**УДК 004.051**

**Обзор средств разработки динамических web–сайтов**

**Меньшикова Лариса Валерьевна –** к.ф.**–**м.н., доцент кафедры информационных технологий и управляющих систем

**Исрафилов Мурад Анатольевич** – студент 5 курса группы Информационные системы и технологии Заочного отделения

Государственное бюджетное образовательное учреждение

высшего образования Московской области «Технологический университет»

имени дважды Героя Советского Союза, летчика **–** космонавта А.А. Леонова,

г. Королев, Московская область

**Найденова Дария Михайловна** – студентка 1–го курса Факультета вычислительной математики и кибернетики

Московский Государственный университет им. М.В. Ломоносова, г. Москва

Данная статья продолжает обзор средств проектирования веб–сайтов [1, C.88], дополняя рассмотрение создания централизованной ИАС технологии Интернет/Интранет – технологией Хранилищ данных. Приведены технологии создания динамических сайтов, их функции, преимущества и недостатки, в том числе по сравнению со статическими. Обсуждается архитектура, информационная безопасность и используемые в динамических и статических сайтах протоколы. На основе рекомендаций по выбору средства проектирования динамических веб–сайтов приведены два примера обоснованного выбора средств разработки: 1) системы заказов столика и меню в ресторане и 2) интернет–магазина автозапчастей.

Проекты информатизации, проектирование централизованных информационно–аналитических систем, веб–сайт, хранилище данных.

**Dynamic website development tools review**

**Menshikova Larisa –** Candidate of Рhysical and Mathematical Sciences, Associate Professor of the Department of Information Technologies and Control Systems

**Israfilov Murad** – 5st year student of the Information systems and technologies of Distance learning Department

State Educational Institution of Higher Education Moscow Region

«University of technology» named after twice Hero of the Soviet Union,

pilot – cosmonaut A.A.Leonova, Korolev, Moscow region

**Naydenovа Darya** – 1st year student of the Department of Applied Mathematics

and Cybernetics of Moscow State University named after M.V.Lomonosov, Moscow

This article continues the review of website design tools [1, C.88], adding Data Warehousing technology to the centralized Internet/Intranet IAS technology. Сreating dynamic sites technologies, their functions, advantages and disadvantages, including in comparison with static ones, are presented. Architecture, information security, protocols as dynamic such as static site are discussed. Based on the recommendations for choosing a dynamic website design tool, two examples of a reasonable choice are provided.

IT–projects, design of centralized information and analytical systems, website, Data Warehouse.

**Введение. Технологии хранилищ данных для динамических сайтов**

Преобразование противоречивой и разрозненной информационной среды любого предприятия в структурированную – первый шаг к открытой и гибкой архитектуре корпоративного хранилища данных (КХД), создание которого является необходимым условием для последующей успешной разработки систем поддержки принятия решения крупного предприятия [2,3,4]. Архитектура КХД определяет трудоемкость реализации возможности создания централизованного непротиворечивого информационного пространства для своевременного принятия решений на основе разработанной модели данных, отвечающей описанию предметной области деятельности предприятия. Именно от архитектуры КХД зависит устойчивость, надежность, а так же скорость работы каждой информационно–аналитической системы (ИАС), как системы–источника данных, так и системы–потребителя информации.

В ходе проекта создания единой централизованной ИАС предприятия на основе КХД необходимо соблюдать ряд организационных принципов:

* поддерживать постоянную заинтересованность руководства в проекте и обеспечивать поддержку со стороны топ–менеджера – куратора проекта для гарантии квалифицированного решения спорных вопросов административным способом в минимально короткие сроки;
* централизовать сбор данных, предпочтительно – атомарных и не сгруппированных в целях минимизации задержки данных при сборе и частичного устранения рассогласования измерений и фактов;
* проводить работы с участием подразделений, участвующих в бизнес–процессах автоматизируемой области деятельности, для корректного анализа в ИАС действий, производимых с данными КХД.

При этом следует отметить, что участие бизнес–специалистов при создании КХД необходимо только на этапах анализа и разработки логической модели данных.

Разработку хранилища данных можно проводить в несколько этапов:

* анализ бизнес–задач в целях определения необходимых для решения этих задач данных;
* разработка модели данных с учетом агрегации и консолидации данных, а так же устранение противоречивых фактов в информационной среде предприятия;
* разработка базы данных на основе построенной модели;
* первичная загрузка данных в КХД;
* организация процессов дозагрузки данных в КХД;
* организация выгрузки данных из КХД в витрины данных (наличие этого этапа зависит от концептуальной архитектуры хранилища).

Создание модели данных КХД возможно только при непосредственном участии специалистов предметной области. Никакие внешние консультанты, имеющие опыт создания ИАС и КХД, но не знающие досконально особенности деятельности предприятия, на котором проводится автоматизация, не могут провести анализ корректно. Типовые системы аналитики и шаблоны хранилищ данных могут способствовать сокращению сроков создания системы и затрат на ее реализацию.

Следование вышеуказанным организационным принципам позволяет наиболее эффективно решить многие проблемы при построении хранилища данных и создании централизованной ИАС, и установить эффективное партнерство между информатизацией и бизнесом в части понятийного аппарата и формализации проблем, которые решаются в рамках проводимой автоматизации. Кроме того, отсутствие понимания у конечного пользователя, какой именно результат должен быть получен от проводимого проекта, и почему этого нельзя добиться с использованием той или иной технологии, еще одна причина появления противоречий между бизнес–подразделениями и ИТ–департаментом.

