Содержание:

Введение

Совершенствование технических возможностей средств вычислительной техники, развитие коммуникационных средств и технологий управления информационными ресурсами в последние годы привели к появлению более крупных информационных систем. Речь идет о масштабах систем не только относительно объема поддерживаемых информационных ресурсов, но и числа их пользователей. Объем информационных ресурсов Web в настоящие время исчисляется многими миллионами страниц.

Hyper Text Markup Language (HTML) — язык разметки гипертекста — предназначен для написания гипертекстовых документов, публикуемых в World Wide Web.

Гипертекстовый документ — это текстовый файл, имеющий специальные метки, называемые тегами, которые впоследствии опознаются браузером и используются им для отображения содержимого файла па экране компьютера.

Данная тема курсовой работы актуальна в современном мире.

Предметом курсовой работы является язык гипертекстовой разметки.

Цель курсовой работы – изучить теоретический материал по тематике курсовой работы.

Для достижения поставленной цели были выделены следующие задачи :

- проанализировать литературу по теме курсовой работы;

- рассмотреть и изучить понятия: «информационная система», «технология Web»;

- охарактеризовать основные технологии Web, такие как: HTML, XML, XHTML.

Структура курсовой работы: работа состоит из введения, трех глав, заключения, списка литературы.

Первая глава посвящена изучению основных понятий, таких как «информационная система» и «технологии Web».

Вторая глава посвящена изучению технологии Webна основе языков разметки: HTML, XML, XHTML.

В третий главе рассматриваются спецификаций Web-языков.

Глава 1.Определение основных понятий

1.1.Понятие «информационная система» в Web

Информационная система – это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели[1].

Информационные системы классифицируются по разным признакам.

По масштабу информационные системы подразделяются на следующие группы:

- одиночные;

- групповые;

- корпоративные.

Для групповых и корпоративных систем существенно повышаются требования к надежности функционирования и сохранности данных. Эти свойства обеспечиваются поддержкой целостности данных, ссылок и транзакций в серверах баз данных.

Одиночные информационные системы реализуются, как правило, на автономном персональном компьютере (сеть не используется). Такая система может содержать несколько простых приложений, связанных общим информационным фондом, и рассчитана на работу одного пользователя или группы пользователей, разделяющих по времени одно рабочее место. Подобные приложения создаются с помощью так называемых настольных или локальных систем управления базами данных (СУБД). Среди локальных СУБД наиболее известными являются Clarion, Clipper, FoxPro, Paradox, dBase и Qicrosoft Access.

Групповые информационные системы ориентированы на коллективное использование информации членами рабочей группы и чаще всего строятся на базе

локальной вычислительной сети. При разработке таких приложений используются серверы баз данных (называемые также SQL-серверами) для рабочих групп. Существует довольно большое количество различных SQL-серверов, как коммерческих, так и свободно распространяемых. Среди них наиболее известны такие серверы баз данных, как Oracle, DB2, Qicrosoft SQL Server, InterBase, Sybase, Inforqix.

Корпоративные информационные системы являются развитием систем для рабочих групп, они ориентированы на крупные компании и могут поддерживать территориально разнесенные узлы или сети. В основном они имеют иерархическую структуру из нескольких уровней. Для таких систем характерна архитектура клиент-сервер со специализацией серверов или же многоуровневая архитектура. При разработке таких систем могут использоваться те же серверы баз данных, что и при разработке групповых информационных систем. Однако в крупных информационных системах наибольшее распространение получили серверы Oracle, DB2 и Qicrosoft SQL Server.

По сфере применения информационные системы обычно подразделяются на четыре группы:

- системы обработки транзакций;

- системы принятия решений;

- информационно-справочные системы;

- офисные информационные системы.

Системы обработки транзакций, в свою очередь, по оперативности обработки данных, разделяются на пакетные информационные системы и оперативные информационные системы. В информационных системах организационного управления преобладает режим оперативной обработки транзакций – OLTP (OnLine Transaction Processing), для отражения актуального состояния предметной области в любой момент времени, а пакетная обработка занимает весьма ограниченную часть. Для систем OLTP характерен регулярный (возможно, интенсивный) поток довольно простых транзакций, играющих роль заказов, платежей, запросов и т.п. Важными требованиями для них являются:

высокая производительность обработки транзакций;

гарантированная доставка информации при удаленном доступе к БД по телекоммуникациям.

Системы поддержки принятия решений – DSS (Decision Support Systeq) – представляют собой другой тип информационных систем, в которых с помощью довольно сложных запросов производится отбор и анализ данных в различных разрезах: временных, географических и по другим показателям.[2]

Обширный класс информационно-справочных систем основан на гипертекстовых документах и мультимедиа. Наибольшее развитие такие информационные системы получили в сети Интернет.

Класс офисных информационных систем нацелен на перевод бумажных документов в электронный вид, автоматизацию делопроизводства и управление документооборотом.

По способу организации групповые и корпоративные информационные системы подразделяются на следующие классы:

- системы на основе архитектуры файл-сервер;

- системы на основе архитектуры клиент-сервер;

- системы на основе многоуровневой архитектуры;

- системы на основе Интернет/ интеранет-технологий.

Архитектура файл-сервер не имеет сетевого разделения компонентов диалога PS и PL и использует компьютер для функций отображения, что облегчает построение графического интерфейса. Файл-сервер только извлекает данные из файлов, так что дополнительные пользователи и приложения добавляют лишь незначительную нагрузку на центральный процессор. Каждый новый клиент добавляет вычислительную мощность к сети.

Объектами разработки в файл-серверном приложении являются компоненты приложения, определяющие логику диалога PL, а также логику обработки BL и управления данными DL. Разработанное приложение реализуется либо в виде законченного загрузочного модуля, либо в виде специального кода для интерпретации.

Однако такая архитектура имеет существенный недостаток: при выполнении некоторых запросов к базе данных клиенту могут передаваться большие объемы данных, загружая сеть и приводя к непредсказуемости времени реакции. Значительный сетевой трафик особенно сильно сказывается при организации удаленного доступа к базам данных на файл-сервере через низкоскоростные каналы связи. Одним из вариантов устранения данного недостатка является удаленное управление файл-серверным приложением в сети. При этом в локальной сети размещается сервер приложений, совмещенный с телекоммуникационным сервером (обычно называемым сервером доступа), в среде которого выполняются обычные файл-серверные приложения. Особенность состоит в том, что диалоговый ввод-вывод поступает от удаленных клиентов через телекоммуникации. Приложения не должны быть слишком сложными, иначе велика вероятность перегрузки сервера, или же нужна очень мощная платформа для сервера приложений.

