МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
(ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ)

*На правах рукописи*

**Леонов Андрей Владимирович**

# РАЗРАБОТКА ТЕХНОЛОГИИ АВТОМАТИЗИРОВАННОЙ ПОДГОТОВКИ ДИНАМИЧЕСКИХ ДОКУМЕНТОВ И ИНТЕРАКТИВНОГО ПОВЕСТВОВАНИЯ

Специальность 05.13.11 –
"Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей"

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель –
доктор физико-математических наук,
профессор С. В. Клименко

Москва – 2005

# Оглавление

# Введение

Работа выполнена на кафедре системной интеграции и менеджмента Московского физико-технического института (г. Долгопрудный) и в Институте физико-технической информатики (г. Протвино), который является базовой организацией этой кафедры.

Автор выражает глубокую признательность Станиславу Владимировичу Клименко за чуткое руководство и постоянную поддержку, значение которых сложно переоценить. Неоценимую помощь на заключительном этапе подготовки диссертации к защите оказал Михаил Исаевич Гуревич, которому автор искренне благодарен. Автор благодарит руководителей компании "Телеком Транспорт" Евгения Гаскевича и Олега Скрипачева за понимание и предоставленную возможность заниматься научной работой. Наконец, автор сердечно благодарит за помощь своих коллег: Бориса Бахбуха, Владимира Лудинова, Виталия Обернихина, Игоря Петренко, Рената Хуснутдинова.

## Общая характеристика работы

**Актуальность проблемы.** В последнее время в сфере систем управления документами наблюдается несколько тенденций. Программные продукты, которые изначально разрабатывались для автоматизации разных аспектов управления документами, постепенно сближаются друг с другом по функциональности, интегрируются с продуктами других производителей[1]. В повседневный оборот входят такие термины, как *виртуальный (virtual)*, *живой (live, alive), оперативно доступный по запросу (on-line), эволюционирующий*

---

[1] Приобретения: Hummingbird [120] + PC DOCS + Fulcrum (1999), Documentum [118] + eRoom (2002), Vignette [145] + Epicentric (2002), Oracle [134] + PeopleSoft [136] + J. D. Edwards [126] и т. д. Интеграция: Documentum [118] + Verity [144], SAP [138] + Documentum [118], Convera [113] + Oracle [134] + SUN [142] и т. д.

*(evolving)* и *интеллектуальный (intelligent)* документ, что находит отражение в отчетах исследовательских групп[2] и публикациях научных сообществ[3]. Наконец, все больше компаний начинают позиционировать свои продукты как *системы управления знаниями*[4]. Все эти процессы глубоко взаимосвязаны друг с другом и отражают разные стороны одного явления: в сфере управления документами происходит качественный переход, связанный с возникновением нового класса электронных документов – динамических документов [1].

До тех пор, пока основной задачей было эффективное управление электронными версиями бумажных документов, в центре внимания находились задачи *автоматизации документооборота* или *автоматизации управления документами*. Уже к середине 90-х гг. прошлого столетия развитие технологий сканирования, автоматического распознавания, регистрации и индексирования документов, устройств хранения данных, систем управления базами данных (СУБД), средств редактирования и печати документов, технологий шифрования, механизмов подтверждения подлинности и целостности документов, повсеместное распространение Интернета и электронной почты позволили в широких масштабах осуществить переход к электронному документообороту [48]. Интеграция перечисленных технологий в единые программные комплексы привела к созданию многофункциональных корпоративных систем управления документами, автоматизирующих

---

[2] Gartner [119], IDC [124], Delphi Group [114], META Group [128], Ovum [135] и др.

[3] IEEE Computer Society [76], IEEE Communication Society [77], IEEE Professional Communication Society [78], Association for Computing Machinery [81], American Society for Information Science and Technology [80], Assosiation of Knowledgework [82] и др.

[4] Например, IBM Lotus [123], Microsoft SharePoint Portal [129], Oracle Collaboration Suite [134], Sun Microsystems ONE Portal Server [142], Documentum [118], Hummingbird [120], Convera [113], Hyperwave [121], Open Text [133], Divine [116] и т. д., из отечественных - Cognitive Technologies ("Астарта") [112], Галактика ("Галактика-ZOOM") [99] и др.