Организационные принципы создания единой централизованной ИАС предприятия распространяются и на проект создания КХД. С технической точки зрения при создании КХД можно определить архитектурные принципы создания КХД, к которым относятся принципы разработки модели данных, принципы работы механизма запросов к данным, принципы построения рабочих хранилищ, принципы построения метаданных, принципы масштабируемости и принципы управления хранилищем данных.

При любом выборе концептуальной архитектуры КХД [5,6] должны быть решены следующие вопросы:

* движение данных от источников к хранилищу;
* распределение и обработка данных для хранилища;
* описание метаданных, проектирование модели и собственно ХД;
* обработка запросов и представление информации.

Архитектура хранилища данных делится на три части:

* архитектуру доступа;
* архитектуру данных;
* технологическую архитектуру.

Архитектура доступа представляет собой описание способов, с помощью которых конечные пользователи получают доступ к данным из хранилища данных, а также к данным, помещаемым в КХД.

Архитектура данных – это описание структуры и жизненного цикла данных.

Технологическая архитектура – это описание технологических деталей КХД и его взаимодействия с унаследованными системами и используемыми на предприятии информационными технологиями.

Фактически же формулировка архитектурных принципов зависит от ИТ–среды, в которой должно быть построено КХД. Если придерживаться вышеуказанных организационных принципов стоит всегда, то архитектурные принципы нельзя рассматривать в качестве полного множества принципов для произвольного проекта. Кроме того, принципы должны соответствовать конкретному проекту, методологии разработки КХД и корпоративной культуре предприятия. Нужно учитывать следующие факторы: величину проектного бюджета, количество ИТ – специалистов, выделенных для участия в проекте, время, выделенное на выполнение проекта, структуру организации и ее опыт в области создания децентрализованных ИАС и хранилищ данных.

При оценке времени и ресурсов, необходимых для построения КХД, необходимо учесть следующие факторы:

* количество и типы пользователей КХД;
* расположение пользователей и количество мест их расположения;
* количество и типы источников данных;
* количество и типы баз данных–источников;
* количество и типы баз данных КХД;
* частоту загрузки данных в хранилище;
* выделенное время подготовки и загрузки данных;
* необходимое время подготовки и загрузки данных;
* сложность типичных незапланированных запросов к хранилищу;
* сложность стандартных отчетов.

Все вышеописанные посылы применимы и к созданию динамических вебсайтов (как прототипов централизованных информационно–аналитических систем предприятий и организаций), для которых все чаще используется трехуровневая архитектура «клиент–сервер» с БД/СУБД на третьем уровне, о которых и идет речь в нашей статье.

**Создание веб–сайта с использованием трехуровневой архитектуры.**

Несмотря на несомненные преимущества статических сайтов, средства разработки которых были рассмотрены в предыдущем обзоре [1], особенно в части информационной безопасности и устойчивости к взлому и DDos –атакам, большая часть ресурсов интернет занята онлайн–магазинами, которые содержат информацию о продаваемом товаре (некоторые типовые данные о товаре, его цене, артикуле и т.п., категорированные некоторым рубрикатором). Такие данные нецелесообразно размещать на статических Интернет–сайтах, так как эти данные не требуются на страницах предоставления в полном объеме и хранятся в базах данных или в витринах данных, то есть на ресурсах, которые являются элементами хранилища данных. Кроме того, что всегда требуется предоставлять эти данные не в полном объеме, а только их часть, часто динамические сайты разрабатываются не программистами, как статические сайты – HTML–документы с использованием CSS или JavaScript, а самими владельцами малого бизнеса, продающими эти товары в онлайн–магазинах.

***Двухуровневая архитектура статических сайтов*** (рис.1) относится к ***«клиент–серверной»***. Приведем основные протоколы передачи данных (табл.1) статических сайтов по сети Интернет/Интранет. Во–первых, это определяющий взаимодействие веб–клиента с веб–сервером протокол передачи гипертекста HTTP (прикладной уровень, уровень представления данных и сеансовый уровень 7–уровневой модели ISO/OSI). Во–вторых, это управляющий отдельными сеансами связи между серверами и клиентами протокол управления передачей данных TCP (транспортный уровень 7–уровневой модели ISO/OSI). И, в–третьих, это протокол межсетевого обмена IP (сетевой уровень 7–уровневой модели ISO/OSI). На канальном и физическом уровнях 7–уровневой модели ISO/OSI используются те протоколы, которые отвечают сетям передачи данных (СПД) и технологии доступа в Интернет/Интранет соответственно в локальной вычислительной сети (ЛВС) предприятия.