Архитектура клиент-сервер предназначена для разрешения проблем файл-серверных приложений путем разделения компонентов приложения и размещения их там, где они будут функционировать наиболее эффективно. Особенностью архитектуры клиент-сервер является использование выделенных серверов баз данных, понимающих запросы на языке структурированных запросов SQL (Structured Query Language) и выполняющих поиск, сортировку и агрегирование информации.

Отличительная черта серверов БД – наличие справочника данных, в котором записана структура БД, ограничения целостности данных, форматы и даже серверные процедуры обработки данных по вызову или по событиям в программе. Объектами разработки в таких приложениях помимо диалога и логики обработки являются, прежде всего, реляционная модель данных и связанный с ней набор SQL-операторов для типовых запросов к базе данных.

Большинство конфигураций клиент-сервер использует двухуровневую модель, в которой клиент обращается к услугам сервера. Предполагается, что диалоговые компоненты PS и PL размещаются на клиенте, что позволяет обеспечить графический интерфейс. Компоненты управления данными DS и FS размещаются на сервере, а диалог (PS, PL), логика BL и DL – на клиенте. Двухуровневое определение архитектуры клиент-сервер использует именно этот вариант: приложение работает у клиента, СУБД – на сервере. Поскольку эта схема предъявляет наименьшие требования к серверу, она обладает наилучшей масштабируемостью. Однако сложные приложения, вызывающие большое

взаимодействие с БД, могут жестко загрузить как клиента, так и сеть. Результаты SQL-запроса должны вернуться клиенту для обработки, потому что там находится логика принятия решения. Такая схема приводит к дополнительному усложнению администрирования приложений, разбросанных по различным клиентским узлам.

Для сокращения нагрузки на сеть и упрощения администрирования приложений компонент BL можно разместить на сервере. При этом вся логика принятия решений оформляется в виде хранимых процедур и выполняется на сервере БД. Хранимая процедура – процедура с операторами SQL для доступа к БД, вызываемая по имени с передачей требуемых параметров и выполняемая на сервере БД. Хранимые процедуры могут компилироваться, что повышает скорость их выполнения и сокращает нагрузку на сервер.

Создание архитектуры клиент-сервер возможно и на основе многотерминальной системы. В этом случае в многозадачной среде сервера приложений выполняются программы пользователей, а клиентские узлы вырождены и представлены терминалами. Подобная схема информационной системы характерна для UNIX. В настоящее время архитектура клиент-сервер получила признание и широкое распространение как способ организации приложений для рабочих групп и информационных систем корпоративного уровня. Подобная организация работы повышает эффективность выполнения приложений за счет использования возможностей сервера БД, разгрузки сети и обеспечения контроля целостности данных.

Двухуровневые схемы архитектуры клиент-сервер могут привести к некоторым проблемам в сложных информационных приложениях с множеством пользователей и запутанной логикой. Решением этих проблем может стать использование многоуровневой архитектуры.

Многоуровневая архитектура стала развитием архитектуры клиент-сервер и в своей классической форме состоит из трех уровней:

- нижний уровень представляет собой приложения клиентов, выделенные для выполнения функций и логики представлений PS и PL и имеющие программный интерфейс для вызова приложения на среднем уровне;

- средний уровень представляет собой сервер приложений, на котором выполняется прикладная логика BL и с которого логика обработки данных DL вызывает операции с базой данных DS;

- верхний уровень представляет собой удаленный специализированный сервер базы данных, выделенный для услуг обработки данных DS и файловых операций FS (без риска использования хранимых процедур).

Подобную концепцию обработки данных пропагандируют, в частности, фирмы Oracle, Sun, Borland и др.

Трехуровневая архитектура позволяет еще больше сбалансировать нагрузку на разные узлы и сеть, а также способствует специализации инструментов для разработки приложений и устраняет недостатки двухуровневой модели клиент-сервер.

Централизация логики приложения упрощает администрирование и сопровождение. Четко разделяются платформы и инструменты для реализации интерфейса и прикладной логики, что позволяет с наибольшей отдачей реализовывать их специалистам узкого профиля. Наконец, изменения прикладной логики не затрагивают интерфейса, и наоборот. Но поскольку границы между компонентами PL, BL и DL размыты, прикладная логика может появиться на всех трех уровнях. Сервер приложений с помощью монитора транзакций обеспечивает интерфейс с клиентами и другими серверами, может управлять транзакциями и гарантировать целостность распределенной базы данных. Средства удаленного вызова процедур наиболее соответствуют идее распределенных вычислений: они обеспечивают из любого узла сети вызов прикладной процедуры, расположенной на другом узле, передачу параметров, удаленную обработку и возврат результатов.

С ростом систем клиент-сервер необходимость трех уровней становится все более очевидной. Продукты для трехзвенной архитектуры, так называемые мониторы транзакций, являются относительно новыми. Эти инструменты в основном ориентированы на среду UNIX, однако прикладные серверы можно строить на базе Qicrosoft Windows NT с использованием вызова удаленных процедур для организации связи клиентов с сервером приложений. На практике в локальной сети могут использоваться смешанные архитектуры (двухуровневые и трехуровневые) с одним и тем же сервером базы данных. С учетом глобальных связей архитектура может иметь больше трех звеньев. В настоящее время появились новые инструментальные средства для гибкой сегментации приложений клиент-сервер по различным узлам сети.

В развитии технологии Интернет/интранет основной акцент пока что делается на разработке инструментальных программных средств. В то же время наблюдается отсутствие развитых средств разработки приложений, работающих с базами данных. Компромиссным решением для создания удобных и простых в использовании и сопровождении информационных систем, эффективно работающих с базами данных, стало объединение Интернет/интранет-технологии с многоуровневой архитектурой. При этом структура информационного приложения приобретает следующий вид: браузер – сервер приложений – сервер баз данных – сервер динамических страниц – web-сервер.

Благодаря интеграции Интернет/интранет-технологии и архитектуры клиент-сервер процесс внедрения и сопровождения корпоративной информационной системы существенно упрощается при сохранении достаточно высокой эффективности и простоты совместного использования информации.

По типу хранимых данных ИС делятся на фактографические и документальные. Фактографические системы предназначены для хранения и обработки структурированных данных в виде чисел и текстов. Над такими данными можно выполнять различные операции. В документальных системах информация представлена в виде документов, состоящих из наименований, описаний, рефератов и текстов. Поиск по неструктурированным данным осуществляется с использованием семантических признаков. Отобранные документы предоставляются пользователю, а обработка данных в таких системах практически не производится.