различные контуры документооборота предприятий [40]. С технической точки зрения, к настоящему моменту задача перехода к электронному документообороту практически решена в таких сферах, как бухгалтерский учет, складской учет (ERP), логистика (SCM, SRM), учет контактов с клиентами (CRM), управление кадрами и других хорошо формализованных областях[5].

Переход от оборота бумажных документов к обороту их электронных аналогов позволил многократно упростить процесс подготовки и согласования документов, ускорить их доставку адресатам и прохождение документами своего жизненного цикла, усовершенствовать систему хранения и поиска документов [10] – но не добавил ничего принципиально нового в сам процесс передачи информации от одних людей к другим посредством документов. Статичный электронный документ создавался, утверждался, регистрировался, передавался на исполнение, двигался внутри организации и попадал в архив практически точно так же, как и бумажный: разница заключалась лишь в удобстве обращения с ним. Пользователь мог получить "из компьютера" только те документы, которые были когда-то "в компьютер" занесены – тексты, таблицы, изображения, аудио- или видеофайлы и т. д.

Дальнейшее развитие технологий интеллектуального поиска и анализа документов, РСУБД и технологий конструирования прототипов (шаблонов) документов, технологий автоматического реферирования и машинного перевода, технологий разбора и генерирования текстов на естественных языках позволило реализовать в системах управления электронными документами совершенно новую функциональность, которая была в принципе недоступна

---

[5] Из популярных программных продуктов можно отметить Oracle E-Business Suite [134], PeopleSoft [136], SAP [138], Siebel [141], i2 [122], J. D. Edwards [126], Sage Group [137], Scala [139], Baan [110], Microsoft Axapta [129], Microsoft Navison [129], из отечественных - 1С: Предприятие [97], Галактика [99], Парус [104], ДЕЛО [107] и др.

при работе с бумажными документами [10]. Электронные информационные системы "научились" не просто выдавать пользователю те документы, которые были когда-то занесены "в компьютер", а автоматически генерировать по запросу пользователя новые документы на основе доступной информации [1]. Такие документы получили в литературе название *динамических,* или *виртуальных* документов [73]. В современных информационных системах управление отдельными этапами *жизненного цикла (document workflow)* динамических документов все чаще осуществляется автоматически, что делает их полноправными участниками бизнес-процессов предприятия [45], [70], [72].

Все это позволяет говорить о качественном переходе к информационным системам следующего поколения, ключевым элементом которых являются динамические документы – *эволюционирующие*, *интеллектуальные*, *живые*. Если раньше основные усилия разработчиков были сконцентрированы на автоматизации ввода бумажных документов в систему, развитии технологий хранения и поиска документов в базах данных и совершенствовании средств коллективной работы с документами, то сейчас акцент сместился на развитие технологий интеллектуальной обработки и анализа информации, совершенствование средств поиска требуемых сведений и их представления в удобной для пользователя форме. Все современные системы управления документами в той или иной степени "умеют" автоматически генерировать новые документы на основе доступной информации [33], [71].

Анализ последних тенденций в сфере электронного документооборота показывает, что разработка методов и технологий работы с динамическими документами становится магистральным направлением развития современных информационных систем [1], [32], [39], [73], [150]. Практически все компании, предлагающие системы управления документами, так или иначе работают над этой задачей, что в ближайшем будущем приведет к глубокой перестройке существующих бизнес-процессов и схем работы с информацией.

**Целью диссертационной работы** является разработка и развитие методов и технологий автоматизированной подготовки динамических документов, в том числе динамических документов нового типа – интерактивного повествования в виртуальном окружении.

В рамках данной работы поставлены и решены следующие задачи:

- Исследование нового класса электронных документов – динамических документов. Описание характеристик динамических документов. Анализ возможностей, которые дает использование динамических документов при построении электронных информационных систем.
- Разработка технологии автоматизированной подготовки динамических документов. Построение на основе этой технологии системы автоматизированной подготовки и публикации документов на корпоративном сайте и ее внедрение в эксплуатацию.
- Разработка технологии записи массива XML-документов в таблицы РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов. Построение на основе этой технологии экспериментальной системы автоматического генерирования DTD.
- Исследование нового типа динамических документов – интерактивного повествования в виртуальном окружении. Описание методов и технологий интерактивного повествования в виртуальном окружении. Анализ его возможных применений для создания электронных информационных, обучающих и тренировочных систем.
- Разработка технологии интерактивного повествования в виртуальном окружении. Интеграция технологий динамических документов и виртуального окружения на технологической платформе Avango. Построение обучающей системы "Виртуальный Планетарий" на основе технологии интерактивного повествования в виртуальном окружении.