**Рисунок 1– Общая архитектура статического и динамического веб–сайта**

***Трехуровневая архитектура динамических сайтов*** (рис.1)так же относится к ***«клиент–серверной»*** и содержит наряду с клиентскими приложениями, с которыми работают пользователи, и серверами приложений, которые есть и в двухуровневой модели «клиент–сервер», третий уровень. Это сервер баз данных, с которым работают сервера приложений, и к которому не имеют прямого доступа пользователи. Перед тем как перейти к рассмотрению протоколов, используемых для передачи данных на динамические сайты (табл.1) из хранилищ данных, средствам и языкам их разработки, отметим, что в зависимости от выбранной категоризации известны и другие виды архитектур программного обеспечения: многослойная (к которой можно отнести двухуровневую), многоуровневая (к которой можно отнести как двухуровневую, так и трехуровневую), сервисно–ориентированная и микросервисная [7, C.115]. Помимо архитектуры «клиент–сервер» есть еще и архитектура «файл–сервер», которая так же может использоваться в сети Интернет/Интранет, и много других видов архитектур приложений, работающих в сети Интернет/Интранет, но это мы оставили за рамками нашего обзора. Кроме того, при создании динамических сайтов используется объектно–ориентированный метод (методология, парадигма) создания приложений, а не структурный метод, как в случае создания статических сайтов, из–за наличия в трехуровневой архитектуре сервера баз данных.

**Протоколы передачи информации динамических веб–сайтов.**

Следует отметить, что при разработке динамических сайтов часто вместо протокола НТТP используется протокол FTP, инкапсулированный в НТТP для прохождения прокси–сервера.

**Таблица 1 – Протоколы компьютерных сетей**

|  | **Уровни модели ISO/OSI** | **Уровни модели TCP/IP** | **Единицы измерения**  **данных/ОС или сеть** | **Функция** | **Варианты протоколов** |
| --- | --- | --- | --- | --- | --- |
| 1 | уровень приложений | уровень приложений | поток/ОС | прикладная задача | HTTP, HTTPS, SMTP, SNTP, DNS, RIP, FTP, FTTPS,  FTP–SSL, Telnet, POP3, NTP  и др. |
| 2 | уровень представления данных | поток/ОС | представление данных, шифрование и т.д. | MIME, SSL, Finger  и др. |
| 3 | сеансовый уровень | поток/ОС | взаимодействие хостов на уровне ОС | DNS, LDAP, iSCSI, RPC, ISAKMP, IPDC, NetBIOS и др. |
| 4 | транспортный уровень | транспортный уровень | сегмент/ОС | соединение точка–точка, контроль передачи данных | TCP, UDP Mobile IP, DCCP, SCTP, RUDP, BGP4  и др. |
| 5 | сетевой уровень | межсетевой уровень | пакет/сеть | логическая адресация и маршрутизация пакетов | IP, ICMP, IGMP, ARP, VRRP, RIP, PIM, GRE, DVMRP и др. |
| 6 | канальный уровень | уровень сетевого доступа | фрейм/сеть | физическая адресация | Ethernet (IEEE 802.3 и др.), Frame Relay, IP/MPLS (вместо ATM), Token Ring, ARP, DHCP и др. |
| 7 | физический уровень | бит/сеть | кодирование и передача данных по физическому каналу | Ethernet (IEEE 802.3 и др.), Frame Relay, IP–VPN (вместо ATM), Token Ring, витая пара и др. |

Это объясняется тем, что несмотря на то, что оба протокола FTP и НТТP, работающие на прикладном уровне над TCP транспортного уровня и IP сетевого уровеня, используются для передачи с клиента на сервер («заливки») и для взятия клиентом с сервера («скачивания») как текста, так и бинарных данных в сетях Интернет/Интранет, характеристики [8] их сильно отличаются (табл.2). Основное отличие – скорость загрузки данных, особенно для низкоскоростных сетей передачи данных, по протоколу FTP намного выше. Кроме того, при передаче большого количества файлов БД небольшого объема, использование заголовков в НТТP так же замедляет передачу данных.

**Таблица 2 – Параметры протоколов передачи данных**

| **Характеристика** | **Протокол передачи гипертекста**  **НТТP** | **Протокол передачи файлов по сети FTP** | **FTP через SSH**  **(туннелирование обычной FTP–сессии через SSH–соединение – т.н. «безопасный FTP»)** |
| --- | --- | --- | --- |
| **Скорость передачи** | средняя, так как нет метаописаний и  не требуется перекодировка | высокая, так как повторное использование установленных соединений повышает производительность TCP | низкая |
| **Дата появления** | 1997 год | 1971 год | 1997 год |
| **«Заливка»** | + | + | + |
| **Форматы ASCII, EBCDIC или бинарный** | все | только бинарный | только бинарный |
| **Заголовки, в которых содержатся метаданные** | **+** | **–** | **–** |
| **Пайплайны/ конвейеры, ускоряющие загрузку файла** | конвейеры | пайплайны | пайплайны |
| **Количество запросов при скачивании одного большого файла или в ходе одной сессии** | 1 | множество | множество |
| **Разделение потоков передачи команд и данных** | **–** | **+** | + |
| **Файрволы и NAT** | **–** | **+** | + |
| **Активный и пассивный режимы** | **–** | **+** | – |
| **Зашифрованные управляющие соединения** | +  (HTTPS) | +  (FTTPS, FTP–SSL и их гибридные версии) | **+** |
| **Схемы авторизации** | Логин/пароль, направленные методом проверки не в виде обычного текста | Логин/пароль, направленные в виде обычного текста | Логин/пароль, |
| **«Скачивание»** | + | + | + |
| **Диапазоны/**  **возобновление скачивания** | Восстановление соединениятолько при скачивании | Восстановление соединений, прерванных при скачивании или закачке файлов | + |
| **Постоянные соединения** | Использует одно соединение для передачи любого количества файлов | Создает новый сеанс связи для каждого файла | + |
| **Кодирование HTTP–чанков** | **+** | **–** | **–** |
| **Сжатие** | Выбор алгоритма сжатия данных одновременно сервером и клиентом | Официальное использование только RLE–сжатия | + |
| **Технология передачи данных с сервера на сервер как от клиента FXP** | **–** | **+** | + |
| **IP** | + | + | + |
| **Виртуальный хостинг на основе имени** | **+** | **–** | **–** |
| **Просмотр каталогов** | Нет | Есть возможность просмотра списка файлов в папке, не скачивая их | + |
| **Поддержка прокси** | **+** | **–** | – |