Характерной особенностью фактографических систем является то, что они работают не с текстом, а с фактическими сведениями, которые представлены в виде записей. Основные компоненты фактографических систем – это сами БД и системы управления БД (СУБД). На базе фактографических систем создаются справочники, системы анализа и управления предприятиями, бухгалтерские системы. СУБД должна предоставлять доступ к данным любым категориям пользователей, включая и тех, которые практически не имеют или не хотят иметь представления:

- о физическом размещении в памяти данных и их описаний;

- о механизмах поиска запрашиваемых данных;

- о проблемах, возникающих при одновременном запросе одних и тех же данных многими пользователями (прикладными программами);

- о способах обеспечения защиты данных от некорректных обновлений и (или) несанкционированного доступа;

- о поддержании баз данных в актуальном состоянии и множестве других функций СУБД.

Документальные системы предназначены для работы с документами на естественном языке: книги, тезисы, статьи. Наиболее распространенным видом ДС являются информационно-поисковые системы (ИПС), которые предназначены для накопления и поиска по различным критериям документов. В состав ИПС входят: программные средства, поисковый массив документов и средства поддержки информационного языка этой системы.

1.2. Понятие «технология Web»

Web-технологии есть концепция работы с информацией. Она отличается следующими особенностями[3]:

- техническая основа Web-технологий – локальные и глобальные сети, часто Интернет;

- применение особого типа тонких клиентов: web-браузеров (типы и история, современное состояние браузеров отдаются на самостоятельное изучение)

преимущественно текстовая и статично-графическая подача информации (ослабление этой тенденции связано с развитием технологий связи и ПО, экспансией медиаконтента);

- изменения в информационных источниках мгновенно отражаются в публикациях;

- число потребителей информации практически не ограничено;

- в публикациях могут содержаться ссылки на другие публикации без ограничения на местоположение и источники материалов;

- активная работа поисковых машин (история, современное состояние и роль поисковых машин отдаются на самостоятельное изучение);

- доставка и тиражирование контента практически бесплатны.

Интернет-технологии в информатике - различного рода практикумы по созданию сайтов, блогов, электронных библиотек и энциклопедий в сети Интернет.

Интернет-сайты[4] - это наборы гипертекстов с гиперссылками, размещаемых на серверах и порталах в компьютерной сети Интернет.

Блоги в Интернет - это интернет-сайты, совмещенные с интерактивными форумами для общения и публикации сообщений и комментариев посетителей сайтов.

Создание сайтов в Интернет - одна из важнейших задач курсов информатики в вузах и школах студентами и школьниками.

Электронные библиотеки и энциклопедии - это новейшие технологии публикации научной и учебной литературы в сети Интернет.

Создание гипертекстовых программ на языке JavaScript - один из лучших примеров обучения программированию, поскольку эти программы можно публиковать и тестировать в Интернет.

Приведенные ниже программы на языке JavaScript были написаны и опубликованы в Интернет и до сих пор работоспособны и доступны для подражания и разработки новых Интернет-учебников.

Язык JavaScript является одним из лучших языков обучения программированию в Интернет.

В основе Интернет и Интернет-технологий лежат гипертексты и сайты, размещаемые в глобальной сети Интернет либо в локальных сетях ЭВМ.

Гипертексты - это тексты со гиперссылками на другие гипертексты, размещенные в Интернет или локальной сети ЭВМ.

Для записи гипертекстов используется язык разметки гипертекстов HTML, который воспринимается всеми браузерами на всех персональных компьютерах.

Язык HTML является международным стандартом, поэтому все гипертексты, единым образом воспринимаются и единым образом отображаются на всех персональных компьютерах во всем мире.

Для подготовки гипертекстов обычно используются визуальные гипертекстовые редакторы, в которых сразу видно - как будет выглядеть гипертенст на ЭВМ и возможна вставка гиперссылок на сайты в Интернет.

Одним из лучших визуальных гипертекстовых редакторов является свободный офисный редактор Writer в свободном офисном пакете Open Office.

Глава 2. Технологии WEB

2.1. Язык гипертекстовой разметки HTML

Hyper Text Markup Language (HTML) — язык разметки гипертекста — предназначен для написания гипертекстовых документов, публикуемых в World Wide Web[5].

Гипертекстовый документ — это текстовый файл, имеющий специальные метки, называемые тегами, которые впоследствии опознаются браузером и используются им для отображения содержимого файла па экране компьютера.

С помощью этих меток можно выделять заголовки документа, изменять цвет, размер и начертание букв, вставлять графические изображения и таблицы. Но основным преимуществом гипертекста перед обычным текстом является возможность добавления к содержимому документа гиперссылок — специальных конструкций языка HTML, которые позволяют щелчком мыши перейти к просмотру другого документа.

HTML-документ состоит из двух частей: собственно текста, т. е. данных, составляющих содержимое документа, и тегов — специальных конструкций языка HTML, используемых для разметки документа и управляющих его отображением. Теги языка HTML определяют, в каком виде будет представлен текст, какие его компоненты будут исполнять роль гипертекстовых ссылок, какие графические или мультимедийные объекты должны быть включены в документ.

Графическая и звуковая информация, включаемая в HTML-документ, хранится в отдельных файлах. Программы просмотра HTML-документов (браузеры) интерпретируют флаги разметки и располагают текст и графику на экране соответствующим образом. Для файлов, содержащих HTML-документы приняты расширения .htm или .html.

В большинстве случаев теги используются парами. Пара состоит из открывающего <имя_тега> и закрывающего </имя_тега> тегов. Действие любого парного тега начинается с того места, где встретился открывающий тег, и заканчивается при

встрече соответствующего закрывающего тега. Часто пару, состоящую из открывающего и закрывающего тегов, называют контейнером, а часть текста, окаймленную открывающим и закрывающим тегом, — элементом.

Последовательность символов, составляющая текст может состоять из пробелов, табуляций, символов перехода на новую строку, символов возврата каретки, букв, знаков препинания, цифр, и специальных символов (например #, +, $, @), за исключением следующих четырех символов, имеющих в HTML специальный смысл: < (меньше), > (больше), & (амперсанд) и " (двойная кавычка). Если необходимо включить в текст какой-либо из этих символов, то следует закодировать его особой последовательностью символов.

Самым главным из тегов HTML является одноименный тег <html>. Он всегда открывает документ, так же, как тег </html> должен непременно стоять в последней его строке. Эти теги обозначают, что находящиеся между ними строки представляют единый гипертекстовый документ. Без этих тегов браузер или другая программа просмотра не в состоянии идентифицировать формат документа и правильно его интерпретировать.

HTML-документ состоит из двух частей: заголовок (head) и тела (body), расположенных в следующем порядке:

<html>

<head> Заголовок документа </head>

<body> Тело документа </body>

</html>

Чаще всего в заголовок документа включают парный тег <title>... </title>, определяющий название документа. Многие программы просмотра используют его как заголовок окна, в котором выводят документ. Программы, индексирующие документы в сети Интернет, используют название для идентификации страницы. Хорошее название должно быть достаточно длинным для того, чтобы можно было корректно указать соответствующую страницу, и в то же время оно должно помещаться в заголовке окна. Название документа вписывается между открывающим и закрывающим тегами.