**Научная новизна результатов.** Понятие динамического документа появилось в зарубежной литературе около 10 лет назад [73]. Однако, до сих пор не был проведен содержательный анализ этого понятия и связанного с ним комплекса методов и технологий. В данной работе концепция динамических документов впервые представлена в целостном, логически связанном виде. Описана история развития электронных документов, рассмотрены возможности электронных документов, показан механизм возникновения нового класса электронных документов – динамических документов, описаны характеристики динамических документов, исследованы их возможности и преимущества.

Представленная технология автоматизированной подготовки динамических документов по своей архитектуре близка к технологии построения так называемых *динамических* сайтов. Однако существующие технологии построения динамических сайтов разработаны и описаны, как правило, либо с точки зрения программиста, либо с точки зрения дизайнера (верстальщика). В первом случае объектом исследования являются программные продукты и языки программирования, а целью – создание на их основе новых программных модулей, интеграция различных программных продуктов друг с другом, разработка новых алгоритмов и приемов программирования. Во втором случае объект исследования – это языки разметки (HTML и др.), а цель – наиболее эффективное отображение информации на экране монитора с учетом характеристик компьютеров и программного обеспечения пользователей.

Технология, представленная в данной работе, разработана и описана с точки зрения разработчика (конструктора) динамических документов, цель которого – наиболее эффективная организация информационного взаимодействия между электронной информационной системой и ее пользователями. Объектом исследования являются динамические документы – новый класс электронных документов, которые предоставляют намного более широкие возможности управления информацией, чем традиционные статичные

электронные документы. Результатом исследования является новая технология работы с информацией, основанная на использовании динамических документов. Эта технология может применяться для построения электронных информационных систем самых разных типов – корпоративных сайтов, баз знаний, экспертных систем и т. д. В частности, на ее основе разработана технология интерактивного повествования в виртуальном окружении, описанная в данной работе.

Представленная технология записи массива XML-документов в РСУБД без использования информации об их структуре и генерирования DTD для этого массива XML-документов является новой. В литературе описан ряд алгоритмов записи отдельного XML-документа в РСУБД без использования информации о его структуре [53], [54], [55]. Также в литературе описан алгоритм построения DTD для отдельного XML-элемента [57]. В данной работе задача генерирования DTD для массива XML-документов впервые рассмотрена как часть более общей задачи автоматического занесения структурированной информации в РСУБД электронной информационной системы. Разработанная технология записи массива XML-документов в РСУБД и генерирования DTD для этого массива XML-документов позволяет автоматизировать наполнение РСУБД информацией и тем самым существенно повысить эффективность автоматизированной подготовки динамических документов.

Технология интерактивного повествования в виртуальном окружении, представленная в данной работе, является новой. В мире есть несколько десятков коллективов, которые занимаются разработкой методов и технологий интерактивного повествования в виртуальном окружении [90], [92]. Однако, как и в любой новой предметной области, понятие *интерактивного повествования* по-разному трактуется разными исследователями. Этот факт в сочетании с широким спектром систем и технологий виртуального окружения приводит к тому, что каждый коллектив фактически разрабатывает свою технологию интерактивного повествования в виртуальном окружении, которая

существенно отличается от других разработок. Представленная технология интерактивного повествования в виртуальном окружении основана на интеграции технологий динамических документов и виртуального окружения на технологической платформе Avango [23]. Это новый подход, который ранее не рассматривался и не был описан другими исследователями.

**Научная и практическая ценность результатов.** Технология автоматизированной подготовки динамических документов, представленная в данной работе, может использоваться для построения электронных информационных систем разной функциональности и масштаба. Она представляет интерес для разработчиков современных электронных информационных систем, которых не удовлетворяет функциональность статичных электронных документов и которые стремятся расширить возможности работы с информацией. Эта технология может использоваться для построения корпоративных информационных систем, баз знаний, систем управления знаниями, корпоративных сайтов, обучающих программ, экспертных систем, публичных информационных порталов и т. д.