Перейдем теперь к средствам разработки (в части создания самих сайтов и баз данных, работающих под управлением систем управления базами данных на третьем уровне архитектуры «клиент–сервер»), языкам разработки динамических веб–сайтов, а так же к средствам их защиты.

**Информационная безопасность динамических веб–сайтов.**

Выше уже было отмечено, что информационная безопасность и устойчивость к взлому и DDos –атакам динамических сайтов значительно ниже, чем у статических сайтов прежде всего потому, что способ и технологии передачи данных на динамические сайты предполагают передачу данных, а не файлов, как в случае статических сайтов, а, следовательно, в эту схему легче встроить вредоносное программное обеспечение [9, C.22], вирусы, рекламу и рекламоподобные вирусы, и требуется постоянным мониторинг динамического веб–приложения, что само по себе является непростой задачей [10, C.125].

Одна из принятых категоризаций уровней защиты в сети Интернет/Интранет основана на 7–уровневой модели ISO/OSI, протоколы в рамках которой мы только что рассмотрели. Так как одна категоризация, полностью описывающая все инциденты информационной безопасности, отличается от другой только методом разбиения множества инцидентов на прямую сумму непересекающихся подмножеств, то можно без ограничения общности изложения считать эту категоризацию единственной и достоверной, если она описывает защиту от всех видов атак, существующих в настоящее время: как глобальных, так и локальных. Для начала отметим, что **локальные атаки**, как правило, направлены на кражу информации или перехват управления на отдельном, нужном мошенникам, сервере и являются взломом самого сайта или хостинга, где он расположен. Их можно разделить на вредоносную рекламу, SQL–инъекции, LDAP – инъекция, XPath –инъекция, SSI – инъекция, переадресацию результатов работы поискового сервера, межсайтовый скриптинг (XSS), виртуальный хостинг, HTML–инъекции (CSS) и использование уязвимостей программ социальных форумов. **Глобальные атаки** направлены на несколько сайтов и ставят своей целью заражение всех посетителей этих сайтов тем или иным видом вируса, таким как, например, «троянский конь» или «червь», методом поиска «фишингом», то есть предприняыв попытку загрузить вредоносное ПО на ваш компьютер или в сеть (ис помощью относящегося к нему средства доставки «спуфинга», то есть принудив вас сообщить конфиденциальную или финансовую информацию злоумышленнику), и т.п. К сетевым атакам, наряду с так же можно отнести социальную инженерию, обман или мошенничество типа «скам», к которому относят самые разные события, например просьба от «сотрудника вашего банка» продиктовать SMS–код, чтобы получить доступ к онлайн–банку, который можно принять за «спам» – рассылку рекламы. Таким образом, cетевые атаки используют 3 и 4 уровни на 7–уровневой модели ISO/OSI и называются «уровня L3/L4», а глобальные – 7 уровень 7–уровневой модели ISO/OSI и называются «уровня L7». DDoS – атаки, то есть «распределенный отказ в обслуживании» с использованием большого числа взломанных или контролируемых источников, происходят на 3, 4, 6 и 7 уровнях 7–уровневой модели ISO/OSI, в то время, как DoS – атаки, то есть «отказ в обслуживании», просто делают недоступным сайт или приложение на сайте для пользователей (уровень L7) или передаваемые пакеты данных (то есть относятся к уровню L3).

Распределение функций безопасности по уровням 7–уровневой модели ISO/OSI [11] без указания механизмов реализации функций защиты (шифрование, электронная подпись, управление доступом, проверка целостности, идентификация, управление маршрутизацией и т.п.), а также мер защиты, представлено в Таблице 3.

**Таблица 3 – Распределение функций безопасности по уровням модели ISO/OSI**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Функция безопасности** | **Уровень 7–уровневой модели OSI/OSI, на котором можно реализовать данную функцию, отмечен «+»** | | | | | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| **Аутентификация** | – | – | + | + | – | – | + |
| **Управление доступом** | – | – | + | + | – | – | + |
| **Конфиденциальность соединения** | + | + | + | + | – | + | + |
| **Конфиденциальность вне соединения** | – | + | + | + | – | + | + |
| **Избирательная конфиденциальность** | – | – | – | – | – | + | + |
| **Конфиденциальность трафика** | + | – | + | – | – | – | + |
| **Целостность с восстановлением** | – | – | – | + | – | – | + |
| **Целостность без восстановления** | – | – | + | + | – | – | + |
| **Избирательная целостность** | – | – | – | – | – | – | + |
| **Целостность вне соединения** | – | – | + | + | – | – | + |
| **Безотказность** | – | – | – | – | – | – | + |

Частично степень защиты зависит от выбора веб–браузера, которые мы рассмотрели в предыдущем обзоре [1, C.105], а так же установленного на компьютерах антивирусного программного обеспечения, если оно не отключено. Кроме того, хостинг–провайдер всегда использует средства информационной безопасности, которые распространяются и на сайты, расположенные в его зоне. Владелец хостинга, обеспечивает анализ входящего трафика, блокировку подозрительных запросов, фильтрацию входящего трафика, блокировку адресов или разрешения на доступ по геолокации и т.д., частично обеспечивая защиту на уровнях L3 (вплоть до маскировки IP–адреса) и L7 (в части ботов и фильтрации заголовков, сравнения с шаблонами скриптов и файлов cookie).