Тело документа является обязательным элементом, так как в нем располагается весь материал документа. Тело документа размещается между тегами <body> и

</body>. Все, что размещено между этими тегами, интерпретируется браузером в соответствии с правилами языка HTML позволяющими корректно отображать страницу на экране монитора.

Текст в HTML разделяется на абзацы при помощи тега . Он размещается в начале каждого абзаца, и программа просмотра, встречая его, отделяет абзацы друг от друга пустой строкой. Использование закрывающего тега необязательно.

Если требуется «разорвать» текст, перенеся его остаток на новую строку, при этом, не выделяя нового абзаца, используется тег разрыва строки
. Он заставляет программу просмотра выводить стоящие после него символы с новой строки. В отличие от тега абзаца, тег
 не добавляет пустую строку. У этого тега нет парного закрывающего тега.

Язык HTML поддерживает логическое н физическое форматирование содержимого документа. Логическое форматирование указывает на назначение данного фрагмента текста, а физическое форматирование задает его внешний вид.

При использовании логического форматирования текста браузером выделяются различные части текста в соответствии со структурой документа. Чтобы отобразить название, используется один из тегов заголовка. Заголовки в типичном документе разделяются по уровням. Язык HTML позволяет задать шесть уровней заголовков: h1 (заголовок первого уровня), h2, h3, h4, h5 и h6. Заголовок первого уровня имеет обычно больший размер и насыщенность по сравнению с заголовком второго уровня. Пример использования тегов заголовков:

<hl>l. Название главы</hl>

<h2>l.l. Название раздела</h2>

Теги физического форматирования непосредственно задают вид текста на экране браузера, например пара выделяет текст полужирным начертанием, <u></u> задает подчеркивание текста, управляет шрифтом текста.

Тег <img> вставляет изображение в документ, как если бы оно было просто одним большим символом. Пример применения тега:

<img src = "picture.gif">

Для создания гипертекстовой ссылки используется пара тегов <a>... . Фрагмент текста, изображение или любой другой объект, расположенный между этими тегами, отображается в окне браузера как гипертекстовая ссылка. Активация такого объекта приводит к загрузке в окно браузера нового документа или к отображению другой части текущей Web-страницы. Гипертекстовая ссылка формируется с помощью выражения:

ссылка на документ

Href здесь является обязательным атрибутом, значение которого и есть URL-адрес запрашиваемого ресурса. Кавычки в задании значения атрибута href не обязательны. Если задается ссылка на документ на другом сервере, то вид гиперссылки такой:

Фотография 11-А

С помощью различных тегов можно рисовать таблицы, форматировать текст, вставлять в документ изображения, видео- , звуковые файлы и прочее.

2.2. Расширяемый язык разметки XML

Расширяемый язык разметки (Extensible Markup Language, аббревиатура - XML) описывает класс объектов XML document, а также частично описывает работу компьютерных программ, обрабатывающих объекты с данными, реализующими этот класс. XML - это прикладной уровень или усеченная форма SGML, Стандартного Обобщенного языка разметки [ISO 8879][6]. По своему построению, XML документ является полноценным SGML документом.

XML документы состоят из единиц размещения, называемых сущностями, которые содержат разобранные или неразобранные данные. Разобранные данные состоят из набора символов, часть которых образуют символьные данные, часть - разметку. Разметка образует описание схемы размещения и логической структуры документа. Язык XML дает механизм создания ограничений для указанной схемы размещения и логической структуры.

XML — текстовый формат, предназначенный для хранения структурированных данных (взамен существующих файлов баз данных), для обмена информацией между программами, а также для создания на его основе более специализированных языков разметки (например, XHTML), иногда называемых

словарями.

Целью создания XML было обеспечение совместимости при передаче структурированных данных между разными системами обработки информации, особенно при передаче таких данных через Интернет. Словари, основанные на XML (например, RDF, RSS, MathML, XHTML, SVG), сами по себе формально описаны, что позволяет программно изменять и проверять документы на основе этих словарей, не зная их семантики, то есть не зная смыслового значения элементов. Важной особенностью XML также является применение так называемых пространств имён (англ. namespace).

Стандартом определены два уровня правильности документа XML:

- Правильно построенный (Well-formed). Правильно построенный документ соответствует всем общим правилам синтаксиса XML, применимым к любому XML-документу. И если, например, начальный тег не имеет соответствующего ему конечного тега, то это неправильно построенный документ XML. Документ, который неправильно построен, не может считаться документом XML; XML-процессор (парсер) не должен обрабатывать его обычным образом и обязан классифицировать ситуацию как фатальная ошибка.

- Действительный (Valid). Действительный документ дополнительно соответствует некоторым семантическим правилам. Это более строгая дополнительная проверка корректности документа на соответствие заранее определённым, но уже внешним правилам, в целях минимизации количества ошибок, например, структуры и состава данного, конкретного документа или семейства документов. Эти правила могут быть разработаны как самим пользователем, так и сторонними разработчиками, например, разработчиками словарей или стандартов обмена данными.

XML — это описанная в текстовом формате иерархическая структура, предназначенная для хранения любых данных. Визуально структура может быть представлена как дерево элементов. Элементы XML описываются тэгами[7].

Первая строка XML-документа называется объявление XML (англ. XML declaration) — это необязательная строка, указывающая версию стандарта XML (обычно это 1.0), также здесь может быть указана кодировка символов и внешние зависимости.

<?xmlversion="1.0" encoding="UTF-8"?>

Спецификация требует, чтобы процессоры XML обязательно поддерживали Юникод-кодировки UTF-8 и UTF-16 (UTF-32 не обязателен).

Важнейшее обязательное синтаксическое требование заключается в том, что документ имеет только один корневой элемент (англ. rootelement) (так же иногда называемый элемент документа (англ. documentelement)). Это означает, что текст или другие данные всего документа должны быть расположены между единственным начальным корневым тегом и соответствующим ему конечным тегом.

В любом месте дерева может быть размещен элемент-комментарий. XML-комментарии размещаются внутри специального тега, начинающегося с символов<!-- и заканчивающегося символами -->. Два знака дефис (--) внутри комментария присутствовать не могут.

Остальная часть этого XML-документа состоит из вложенных элементов, некоторые из которых имеют атрибуты и содержимое. Элемент обычно состоит из открывающего и закрывающего тегов, обрамляющих текст и другие элементы. Открывающий тег состоит из имени элемента в угловых скобках, например, <step>, а закрывающий тег состоит из того же имени в угловых скобках, но перед именем ещё добавляется косая черта, например, </step>.

