонентов Web-ресурсов, обладают всеми необходимыми выразительными возможностями и введение их в практику преподавания оказывается весьма плодотворным.

4. ПРЕДЛАГАЕМАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ РАЗРАБОТКИ

1. Описание категорий пользователей и вариантов использования ресурса.

Описание дает общее представление о назначении основных компонентов ресурса (реализующих варианты использования) и их функциях. Вариант использования, он же – прецедент (в зависимости от перевода) – по сути отдельный сервис ресурса.

Средства представления результата: Use Case-диаграммы.

2. Анализ требований и переход к проектированию.

Уточнение задачи и введение ограничений (что будем и чего не будем делать и почему). Представление ресурса с точки зрения пользователя.

Определяется назначение, информационный состав, общая структура, взаимосвязь, последовательность и оформление страниц ресурса. Все перечисленное в основных чертах определяет взаимодействие с пользователем, т.е. интерфейс ресурса.

Средства представления результата

Описание ограничений.

Дизайн-макет, или графический образ сайта. Демонстрирует внешний вид сайта. Это статический макет, по сути совокупность графических изображений страниц ресурса. Выполняется как правило в графических редакторах.

3. Проектирование общей структуры ресурса.

Распределение информации по страницам.

Определение основных функциональных компонентов и общей схемы функционирования ресурса, т.е. функциональное моделирование. Моделирование информационных потоков, или потоков данных

Средства представления результата

Структурные диаграммы IDEF0, диаграммы потоков данных DFD; инструмент моделирования – AllFusion Process Modeler 7 (ранее – BpWin).

4. Переход к проектированию Web-ресурса как программного приложения.

Фиксация результатов предшествующей работы в диаграммах и сценариях UML. Реализация общей схемы функционирования в сети (построение динамического макета).

Средства представления результата

Диаграммы последовательности UML; текстовые сценарии работы пользователя; обобщенные диаграммы классов UML для отдельных прецедентов (классы – пока статические страницы).

Динамический макет. Демонстрирует функционирование ресурса в сети с точки зрения пользователя, ограниченное переходом только по основным путям и отсутствием обработки данных. Динамический макет реализуется в коде HTML, где страницы связаны переходами по ссылкам, а обработка данных заменена заглушками (возможно, пустыми – просто текстом).

5. Определение структуры базы данных.

Средства представления результата.

ER-диаграммы (IDEF1X), инструмент ErWin.

6. Определение структуры ресурса как программной клиент-серверной сетевой системы на основе диаграмм последовательности и сценариев: выделение статических компонентов ресурса; определение, какие компоненты динамически формируются программным путем на стороне сервера.

Средства представления результата.

Обобщенные диаграммы классов (class diagrams с использованием WAE-расширения этих диаграмм для Web) отдельных подсистем, описывающие их функционирование и включающие статические компоненты, программные компоненты – сценарии обработки данных на стороне сервера и клиента и страницы, формируемые как результаты работы сценариев.

Диаграммы последовательности sequence diagrams (функционирование акторов и компонентов во времени с учетом формирования ряда компонентов программными сценариями).

7.Поэтапная нисходящая разработка компонентов и отладка ресурса в процессе разработки.

Разработка алгоритмов сценариев обработки данных отдельных компонентов с заменой ряда подзадач заглушками. Программная реализация компонентов. Отладка ресурса. Раскрытие заглушек и т.д. до окончательной отладки.

Средства представления результата.

Диаграммы классов для программных сценариев (WAE-диаграммы); диаграммы последовательности; диаграммы деятельности; выбранный язык программирования.

5. ПРИМЕР ИСПОЛЬЗОВАНИЯ НОТАЦИИ UML И РАСШИРЕНИЯ WAE

Рассматривается пример разработки системы для проведения удаленного тестирования в Интернете. Приводятся только моменты проектирования, наиболее ярко отображающие возможности языка UML и отсутствующие в других средствах: переход от целей к функциям системы и проектирование подсистемы тестирования с отображением специфики ее функционирования как клиент-серверного приложения с обработкой данных на стороне сервера. Соответствующие диаграммы являются основой для программной реализации. Остальные моменты, связанные с созданием макета ресурса и оформлением страниц; нисходящей декомпозиционной разработкой проекта; реализацией ресурса и т.п., опущены.