Отдельно опишем атаки, цель которых – динамические сайты (табл.4).

**Таблица 4 – Атаки и уязвимости динамических сайтов**

| **Атака** | **Уязвимость** | **Уровень модели OSI/OSI, какое ПО подвержено атаке на этом уровне** |
| --- | --- | --- |
| Внедрение операторов XPath (XPath –инъекция) – атака на сервер, который создает запросы на языке XPath на основе данных, которые вводит пользователь | Переполнение буфера (Buffer Overflow) – уязвимость, которая позволяет изменить путь исполнения программы с помощью перезаписи данных в памяти системы | L5, например, скрипты |
| Внедрение серверных сценариев (SSI – инъекция) – передача исполняемого кода, который будет выполнен на веб–сервере | Выполнение команд ОС – уязвимость, которая позволяет направлять команды операционной системе на веб–сервере с помощью манипуляций входными данными | L5 и L6, операционная система |
| Внедрение операторов SQL (SQL – инъекция) – атака на сервера, которые создают SQL–запросы к серверам систем управления базами данных на основе того, что вводит пользователь | SQLi – уязвимость, которая позволяет атакующему использовать фрагмент вредоносного кода на языке структурированных запросов SQL для манипулирования базой данных и получения доступа к потенциально ценной информации | L7, подвержены любые сайты, которые взаимодействуют с базой данных на языке запросов SQL |
| Внедрение операторов LDAP (LDAP – инъекция) – атаки на сервера, которые создают запросы к службе LDAP на основе данных пользователей | Уязвимость в изменении LDAP – оператора, после чего процесс будет выполнятся с такими же привилегиями, как и компонент, который выполняет команду (СУБД, Веб–сервер). | L5, скрипты |
| Атака на функции форматирования строк (Format String Attack) – путь исполнения программы модифицируется методом перезаписи областей памяти с помощью функций форматирования символьных переменных | Уязвимости форматирования | L7, форматные строки программного кода |

Конечно, большинство атак уровней L5, L6 и L7, направлены на веб–сайты, написанные на HTML, как наиболее распространенном языке написания и представления (отображения) веб–сайтов.

**Языки и программные средства разработки динамических веб–сайтов.**

Наиболее популярными технологиями (средами разработки) создания динамических веб–страниц являются: CGI, PHP, ASP, ASP.NET, JSP, Cold Fusion, AJAX, Python, CSS, базы данных MySQL, PostgressSQL, Oracle, Access и т.д.

Основным языком разработки статических сайтов является HTML, который поддерживается всеми рассмотренными в обзоре [1, C.101] программами разработки веб–сайтов. В динамических сайтах используются на сервере различные языки [11] с интерпретатором для обеспечения подстановки данных, а не компилятором: PHP, Python, Ruby, Go, Perl, а также ASP.NET и JSP, серверный JavaScript и т.д., а на стороне клиента соответствующие им: AJAX, DHTML JavaScrip, Visual Basic и XMLHttpRequest. Вообще говоря, при разработке динамических сайтов используется две технологии – вышеуказанная – на основе интерпретируемых сценариев, и с помощью компилируемых модулей (CGI, написанные в основном на языке Perl), которые мы оставим за рамками нашей статьи.

Более подробный обзор языков программирования динамических сайтов, технологий и средств их разработки можно увидеть на сайте «Словари и энциклопедии на Академике» [11], на основе которого составлен ряд сводных таблиц, интегрирующих эту информацию в том же виде, что и в предыдущем обзоре [1, C.89].

Сначала дополним Таблицу 5 программ разработки сайтов обзора средств проектирования веб–сайтов [1, C.89] средствами разработки динамических сайтов с учетом, что ряд средств предыдущего обзора пригодны и для динамических сайтов ***(Notepad++, Komodo Edit, Atom, Visual Studio Code, Vim нами уже были рассмотрены)***.