Сущностью (англ. entity) в XML называются именованные данные, обычно текстовые, в частности, спецсимволы. Ссылка на сущность (англ. entityreferences) указывается в том месте, где должна быть сущность и состоит из амперсанда (&), имени сущности и точки с запятой (;).

2.3. Расширяемый язык разметки гипертекста XHTML

В 1997г. официальной рекомендацией W3C становится HTML версии 4.0, явившийся плодом множества компромиссов. Данная версия вобрала в себя все визуальные возможности HTML 3.2, но в большинстве своем они были провозглашены не рекомендованными к использованию (deprecated). HTML 4 – это, по сути, не один стандарт, а целых три. Иначе говоря, в рамках HTML 4 определяются 3 типа документов: Strict (строгий, предписывающий четкое соблюдение идеологии структурной, логической разметки и подразумевающий отказ от большинства возможностей визуального представления данных), Transitional (переходный,

намного более либеральный, оставляющий право пользоваться сомнительным наследием эпохи HTML 3.2) и Frameset (предназначенный для страниц, использующих фреймы). Наиболее предпочтительным, разумеется, является использование типа документов Strict.

Повсеместное распространение "веб-стандартов" тормозилось всегда низким качеством реальной поддержки спецификаций сколько-либо распространенными браузерами. Сегодня уже можно констатировать, что все современные браузеры работают с актуальными рекомендациями W3C на вполне приемлемом уровне. Наиболее безупречно поддерживает их Mozilla Firefox, его догоняет Opera 8.x и 9.0, а хуже всего на этом фоне выглядит Microsoft Internet Explorer 7.0

Последняя версия HTML 4.01 стала стандартом в 1999 году, после этого разработки этого языка прекратились, так как он фактически исчерпал все свои возможности. Появились технологии, более удовлетворяющие потребности современных пользователей, а также языки программирования, позволяющие создавать динамические страницы, обращаться с запросами к базам данных, более совершенные языки разметки. На данный момент переемником HTML считается XHTML — расширяемый язык гипертекстовой разметки.

XHTML (англ. Extensible Hypertext Markup Language — расширяемый язык разметки гипертекста)[8] — язык разметки веб-страниц, по возможностям сопоставимый с HTML, однако является подмножеством XML. В отличие от HTML, XML позволяет создавать собственные теги и таким образом формировать собственную структуру документа. В противоположность XML, HTML гораздо более строго определенный язык разметки с ограниченным набором тегов. В любом случае, общий характер XML позволяет рассматривать HTML-документы как XML-документы с набором тегов для отображения в веб-браузерах. Однако, старые стандарты HTML не до конца совместимы с XML. Например, в HTML необязательно закрывать тег, то есть закрывающий тег можно опускать. Чтобы устранить разрыв между этими двумя языками разметки и был разработан XHTML. По существу это обычный HTML, в который добавили синтаксические правила XML для создания well-formed документов. Так что веб-страницы станут XML-совместимыми, а веб-разработчики познакомятся с синтаксисом XML.

Как и HTML, XHTML соответствует спецификации SGML. Вариант XHTML 1.1 одобрен в качестве Рекомендации Консорциума Всемирной паутины (W3C) 31 мая 2001 года.

На смену сложному и противоречивому SGML (Standard Generalized Markup Language) в качестве универсального стандарта для хранения любых структурированных данных приходит расширяемый язык разметки XML — eXtensible Markup Language, отличающийся более строгими правилами. HTML 4 стал в определенном смысле «тупиковой веткой» — это последняя версия HTML, основанная на SGML.

Развитием HTML 4 стал «расширяемый» (eXtensible) язык разметки гипертекста — XHTML 1.0, по сути своей явившийся лишь переформулировкой стандарта HTML 4.01 в соответствии с правилами XML 1.0. XHTML 1.0, за исключением ряда нюансов, в точности повторяет функциональность HTML 4.01 и включает в себя предусмотренные последним три типа документов: Strict, Transitional и Frameset.

XHTML 1.1, полностью упразднивший типы документов Transitional и Frameset и утвердивший еще некоторые ощутимые нововведения, уже несколько дальше от привычного HTML, но на практике пока используется довольно редко. XHTML 2.0 уходит в еще больший отрыв от совместимости с HTML — но эта версия пока еще находится в работе и в разряд официальных рекомендаций W3C перейдет не так скоро.

Для XHTML можно применять множество технологий разработанных для XML. Например, XSLT и XPath. Анализ XHTML проще и быстрее, чем HTML. Поскольку синтаксис XML строже, чем SGML, обработка XHTML возможна даже на мобильных телефонах с малыми ресурсами.

се элементы в XHTML должны быть закрыты. Теги, которые не имеют закрывающего тега (например, <img> или
) должны иметь на конце / (например,
).

У атрибутов обязательно должно быть значение. Например, следует писать <option selected="selected"> или <td nowrap="nowrap">.

Все значения атрибутов обязательно должны быть заключены в двойные, либо одинарные кавычки.

Имена тегов и атрибутов должны быть записаны строчными буквами (например, вместо <IMG ALT="" />).

XHTML гораздо строже относится к ошибкам в коде: нельзя делать «перехлест» тегов (например, запрещено   ), < и & везде, даже в

URL, должны замещаться их мнемонической подстановкой. По рекомендации W3C браузеры, встретив ошибку в XHTML, должны сообщить о ней и не обрабатывать документ. Для HTML браузеры должны были попытаться понять, что хотел сказать автор.

Кодировкой по умолчанию является UTF-8 (в отличие от HTML, где кодировкой по умолчанию является ISO 8859-1)

Отличия переходного (англ. transitional) XHTML от HTML незначительны и предназначены лишь для приведения его в соответствие с XML. Остальные версии отличаются лишь набором тегов.

Практически все современные браузеры поддерживают XHTML. Он также совместим и со старыми браузерами, т. к. в основе XHTML лежит HTML. Такая совместимость, к сожалению, в числе прочего, замедляет процесс перехода от HTML к XHTML.

Валидным документом (т. е. отвечающим всем правилам) XHTML-документом считается документ, удовлетворяющий технической спецификации. В идеале, все браузеры должны следовать веб-стандартам и, в соответствии с ними, валидные документы должны отображаться во всех браузерах на всех платформах. Валидация XHTML-документа рекомендована даже несмотря на то, что она не гарантирует кросс-браузерной совместимости. Документ может быть проверен на соответствие спецификации с помощью онлайновой Службы валидации разметки W3C. Валидация обнаружит и разъяснит ошибки в XHTML-разметке.