Технология записи массива XML-документов в РСУБД без использования информации о их структуре и генерирования DTD для этого массива XML-документов, описанная в данной работе, представляет интерес для разработчиков электронных информационных систем, которым необходимо автоматизировать наполнение РСУБД структурированной информацией. Эта задача неизбежно возникает при развитии любой электронной информационной системы, когда ручное занесение информации в систему становится неэффективным и перестает удовлетворять возросшим требованиям к объему и качеству структурирования информации. Представленная технология генерирования DTD для массива XML-документов в комплексе с системами автоматического поиска информации и конвертерами информации из документов и баз данных в формат XML обеспечивает эффективное решение задачи автоматического наполнения РСУБД структурированной информацией.

Структурированная информация из таблиц РСУБД может быть легко использована для автоматизированного построения динамических документов.

Технология интерактивного повествования в виртуальном окружении, описанная в данной работе, представляет интерес для разработчиков электронных информационных, обучающих и тренировочных систем. Эта технология основана на технологической платформе Avango, которая имеет открытый исходный код и распространяется свободно [23]. Стоимость системы виртуального окружения на Linux-кластере персональных компьютеров сегодня вполне доступна для крупных отечественных научных центров, ВУЗов, промышленных и добывающих корпораций [93]. Учитывая, что стоимость разработки приложений виртуального окружения на базе программного обеспечения с открытым исходным кодом на порядок меньше, чем стоимость фирменных систем с аналогичной функциональностью, можно предположить, что круг потенциальных пользователей предложенной технологии интерактивного повествования в виртуальном окружении достаточно широк. Среди возможных применений данной технологии – создание инструкций по эксплуатации и документации к технологически сложным изделиям, в том числе, "двойного" назначения, в рамках концепций CALS, PLCS, PLM [102].

**Достоверность и обоснованность** полученных результатов подтверждается публикациями результатов в ведущих научных журналах и трудах международных конференций, в которых проводится тщательное рецензирование.

**Личный вклад автора.** Автору принадлежит инициатива в постановке и решении основных задач диссертации. Личный вклад автора состоит в разработке целостной научной концепции динамических документов [1], разработке новой технологии автоматизированной подготовки динамических документов [2], исследовании задачи построения системы автоматизированной подготовки динамических документов с использованием программного обеспечения с открытым исходным кодом [3], развитии и конструктивной

проработке методов и алгоритмов записи XML-документов в РСУБД без использования информации об их структуре [4], развитии и конструктивной проработке методов и алгоритмов генерирования DTD для массива XML-документов [5], разработке новой технологии интерактивного повествования в виртуальном окружении [6].

**Апробация результатов.** Технология автоматизированной подготовки динамических документов, представленная в данной работе, была использована для создания системы автоматизированной подготовки и публикации документов на корпоративном сайте. Эта система была внедрена в эксплуатацию в компании "Телеком Транспорт" в 2000-2002 гг. и успешно функционирует в настоящее время.

Технология записи массива XML-документов в РСУБД без использования информации об их структуре и генерирования DTD для этого массива XML-документов, представленная в данной работе, была реализована в виде экспериментальной системы, которая может использоваться как для решения практических задач, так и для дальнейших исследований и разработок.

Технология интерактивного повествования в виртуальном окружении, представленная в данной работе, была использована для построения экспериментальной обучающей системы "Виртуальный Планетарий". Разработка и развитие этой системы продолжается в настоящее время в Институте физико-технической информатики.

**Публикации.** По материалам диссертации опубликовано 6 работ [1-6].

**Структура и объем диссертации.** Диссертация состоит из введения, четырех глав, заключения, приложений, глоссария и списка литературы. Полный объем диссертации: 125 страниц основного текста (9 таблиц, 10 иллюстраций) и 13 страниц приложений. Список литературы, использованной при работе над диссертацией, содержит 181 наименование.

## Структура диссертации

**Во введении** дана общая характеристика работы, описана структура диссертации и перечислены результаты, выносимые на защиту.