**Таблица 5 – Программы разработки веб–сайтов**

| **Название** | **Описание** | **Основные функции и/или особен–ности**  **ПО** | **Поддер–живаемые**  **языки** | **Компания/**  **сайт/ год первого релиза/год последнего релиза (если поддержка завершилась)/ОС** | **Преимущества** | **Недостатки** |
| --- | --- | --- | --- | --- | --- | --- |
| Apache | Cвободное программ–ное обеспече–ние для создания и размеще–ния веб–сервера | Cоздание и размеще–ние веб–сервера | ASP.NET,C++, CFML,  ЕСМА  Script,  Java, JavaScript,  Scala,  PHP,  Python,  и т.д. | http://www.  fusebox.org/  1995/  Microsoft Windows, Unix–подобные системы | Динамическая модульная структура, содержащая  более 500 модулей различной функциональ–ности | 1. Скорость обработки запросов ниже, чем у конкурентов; 2.гибкость веб–сервера иногда уменьшает производи–тельность: каждый раз считывается несколько конфигура–ционных файлов на сервере, расходуя системные ресурсы и время. |
| Fusebox | Программ–ное обеспече–ние для создания сайта, способ–ствующее структу–рирова–нию кода разработ–чиками с помощью набора простых соглаше–ний. | Создание большого, много–функцио–нального приложе–ния | CFML,  PHP | http://www.fusebox.org/  1997/  2012/  Microsoft Windows | Использова–ние шаблонов проектирова–ния и методов объектно–ориентированного программирования;  простое в освоении | 2012 год – разработка прекращена |
| Horde | Бесплатное групповое программ–ное обеспечение разработки приложений на основе PHP с совсмест–ным доступом к коду нескольких человек | Почтовый клиент Horde IMP; пакет группо–вого программ–ного обеспече–ния (календарь, заметки, задачи, файловый менеджер) | PHP | https://github.  com/horde/base  2001/  кроссплат–форменный | Есть программное обеспечение для отслеживания времени и задач | Отсутствует описание на русском языке |
| Zeta Components | Высоко–качествен– ная библиотека слабо связанных компонент общего назначения для разработки приложе–ний на основе PHP | Механизм шаблонов, инструмен–ты базы данных;  компонент MVC | PHP | https://irc.  freenode.net/  #zetacomponents  /2010/ | Командная строка; различные инструменты; библиотека визуализации диаграмм; универсальный компонент преобразова–ния документов | Отсутствует описание на русском языке |

На основе рассмотрения основных функций и особенностей ПО и в соответствии с задачами динамического сайта можно выбрать программу и язык разработки. Следует отметить, что выбор программы был сделан ранее, и в таблицу попали только программы, содержащие Apache и язык PHP.

**Системы управления базами данных разработки динамических веб–сайтов.**

И, наконец, остановимся на БД и СУБД, которые можно использовать при разработке динамических веб–сайтов.

Для этого дополним Таблицу 6 систем управления контентом [1, C.92] средствами обработки информации динамических сайтов – клиент–серверными базами данных с учетом, что ряд средств, которые работают совместно с БД/СУБД, в предыдущем обзоре мы уже занесли в аналогичную таблицу***: Joomla и Drupal.***

**Таблица 6 – Средства обработки информации для динамических сайтов**

| **Название** | **Описание** | **Язык запросов/ БД** | **Отличие от других СУБД** | **Сайт/** **год первого релиза/ОС** | **Преиму–щества** | **Недостатки** |
| --- | --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MySQL | Бесплатная реляционная СУБД, разработку и поддержку которой осуществляет фирма Oracle | SQL/  С, С++ | Максималь–ный размер таблиц – до  4 Гбайт | http://mysql.com  1995/  Linux, Microsoft Windows, macOS, FreeBSD, Solaris и Unix–подобные ОС | 1.Хорошо докумен–тирована;  2. может работать с другими БД;  3. мас–штабиру–ема | 1.Ограничение на количество столбцов, которое можно добавить в одну таблицу (409);  2. автомати–ческие функции других СУБД отсутствуют – например, создание инкрементных резервных копий |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| PostgreSQL | Бесплатная объектно–реляционная СУБД | SQL/С | 1.Максималь–ный размер таблиц – до  32 Тбайт ;  2. поддержка [индексов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B4%D0%B5%D0%BA%D1%81_(%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) следующих типов: [B–дерево](https://ru.wikipedia.org/wiki/B-%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE), [хеш](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88-%D1%82%D0%B0%D0%B1%D0%BB%D0%B8%D1%86%D0%B0), [GiST](https://ru.wikipedia.org/wiki/GiST), [GIN](https://ru.wikipedia.org/wiki/GIN), [BRIN](https://ru.wikipedia.org/wiki/BRIN), [Bloom](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%BB%D1%8C%D1%82%D1%80_%D0%91%D0%BB%D1%83%D0%BC%D0%B0). | https://www.postgresql.org/  1996/  FreeBSD, OpenBSD, Linux, macOS, Solaris, Microsoft Windows и Unix–подобные ОС | 1.Поддер–живает ряд объектно–ориенти–рованных функций;  2. асси–хронная реплик–ция | 1. Хранимые процедуры |
| MS SQL Server | Система управления реляционными БД фирмы Microsoft | SQL/  С, С++,  С# | 1.Есть генерация скриптов для перемещения данных | http://www.microsoft.com/sqlserver/en/us/default.aspx  /1989/ UNIX, OS/2,  Microsoft Windows | 1.Простота использо–вания;  2. визуали–зация с мобильных устройств | 1Проблемы при импорте файлов;  2. занимает все ресурсы, которые имеются |
| **Название** | **Описание** | **Язык запросов/ БД** | **Отличие от других СУБД** | **Сайт/** **год первого релиза/ОС** | **Преиму–щества** | **Недостатки** |
| Oracle | Объектно–реляционная СУБД фирма Oracle | PL/SQL  /Java, С, С++,  Python | 1.Многоверсионность данных для управления параллельными транзакциями; | https://www.oracle.com/  database/  1977/  кроссплат–форменная: UNIX,  OS/2,  Microsoft Windows и т.д. | 1. Эталон надеж–ности | 1.Высокая стоимость внедрения и владения;  2. требования  к ресурсам –  высокие |

На основе рассмотрения основных функций и особенностей СУБД и ранее рассмотренных CMS (систем управления содержимым), созданных на основе СУБД, и в соответствии с задачами динамического сайта можно выбрать необходимое средство разработки. Так как количество клиент– серверных СУБД и БД невелико, то отметим, что за рамками обзора остались только следующие клиент – серверные СУБД: Firebird, Interbase, IBM DB2, Informix, Sybase Adaptive Server Enterprise, Cache, ЛИНТЕР.