Ресурс предполагает наличие трех пользователей – тестируемого, эксперта и администратора системы. Для каждого из них должны быть реализованы свои варианты использования, что отображено на Use Case диаграмме ниже. Для иллюстрации процесса проектирования с использованием расширения WAE выбрана подсистема прохождения тестирования, поскольку суть процесса тестирования достаточно прозрачна и не требует особых пояснений.

Подсистема предполагает проведение теста, сохранение данных тестируемого и оценки прохождения теста и возможность анализа полученных результатов. Тестовый материал представлен опросником, предполагающим выбор из набора предлагаемых ответов. Соответствующая информация хранится в базе данных, где для каждого вопроса представлен его текст, множество ответов и правильный вариант ответа. Там же хранятся результаты тестирования. Описание теста, включая правила обработки ответов (ключ), входит в документацию разработки; правила обработки ответов реализуются фрагментом серверного сценария.

Диаграммы представляют собой фрагменты методических материалов учебных курсов автора.

5.1. Диаграмма вариантов использования (Use Case diagram)

Основными персонажами, представляющими целевую аудиторию, являются тестируемый и эксперт – лицо, целью которого является получение оценок тестируемой аудитории по полученным результатам. Обязательный персонаж в системе, предполагающей защиту данных и разграничение прав доступа к ним – администратор. Его цели и роль можно назвать техническими. Use Case диаграмма системы приведена на рис. 3.