**В главе 1** вводится объект исследования: широкий класс электронных документов, получивший в литературе название динамических (виртуальных) документов. *В разделе 1.1* определены основные термины и понятия: "информация", "документ", "электронный документ" и "динамический документ". *В разделе 1.2* дан краткий обзор истории развития электронных документов и описаны основные технологии, развитие которых позволило осуществить масштабный переход от оборота "бумажных" документов к обороту их электронных аналогов. *В разделе 1.3* описаны новые возможности электронных документов, которые были в принципе недоступны при работе с "бумажными" документами: гиперссылки, мультимедийность (использование аудио- и видео-компонентов), использование прототипов (шаблонов), автоматический поиск, анализ и обработка документов. *В разделе 1.4* показано, что развитие технологий привело к появлению нового типа электронных документов – динамических документов, которые являются ключевым элементом современных электронных информационных систем. Наконец, *в разделе 1.5* описаны перспективные направления развития методов и технологий работы с динамическими документами.

**В главе 2** представлена технология автоматизированной подготовки динамических документов, основанная на хранении структурированной информации в таблицах РСУБД и использовании прототипов. Эта технология описана на примере системы автоматизированной подготовки и публикации документов на корпоративном сайте, которая была разработана автором в сотрудничестве с коллегами [2] и внедрена в компании "Телеком Транспорт" [106] в 2000-2002 гг. *В разделе 2.1* обоснована необходимость автоматизации подготовки и публикации документов. В *разделе 2.2* сформулированы требования, которым должна удовлетворять система автоматизированной

подготовки и публикации документов. *В разделе 2.3* описана технология хранения структурированной информации в таблицах РСУБД, технология конструирования прототипов и схема работы интерпретатора. *В разделе 2.4* представлена методология подготовки основных типов документов и описана архитектура системы автоматизированной подготовки и публикации документов. Наконец, *в разделе 2.5* описаны критерии выбора программного обеспечения для построения информационной системы, основанной на использовании динамических документов.

**В главе 3** представлена технология записи массива XML-документов без использования информации об их структуре в РСУБД, и автоматического генерирования DTD для этого массива XML-документов. Эта технология является важным элементом системы автоматизированной подготовки динамических документов, так как позволяет автоматизировать создание таблиц РСУБД и их заполнение структурированной информацией. Технология записи XML-документов в РСУБД и генерирования DTD была разработана автором в сотрудничестве с Р. Р. Хуснутдиновым [4], [5] и реализована в виде экспериментальной системы в 2003-2004 гг. *В разделе 3.1* показано, что технология записи массива XML-документов в РСУБД и генерирования DTD для них является одним из ключевых элементов системы автоматизированной подготовки динамических документов. *В разделе 3.2* описаны известные методы записи XML-документов в РСУБД без использования информации об их структуре (такой, как DTD, XML Schema и т. п.). *В разделе 3.3* избранные методы развиты и модернизированы для решения задачи записи массива XML-документов в РСУБД за один проход. *В разделе 3.4* представлена технология генерирования DTD для массива XML-документов, развитая на основе методов и алгоритмов, предложенных в [57]. Наконец, *в разделе 3.5* описана архитектура системы записи массива XML-документов в РСУБД и генерирования DTD для них.

**Глава 4** посвящена новому типу динамических документов – интерактивному повествованию в виртуальном окружении. Это перспективное направление развития компьютерных технологий, которое находится на стыке электронных информационных систем, компьютерных игр, обучающих программ, виртуальных тренажеров и интерактивных моделей. Технология интерактивного повествования в виртуальном окружении описана на примере обучающей системы "Виртуальный Планетарий" [6], которую автор в сотрудничестве с коллегами разрабатывает в настоящее время. *В разделе 4.1* представлен новый тип динамических документов – интерактивное повествование в виртуальном окружении. *В разделе 4.2* рассмотрены основные методы и технологии интерактивного повествования. *В разделе 4.3* дан обзор основных технологий виртуального окружения. *В разделе 4.4* рассмотрена технологическая платформа Avango – программное обеспечение для разработки интерактивных приложений в виртуальном окружении. Наконец, *в разделе 4.5* описана архитектура и принципы интерактивного повествования в виртуальном окружении на примере обучающей системы "Виртуальный Планетарий".