**Сервера разработки и функционирования динамических веб–сайтов.**

И, наконец, рассмотрим популярные готовые сервера разработки динамических сайтов, в состав которых входят те или иные из рассмотренных нами ранее компонентов. Википедия объединяет некоторые из них в три большие категории в зависимости предполагаемой к использованию на компьютере (либо сервере) операционной системы: WAMP (для OC Microsoft Windows), LAMP (для ОС Linux), MAMP (для ОС MacOS). Системы WAMP, MAMP и LAMP представляют собой пакет прикладных программ (ППП), связывающих входящие в их состав компоненты, которые мы перечислим, без указания их назначения, в Таблице 7. ППП поставляются так, чтобы их не нужно было устанавливать и настраивать по отдельности, то есть в режиме «plug and play». Это означает, что нужно просто запустить программу развертывания сервера и следовать простым инструкциям для подготовки сервера разработки сайтов и пуска его в кратчайшие сроки и с минимальными усилиями.

Часть компонентов этой таблицы нами уже рассмотрена, функциональность остального перечисленного программного обеспечения, входящего в сервер, можно посмотреть в Википедии, которая, в основном, дает корректные описания для ППП. Отметим только, что обязательными элементами таких серверов разработки динамических веб–сайтов, является:

1. СУБД/БД или CMS;
2. средства ETL (загрузки и выгрузки файлов БД, иногда называемые по–другому, например ELT и т.п. в зависимости от используемых технологий и порядка очистки/загрузки/выгрузки данных из БД;
3. средства разработки динамического сайта;
4. среда программирования и компилятор/интерпретатор языка разработки динамического сайта;
5. средства информационной безопасности сайта и защиты его данных.

**Таблица 7 – Сервера разработки динамических веб–сайтов**

| **Название** | **Компоненты ППП стандартной поставки** | **СУБД** | **Категория/OC** |
| --- | --- | --- | --- |
| Open Server | Apache, Nginx, PHP,  SlimFTPd,  Fake Sendmail,  NNCron Lite,  Adminer, Webgrind,  PHPPgAdmin, PHPMyAdmin,  PHPMemcachedAdmin | MySQL,  PostgreSQL,  HeidiSQL,  MariaDB, Memcached,  Redis | WAMP |
| Denver | Apache,  PHP, PHPMyAdmin | MySQL, PostgreSQL, InterBase/FireBird | WAMP |
| XAMPP | Apache,  PHP, PHPMyAdmin | MySQL | WAMP, LAMP, MAMP, OC Solaris |
| EasyPHP | Apache,  PHP | MySQL,  PHPMyAdmin | WAMP, LAMP |
| VertrigoServ | Apache,  PHP,  Smarty,  Xdebug, PHPMyAdmin, SQLiteManager | MySQL,  SQLite, | WAMP |
| AppServ | Apache,  PHP,  PHPMyAdmin | MySQL | WAMP |
| WampServer | Apache,  PHP,  XDebug,  XDC,  webGrind,  PHPMyAdmin | MySQL | WAMP |
| АМPPS | Softaculous AMPPS,  Apache, PHP,  PERL, Python | MySQL,  MongoDB | WAMP, MAMP, |
| Zend Server | Apache HTTP Server, PHP,  Zend Framework, PHPMyAdmin | Oracle, DB2, MySQL, PostgreSQL, MariaDB | LAMP |

**Примеры выбора программного обеспечения и языка разработки в соответствии с назначением динамического сайта.**

1. Пусть организуется разработка автоматизированной информационной системы (АИС) для ресторана ООО «Виктория» в виде динамического веб–приложения, которое должно обеспечить возможность совершать онлайн–заказ столика, выбора меню и времени посещения заведения. Важным компонентом разрабатываемого программного средства является реляционная база данных, которую в целях экономии решено использовать бесплатную – MySQL, которая отвечает следующим требуемым характеристикам: 1) является решением для малых и средних приложений; 2) входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Denwer, XAMPP, VertrigoServ, которые планируют использовать для хранения данных и создания АИС. Кроме того, MySQL будет использоваться в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL и в автономные программы. Так же АИС будет подключена к базе данных MySQL для хранения и обработки информации связанной с заказами, гостями и персоналом заведения. Требуется хранение на хостинге 30 резервных копий, а так же почта для доменов. Таким образом, вышеуказанные сервера были выбраны благодаря тому, что в их состав входит программа разработки динамических сайтов веб–сервер Apache, язык программирования PHP, используемый для создания веб–приложений, операционная система Linux и база данных MySQL, как наиболее пригодные для задач создания системы заказов и бесплатные. Так как требовалось 40ГБ на жестком диске, SSL–сертификат от GlobalSign или Let’s Encrypt, а доступ организуется по протоколу FTP/SSH с защитой от DDoS на уровнях L3/L4, то разработка планировалась на языке программирования PHP в рамках веб–приложения, размещенного на хостинг–площадке и имеющего свой домен в сети Интернет. Хостинг планируется с хорошим оборудованием и достаточными для обеспечения бесперебойной и быстрой работы АИС характеристиками.