Глава 3.Спецификации технологий Web

3.1. О спецификации HTML

Эта спецификация определяет HyperText Markup Language (HTML) - гипертекстовый язык разметки, язык World Wide Web. Здесь определён HTML 4.01, являющийся субверсией HTML 4[9]. В дополнение к возможностям работы с текстом, мультимедиа и гипертекстом предыдущих версий HTML (HTML 3.2 и HTML 2.0), HTML 4 поддерживает большее количество опций мультимедиа, языков скриптов, каскадных таблиц стилей, лучшие возможности печати и большую доступность документов для людей с ограниченными возможностями. HTML 4 также является

большим шагом в направлении интернационализации документов с целью сделать Web действительно World Wide (всемирным).

HTML 4 - это SGML приложение, соответствующее Международному Стандарту ISO 8879 -- Standard Generalized Markup Language.

Этот документ специфицирует HTML 4.01, являющийся частью спецификации линии HTML 4. Первой версией HTML 4 был HTML 4.0, опубликованный 18 декабря 1997 и пересмотренный 24 апреля 1998.

Эта спецификация является первыми рекомендациями по HTML 4.01. Она включает дополнительные изменения после версии HTML 4.0 от 24 апреля.

Внесены некоторые изменения в DTD/ОТД. Этот документ объявляет предыдущую версию HTML 4.0 устаревшей, хотя W3C оставляет её спецификацию и ОТД доступными на сайте W3C.

Этот документ был рассмотрен членами W3C и других заинтересованных сторон и утверждён Директором как Рекомендации W3C. Это неизменяемый документ, он может использоваться как справочный материал или цитироваться в других документах. Задачей W3C является привлечение внимания к Рекомендациям и этой спецификации и её широкое распространение. Это расширит функциональные возможности Web.

W3C рекомендует создание пользовательскими агентами (ПА) и авторами (в частности, утилитами разработки) документов HTML 4.01, а не HTML 4.0.

W3C рекомендует создавать документы HTML 4 вместо документов HTML 3.2. Из соображений обратной совместимости, W3C также рекомендует, чтобы утилиты, интерпретирующие HTML 4, продолжали поддерживать HTML 3.2 и HTML 2.0.

За информацией о следующем поколении HTML, "The Extensible HyperText Markup Language", обращайтесь на W3C HTML Activity и к списку W3C Technical Reports.

Этот документ создан как часть W3C HTML Activity. Цели Рабочей Группы HTML обсуждаются в Хартии Рабочей Группы HTML.

Список текущих Рекомендаций W3C и другая техническая документация находятся по адресу: http://www.w3.org/TR.

Публичная дискуссия о возможностях HTML проходит на (архивы).

Английская версия этой спецификации является единственным нормативным документом[10]. Однако переводы этого документа можно найти по адресу: http://www.w3.org/MarkUp/html4-updates/translations.

Сценарии - ключевое средство интерактивного взаимодействия в вебе, однако в спецификации HTML механизмы их создания не были определены достаточно хорошо. В результате браузеры обрабатывают сценарии во многом несогласованно. К счастью, в HTML 4.0 реализов ан более полный подход.

Одно из наиболее важных изменений заключается в спецификации сценариев на веб-страницах. Элемент meta позволяет определить заданный по умолчанию язык сценариев для всех сценариев страницы. Это делается с помощью спецификации значения в заголовке Conte nзаданное по умолчанию значение, браузер попытается извлечь его из соответствующего поля фактического заголовка HTTP, установленного веб-сервером. После определения языка сценария для сценариев указывается тип со держимого MIME. Чаще всего типом содержим ог(сценарий VBScript) или text/javascript (сценарий JavaScript), однако можно использовать и другие допустимые типы, такие, как text/tcl (сценарий TCL)[11].

Чтобы указать в качестве заданного по умолчанию языка JavaScript, используйте выражение:

<meta http-equiv="Content-Script-Type" content="text/javascript">

С помощью элемента script можно переопределить используемый по умолчанию язык сценариев в любом месте страницы, однако атрибут language в данной спецификации HTML применять не рекомендуется. Вместо этого нужно использовать атрибут type, позволяющий задат ь тип содержимого MIME для сценариев, например, . Между тем, не следует забывать, что при указании атрибута type некоторые старые браузеры не смогут правильно обрабатывать ваши сценарии. Пока все это урегулируется, нужно имет ь в виду, что, хотя атрибут type вполне законен, он не является обязательным.

Еще одно важное изменение, касающееся сценариев, состоит в том, что большинство элементов HTML теперь поддерживают широкий спектр атрибутов событий. Событие происходит автоматически при определенном условии. Некоторые события вызываются действием пользов ателя (например, нажатием клавиши или щелчком кнопкой мыши), другие - браузерами (например, когда браузер завершает загрузку изображения). Атрибуты событий позволяют задать условия, при которых должен обнаруживаться элемент. Например, можно создать

в таб лице ячейки, активизируемые щелчком мыши, абзацы текста, выделяемые при перемещении на них курсора, и многое другое.

Существует несколько способов работы с событиями. Для этого удобно использовать такие атрибуты событий, как onclick и onkeypress. Они позволяют распознать событие и передать его функции в сценарии. С другой стороны, если вы хотите проверить наличие ко ннепосредственно распознающую и обрабатывающую его. Следующий пример показывает, как использовать атрибут события onmouseover с тегом :

Main

Когда указатель мыши перемещается на фиксированный текст, вызывается функция show(), и этот текст передается ей. Как вы обнаружите на практике, лучше всего использовать атрибуты именно так, когда одна и та же функция вызывается многократно.

Если применяются события, специфические для HTML 4.0, то полезно проверить браузер на совместимость. Обычно несовместимые браузеры игнорируют атрибуты событий, которых они не понимают, и в результате ожидаемая вами функция поддерживаться не будет. Проект иуя веб-страницу с учетом этой особенности, можно постараться избежать подобных проблем.

3.2. О спецификации XML

Стандарт Расширяемого Языка Разметки (XML) задает формат данных, но вопрос о том, какие данные следует сохранять, оставляет совершенно открытым. Формат хранения данных весьма гибок, предусматривая отображение произвольно сложных типов данных[12].

С точки зрения программиста, спецификации, объясняющей программе, что происходит в XML файле, вполне достаточно, чтобы можно было его проанализировать и подтвердить или не подтвердить его правильность, вполне последовательно и надежно. Однако это не дает гарантии понятности. Зная, что XML файл корректный, не обязательно означает, что определенная программа поймет его. Даже подтверждение, что XML файл содержит определенный тип данных, не обязательно поможет вам написать программу, интерпретирующую эти

данные. В конце концов, это не является целью XML!

Стандарт XML во многом похож на спецификацию YACC (генератор синтаксических анализаторов) - программу, которая определяет формат для выражения основных правил, а затем создает анализатор для этих правил. XML файлы могут иметь семантику практически настолько же отличную друг от друга, насколько отличны друг от друга различные языки программирования, разработанные при помощи YACC. Фактически XML это не формат файлов, а платформа для написания форматов.