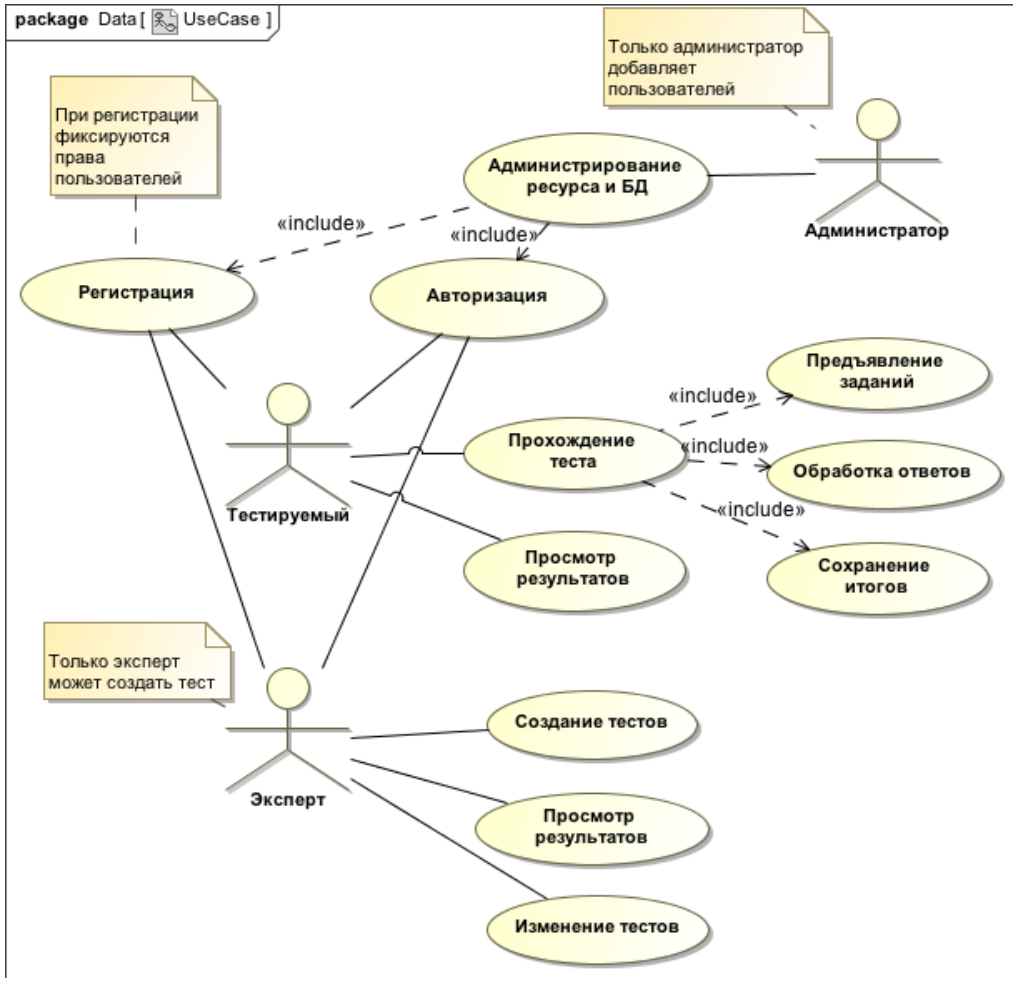


Рис. 3. Диаграмма вариантов использования (UseCase диаграмма) системы интернеттестирования.

5.2. Расширенная диаграмма классов (WAE-диаграмма)

Приводится диаграмма подсистемы тестирования, отображенной на рис. 3 вариантом использования «Прохождение теста» (рис. 4). Это логическая схема функционирования подсистемы, отображающая функционирование и взаимосвязь специфических для Web-ресурса элементов, а также специфику передачи данных между клиентской и серверной частью приложения с помощью элементов WAE (см. рис. 2).

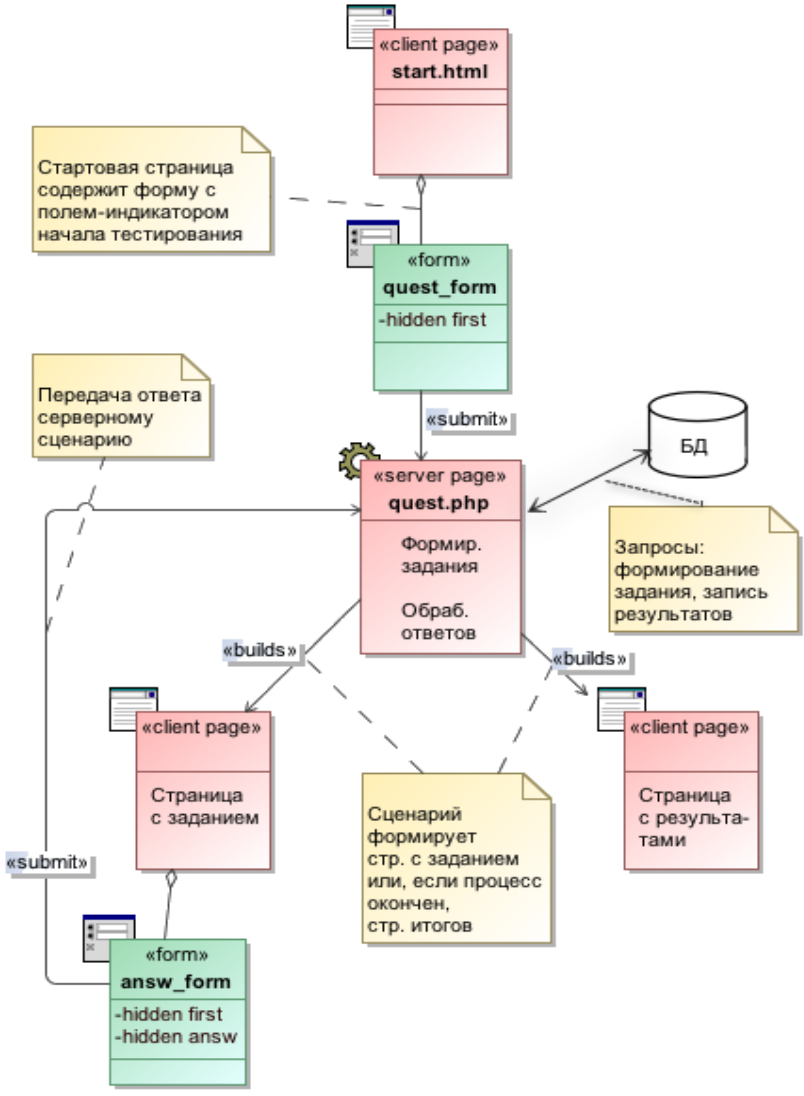


Рис. 4. WAE-диаграмма функционирования подсистемы тестирования.