2. Приведем еще один пример выбора средства разработки динамического веб–сайта Интернет–магазина. Пусть требуется создать интернет–сайт магазина, который имеет большую номенклатуру товаров, например, запчастей любой автомобильной фирмы. Запчасти требуются во всем мире и их поставщики хотят видеть названия их на языке страны, в которой они продаются. В этом случае стоит задуматься над тем, чтобы сделать динамический сайт, так как перевод номенклатурных единиц носит специализированный характер. Cначала стоит понять, какой трафик потребуется для передачи запросов такого сайта на ближайшие склады, которых не так много у любого дилера, то есть определить объем предполагаемого трафика и пропускную способность сети, которая его пропустит без задержек [12, C. 95], так как передача потребуется не только артикулов, но и картинок, на которых искомые запчасти представлены в комплекте узлов автомобиля. Надежность и защищенность информации могут определить требования к БД. Пусть определено, что фирма может себе позволить СУБД ORACLE, так как при заказе дорогостоящих запчастей любая покупка имеет большую стоимость, а количество «серых» запчастей на рынке не позволит фирме получить свою прибыль в полном объеме и может привести к тому, что неоригинальная запчасть сильно уменьшит безопасность отремонтированной машины. Таким образом, появляется необходимость в хорошо защищенных каналах передачи данных и нетиповом варианте создания сайта. Посмотрев таблицу 7 нашего обзора можно остановить свой выбор на PL/SQL, так как предполагается быстрая разработка web–приложений с ограниченной функциональностью с последующей пакетной передачей данных с помощью PL/SQL блоков, т.е. небольшой программы на языке программирования PL/SQL. Из представленных средств разработки Drupal лучше всего подходит, и в соответствии с этим выбираем язык PHP и OC Microsoft Windows, как более защищенную от взлома.

**Выводы**

Таким образом, в статье приведен обзор средств разработки динамических веб–сайтов и рассмотрены примеры выбора средств для разработки динамических сайтов в соответствии с их целями и решаемыми задачами.

В статье так же обсуждается «Методика создания КХД» [4, С.50], адаптированная под веб–технологию:

1. Выбор архитектуры приложения;

2. Выбор СУБД, наиболее полно соответствующей архитектуре приложения, основанный на размерах организации, технических требованиях и финансовых ограничениях;

3. Выбор варианта обеспечения информационной безопасности АИС и БД, для чего рекомендуется перед началом разработки приложения построить логические модели БД и приложения, адаптировав их в дальнейшем под хостинг и средства разработки: подобрать вариант соответствующего выбору аппаратного обеспечения; запланировать масштабируемость; определить требуемое время отклика и пропускную способность СПД, а так же четко прописать функциональные возможности в зависимости от назначения программного обеспечения и степени его защищенности;

4. Адаптация приложения для целей и задач, решаемых в рамках веб–сайта.

**Литература**

1. Меньшикова Л.В., Найденова Д.М., Шорохов Н.В., Яковлев Д.А. Обзор программного обеспечения разработки web–сайтов.// «Информационно–технологический вестник». 2023. №1. С.89–106.
2. Меньшикова Л.В. Модели хранилищ данных информационно–аналитических систем. Часть 1// Технологии и средства связи. 2011. №3. С. 42–43.
3. Меньшикова Л.В. Модели хранилищ данных информационно–аналитических систем. Часть 2// Технологии и средства связи. 2011. №4. С. 46–48.
4. Меньшикова Л.В. Модели хранилищ данных информационно–аналитических систем. Часть 3// Технологии и средства связи. 2012. №1 С. 50–51.
5. Спирли Эрик. Корпоративное хранилище данных. Концепция Инмана.// Издательский дом «Вильямс». 2001г. – 396 с.
6. Уильям Инман. Типы хранилищ данных // Источник – http://www.iso.ru.
7. Артюшенко В.М. Сервис информационных систем в электротехнических комплексах. Монография/ Москва. 2010. – 142 с.
8. Булешный Н.М. HTTP против IP./ https://nsoft–s.com/mychatarticles/1164–http–vs–ftp.html.
9. Артюшенко В.М., Аббасова Т.С. Эффективность защиты от внешних помех электропроводных каналов структурированных кабельных систем для передачи высокоскоростных информационных приложений// Информационные технологии. 2014. С.13 – 24.
10. Меньшикова Л.В., Найденова Д.М. Мониторинг радиочастотного спектра сети спутниковой связи, охватывающей всю территорию Российской Федерации: особенности практики территориально распределенного решения.// «Информационно–технологический вестник». 2022. №4 С. 119–135.
11. https://dic.academic.ru [Электронный ресурс]. Режим доступа: https://dic.academic.ru/dic.nsf/ruwiki/1871848 (дата обращения: 07.02.2023).
12. Меньшикова Л.В., Найденова Д.М. Пропускная способность сети спутниковой связи с космическим сегментом – геостационарным искусственным спутником Земли.// «Информационно–технологический вестник». 2022. №3. С.89–106.