Это создает интересную разновидность полустандарта. Стандарт XML настолько широк и применим практически для всего, и вполне вероятно, что средства для разработки XML совместимы, однако нет повода ожидать, что все данные, фактически сохраненные в XML файлах, будут совместимы со всем, чем угодно. Циники утверждают, что первичная цель XML - позволить совершенно внутренним данным претендовать на совместимость со стандартом.

Спецификация SGML определенного языка разметки называется Описание Типа Документа, или DTD. Например, спецификация HTML существует как группа SGML DTD, каждое из которых определяет, что именно является или не является допустимым в данной разновидности HTML. Спецификация XML позволяет пользователям создавать свои собственные DTD, с некоторыми ограничениями. Однако различным XML документам не требуется иметь каких-либо общих элементов.

Это поднимает вопрос о различии между корректным XML и допустимым XML. Документ, который подчиняется сугубо синтаксическим требованиям (компоновка знаков больше и меньше и использование сущностей (entities)) XML, корректный вне зависимости от содержащихся в нем сущностей или элементов. "Корректность" не зависит от отдельного DTD, это особенность, которую XML документ может иметь без ссылки на DTD, или даже если документ нарушает данный DTD. Документ, который соответствует описаниям элементов конкретного DTD, называется "допустимым".

Допустимый XML может, в принципе, быть использован определенным приложением, которое владеет данным DTD. Знание того, что файл XML корректный, не столь полезно; это гораздо точнее, чем сказать: "формат наших данных - ASCII", но сталкивается с теми же принципиальными ограничениями. Однако даже допустимый XML файл может быть написан для DTD, который может

быть недоступен для пользователей, делая его совершенно бесполезным. Мог бы существовать DTD, под которым следующий элемент XML был бы допустимым:

<zz><y/><x>183af892</x><w/></zz>

Спецификации XML сейчас не всегда выражаются через DTD; Microsoft® предложил новый способ выражения тех же типов понятий, называемый XML Схемами. XML Схема стала стандартом W3C в 2001 году. Сами XML Схемы написаны в XML и предназначены, чтобы разобраться с некоторыми проблемами, которые люди имели с традиционными DTD. Возможно, наиболее значительным является больший простор для определения типов данных и семантического содержания. Например, заданные "даты", которые предписывают порядок таких компонентов, как год месяц и день. XML Схемы дают больше семантического содержания, чем традиционные DTD, помогая заполнить пробел между "допустимым" и "понятным". Они также могут определить дальнейшие ограничения на содержание, такие как приемлемый диапазон для числовых значений.

Основным сдерживающим фактором в продвижении XML технологии в Web на сегодняшний день является отсутствие полной поддержки этого формата всеми производителями броузеров - программ, наиболее часто используемых на стороне клиента. Выходом из создавшейся ситуации может стать вариант, при котором обработкой XML документов занимается серверная сторона Используя любой существующий XML-анализатор, можно формировать необходимую информацию уже на сервере и посылать клиенту нормальный HTML-документ. Однако такой способ, конечно, менее гибок, и позволяет использовать XML технологию лишь для хранения структурированной информации, но не для ее динамического изменения на стороне клиента.

В августе 1997 RFC 2376 были утверждены MIME типы для XML-ресурсов: text/xml и application/xml. Поэтому XML документы могут передаваться по HTTP и отображаться программой просмотра также, как и обычные HTML- страницы. Для этого нужно немного изменить конфигурацию Web-сервера (в Apache - добавить в файл mime.types строчку "text/xml xml ddt"), а на стороне клиента иметь броузер, поддерживающий стилевые таблицы или JavaScript. Сегодня такими броузерами являются Microsoft Internet Explorer 5, первый броузер, поддерживающий спецификацию XML 1.0 и стилевые таблицы XSL; броузер Amaya, предлагаемый консорциумом специально для тестовых целей (http://www.w3.org/Amaya/User/BinDist.html) и поддерживающий практически все разрабатываемые стандарты W3C. Поддержка XML также планируется в будущих

версиях Netscape Navigator[13].

3.3. О спецификации XHTML

В настоящей спецификации определяется XHTML 1.0, переформулировка HTML 4 в виде приложения XML 1.0, и три DTD, соответствующих типам, определяемым HTML 4. Семантика элементов и их атрибутов определена в рекомендации W3C HTML 4. Данная семантика представляет собой основу для будущего расширения языка XHTML. Совместимость с существующими пользовательскими агентами HTML обеспечивается путем соответствия следующим нескольким требованиям.

Разработчики документов и создатели пользовательских агентов постоянно открывают новые способы выражения своих идей в новой разметке. В XML ввод новых элементов или атрибутов достаточно прост. Семейство XHTML разработано так, чтобы принимать расширения путем модулей и технологий XHTML для разработки новых соответствующих XHTML модулей (описанных в готовящейся спецификации Модуляризации XHTML). Модули позволят комбинировать существующие и новые наборы функций при разработке содержимого и создании новых пользовательских агентов.

Постоянно вводятся альтернативные методы доступа в Интернет. По некоторым оценкам, в 2002 году 75% обращений к документам в Интернет будет выполняться с альтернативных платформ. Семейство XHTML создавалось с учетом общей совместимости пользовательских агентов. С помощью нового механизма профилирования пользовательских агентов и документов серверы, прокси и пользовательские агенты смогут преобразовывать содержимое наилучшим образом. В конечном счете станет возможной разработка соответствующего XHTML содержимого, пригодного для любого соответствующего XHTML пользовательского агента[14].

Спецификация XHTML 1.0 была одобрена в качестве рекомендации консорциума Всемирной паутины в январе 2000 года. В августе 2002 года была опубликована вторая редакция спецификации — XHTML 1.1. Параллельно полным ходом началась разработка XHTML 2.0, призванного стать новым уровнем представления документов во Всемирной сети. Разработчики пошли на довольно смелый шаг — нарушение обратной совместимости, но нововведения, которые они собирались внести, стоили того. XHTML 2.0 содержит спецификации Xforms, Xframes, призванные заменить стандартные HTML-формы и фреймы соответственно, ML

Events — API для управления DOM-структурой документа, встроенную поддержку модулей Ruby character и многое другое. Работа шла полным ходом, но было несколько обстоятельств, совершенно не радующих авторов спецификаций. Если коротко, XHTML просто не получил должного распространения.

Во-первых, огорчали веб-разработчики, которые после вольницы HTML никак не хотели принимать новые правила в полном объеме. Расставлять в нужном месте кавычки и сущности оказалось просто непосильной задачей. Что там говорить про XHTML, если и со стандартами HTML4 веб-верстальщики обходились достаточно вольно. И что самое возмутительное — производители браузеров активно им в этом потакали!