Процесс тестирования инициируется со статической стартовой страницы start.html, содержащей форму quest_form для передачи начальных значений данных на сервер и запуска программного сценария quest.php, формирующего страницу с первым вопросом.

Далее этот сценарий реализует процесс тестирования:

– формирование страницы с очередным заданием;

– обработку ответа, который передается на сервер этому же сценарию через форму answ_form;

– отслеживание момента исчерпания вопросов;

– обработку ответов согласно ключу теста;

– формирование страницы с результатами по исчерпании вопросов;
– занесение результатов тестирования в базу данных.

6. ЗАКЛЮЧЕНИЕ

Использование элементов RUP и языка UML для проектирования в целом и диаграмм WAE для специфических аспектов описания Web-ресурсов в рамках учебных курсов, а также при выполнении курсовых и дипломных проектов обеспечило мощный инструмент описания Web-проектов как в целом, так и со стороны программной реализации и позволило эффективно представить процесс их функционирования. Однако это далеко не исчерпывает проблем многоаспектности проектирования Web-ресурсов.

Как в плане профессиональной работы, так и в плане преподавания курсов, связанных с технологиями разработки Web-ресурсов, просматриваются, по крайней мере, следующие задачи:
– дальнейший анализ существующих (в том числе новейших) средств Web-разработки;
– использование возможностей аппарата UML применительно к Web-технологиям;
– более полное внедрение в практику преподавания интернет-технологий основных элементов методологии RUP и языка UML .

ЛИТЕРАТУРА

1. Купер, Алан. Алан Купер об интерфейсе. Основы проектирования взаимодействия [Текст] /Алан Купер, Роберт Рейман, Дэвид Кронин. – СПб – М.: Символ, 2009. – 686 с.
2. Гарретт, Джесс. Веб-дизайн: книга Джесса Гарретта. Элементы опыта взаимодействия [Текст] / Джесс Гарретт. – СПб: Символ, 2008. – 180 с.
3. Купер, Алан. Психбольница в руках пациентов. Почему высокие технологии сводят нас с ума и как восстановить душевное равновесие [Текст] /Алан Купер. – СПб – М.: Символ, 2005. – 328 с.
4. Розенфельд Л. Информационная архитектура в Интернете [Текст] / Л. Розенфельд, П. Морвиль. – СПб.: Символ, 2005. – 544 с.
5. Раскин, Джеф. Интерфейс. Новые направления в проектировании компьютерных систем [Текст] /Джеф Раскин. – СПб-М.: Символ, 2005. – 268 с.
6. Константайн, Л. Разработка программного обеспечения [Текст] / Л. Константайн, Л. Локвуд – СПб – М.: Питер, 2004. – 592 с.
7. Арлоу, Д. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование [Текст] / Д. Арлоу, И. Нейштадт И. – СПб: Символ, 2007. – 624 с.
8. Фаулер, Мартин. UML. Основы [Текст] / Мартин Фаулер. – 3-е изд. – СПб: Символ, 2004. – 192 с.
9. Коналлен, Джим. Разработка Web-приложений с использованием UML [Текст] / Джим Коналлен. – М.-СПб-Киев: Вильямс, 2001. – 285 с.
10. Нильсен, Якоб. Веб-дизайн: книга Якоба Нильсена [Текст] / Якоб Нильсен. – СПб.: Символ, 2001. – 504 с.
11. Маклаков, С. В. BpWin и ErWin. CASE-средства разработки информационных систем [Текст] / С. В. Маклаков. – М.: Диалог-МИФИ, 1999. – 256 с.
12. Информационная архитектура [Электронный ресурс]: статья энциклопедии/ Информационная архитектура. – Режим доступа: http://ru.wikipedia.org/wiki/ Информационная архитектура

Работа поступила 19.09.2012