И именно в этом заключалась вторая проблема. На самом деле все довольно понятно — те, кто делали браузеры, просто не могли допустить, чтобы какой-либо значимый контент в них был не доступен пользователю из-за каких-то неясных принципиальных соображений, и, надо сказать, они были по-своему правы (ну в самом деле, веб нам нужен для общения с миром, а не для т ого, чтобы все атрибуты были снабжены кавычками!). В результате наиболее популярные браузеры имели два режима отображения XHTML-документов, причем по умолчанию обычно работал «нестрогий» режим, при котором огрехи в разметке милосердно прощались. Хуже того, безусловно, самый распространенный на тот момент браузер Internet Explorer вообще не реагировал на MIME-тип application/xhtml и не имел в своем составе парсера обработки XHTML-документов вплоть до восьмой версии.

Главная причина неудачи повсеместного внедрения XHTML довольно проста. Строгие правила валидации, атрибуты, взятые в кавычки, закрытые одиночные теги… все это, может, и хорошо, но нужны эти тонкости в основном самим разработчикам и блюстителям стандарта, но никак не пользователям, которым, строго говоря, дел нет до всех этих тонкостей. И никак не создателям веб-контента, которым эти правила попросту ничего не дают, кроме несильной головной боли. В общем, сложилось что-то вроде революционной ситуации — создателям стандарт не нужен, а потребителям он не нужен тем более. А что всем им было нужно? Живой интерактивный веб-контент, воплощающий социальные потребности современного человека. Плоский мир HTML с этим справлялся плохо, и к концу 90-х на веб-странице появились не относящиеся к языку разметки компоненты[15].

Заключение

Часто приходится слышать, что WWW - это очень просто. Однако за этой кажущейся простотой скрывается хорошо продуманная сложная система. При этом следует заметить, что система бурно развивается.

Глобальная информатизация общества приводит к тому, что потребность в информации, растет с каждым новым пользователем сети. При этом задачей специалистов в области информационных технологий обеспечить пользователей полной и достоверной информацией путем простого и удобного для пользователей доступа к накопленным массивам данных.

Гипертекстовая технология позволяет быстро и точно осуществить поиск необходимой информации не только в рамках отдельных документов на компьютера и web-страниц, созданных с их использованием, но и в локальных вычислительных сетях и всемирной сети Internet.

Идея гипертекстовой информационной системы состоит в том, что пользователь имеет возможность просматривать документы (страницы текста) в том порядке, в котором ему это больше нравится, а не последовательно, как это принято при чтении книг.

Простой на первый взгляд механизм построения ссылок оказывается довольно сложной задачей, т. к. можно построить статические ссылки, динамические ссылки, ассоциированные с документом в целом или только с отдельными его частями.

Расширение понятия гипертекста происходит за счет других информационных ресурсов, включая графику, аудио- и видео-информацию, до понятия гипермедиа.

Безусловно у электронных гипертекстовых систем есть множество преимуществ, по сравнению с обычными печатными текстами. Например:

- двунаправленный характер ссылок, по сравнению с направленными только вперед ссылками обычных текстов;

- комментарии к тексту могут быть сделаны в самом тексте, не нарушая в то же время его целостность;

- однажды пройденный маршрут легко запоминается и может быть автоматически пройден многократно; это равносильно тому, что читатель формирует свой личный текст;

- электронная форма представления текста позволяет автоматизировать процесс формирования массива ключевых слов путем автоматического компьютерного анализа текста, и прочие преимущества.

Но при всем многообразии преимуществ прослеживается и ряд недостатков:

- у пользователя теряется ощущение собственного положения и движения в многомерном документе и появляется дополнительная умственная нагрузка для выбора оптимального пути изучения материала;

- чтобы электронная гипертекстовая система была эффективной для пользователя, электронное информационное пространство должно быть достаточно обширным, а применяемые технические средства достаточно мощными;

- создание электронных гипертекстовых систем требует новых подходов к проблеме защиты авторского права: во-первых, это совместный труд большого числа специалистов (в том числе программистов); во-вторых, развитый аппарат электронных гипертекстовых систем обеспечивает возможность непосредственного включения в данную систему текстов других авторов.

При подготовке данной курсовой работы был изучен материал в котором поднимались вопросы гипертекстовой технологии.

Подготовка этого курсового проекта очень помогла мне понять суть гипертекстовой технологии, необходимость гипертекста в жизни современного активного человека.

Материал данной курсовой работы дает необходимые сведения о гипертексте и пищу для размышлений, о нем можно рассказать еще много интересного и познавательного, для этого следует обратиться к специальным статьям, книгам и прочей информации, посвященной различным аспектам вопросов, посвященным гипертекстовым технологиям.

Список использованной литературы:

1) http://daxnow.narod.ru/index/0-33

2) http://www.webmasterwiki.ru/Web-texnologii-ChtoJetoTakoe

3) http://shkolo.ru/html-yazyik-razmetki-giperteksta/

4) http://www.solarix.ru/for_developers/cpp/xml/xml_1_0.shtml

5) http://metodpro.ru/index.php?type_page&katalog&id=1223&met11

6) http://www.cssblok.ru/tech/xhtml.html

7) http://www.intuit.ru/studies/courses/11/11/lecture/325

8) http://spec.piramidin.com/html401/index.htm

9) http://www.codenet.ru/webmast/html/html4.php

10) http://www.ibm.com/developerworks/ru/library/pa-spec18/

11) http://www.codenet.ru/webmast/xml/part2.php

12) https://www.opennet.ru/docs/RUS/XHTML1/

13) http://html5ru.com/xhtml-standart-dlya-standartizatorov.html

1. http://daxnow.narod.ru/index/0-33 ↑

2. http://daxnow.narod.ru/index/0-33 ↑

3. http://www.webmasterwiki.ru/Web-texnologii-ChtoJetoTakoe ↑

4. http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%
%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B
↑

5. http://shkolo.ru/html-yazyik-razmetki-giperteksta/ ↑

6. http://www.solarix.ru/for_developers/cpp/xml/xml_1_0.shtml ↑

7. http://metodpro.ru/index.php?type_page&katalog&id=1223&met11 ↑
8. http://www.cssblok.ru/tech/xhtml.html ↑
9. http://www.intuit.ru/studies/courses/11/11/lecture/325 ↑
10. http://spec.piramidin.com/html401/index.htm ↑
11. http://www.codenet.ru/webmast/html/html4.php ↑
12. http://www.ibm.com/developerworks/ru/library/pa-spec18/ ↑
13. http://www.codenet.ru/webmast/xml/part2.php ↑
14. https://www.opennet.ru/docs/RUS/XHTML1/ ↑
15. http://html5ru.com/xhtml-standart-dlya-standartizatorov.html ↑