**В заключении** приведены основные результаты работы.

## Результаты, выносимые на защиту

- Исследован новый класс электронных документов – динамические документы. Динамический документ – это документ, автоматически создаваемый системой по запросу пользователя на основе доступной информации. Динамические документы обладают намного более широкой функциональностью, чем традиционные статичные электронные документы. Динамические документы широко применяются при построении современных электронных информационных систем.
- Разработана технология автоматизированной подготовки динамических документов. Эта технология основана на хранении структурированной информации в таблицах РСУБД и использовании прототипов. С

использованием представленной технологии автоматизированной подготовки динамических документов построена система автоматизированной подготовки и публикации документов на корпоративном сайте. Эта система была внедрена в эксплуатацию в компании "Телеком Транспорт" в 2000-2002 гг. и успешно функционирует в настоящее время.

- Разработана технология записи массива XML-документов в таблицы РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов. Эта технология позволяет автоматизировать занесение структурированной информации в таблицы РСУБД и тем самым существенно повысить эффективность автоматизированной подготовки динамических документов. Технология записи массива XML-документов в таблицы РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов была разработана и реализована в 2003-2004 гг. в виде экспериментальной системы.
- Исследован новый тип динамических документов – интерактивное повествование в виртуальном окружении. Интерактивное повествование в виртуальном окружении – это новый жанр компьютерных приложений, который находится на стыке электронных информационных систем, компьютерных игр, обучающих программ, виртуальных тренажеров и интерактивных моделей.
- Разработана технология интерактивного повествования в виртуальном окружении, основанная на интеграции технологий динамических документов и виртуального окружения на технологической платформе Avango. На основе представленной технологии интерактивного повествования в виртуальном окружении разработана экспериментальная обучающая система "Виртуальный Планетарий".

# 1. Объект исследования: динамический документ

В данной главе вводится объект исследования: широкий класс электронных документов, получивший в литературе название динамических (виртуальных) документов. *В разделе 1.1* определены основные термины и понятия: "информация", "документ", "электронный документ" и "динамический документ". *В разделе 1.2* дан краткий обзор истории развития электронных документов и описаны основные технологии, развитие которых позволило осуществить масштабный переход от оборота "бумажных" документов к обороту их электронных аналогов. *В разделе 1.3* описаны новые возможности электронных документов, которые были в принципе недоступны при работе с "бумажными" документами: гиперссылки, мультимедийность (использование аудио- и видео-компонентов), использование прототипов (шаблонов), автоматический поиск, анализ и обработка документов. *В разделе 1.4* показано, что развитие технологий привело к появлению нового типа электронных документов – динамических документов, которые являются ключевым элементом современных электронных информационных систем. Наконец, *в разделе 1.5* описаны перспективные направления развития методов и технологий работы с динамическими документами.

## 1.1. Определение терминов и понятий

Прежде чем говорить о динамических документах и интерактивном повествовании в виртуальном окружении, автор считает необходимым высказать свое представление об основных понятиях – таких, как *информация*, *документ*, *электронный документ* и *динамический документ*, но не в виде строгих и универсальных математических определений, а в прикладном смысле, повествовательно и субъективно, в соответствии с тем, как они будут употребляться в настоящей работе.

Как известно, понятие информации является одним из основных понятий, уточнение содержания которого не может быть достигнуто с помощью определения, так как последнее лишь сводило бы это понятие к другим не определенным основным понятиям [8]. Информация субъективна (зависит от подготовленности субъекта воспринимать информацию) и может содержаться в самых разнообразных сведениях, сообщениях, знаниях и умениях.

В прикладном смысле **информация** – это набор сведений (сигналов, символов), которые уменьшают степень неопределенности у их получателя [1]. Например, по законодательству РФ: "Информация – сведения о лицах, предметах, фактах, событиях, явлениях и процессах независимо от формы их представления" [181].

Классическая теория информации не определяет, что такое информация, а просто предполагает, что для источника с заданным распределением вероятностей состояний мерой информации является энтропия $H$ [17]. Для дискретного источника, состояния которого $s_1 \ldots s_q$ имеют вероятности $p_1 \ldots p_q$, энтропия H определяется как: $H = -\sum_{i=1}^{q} p_i \log p_i$ . Для источника с непрерывным распределением вероятности состояний с функцией плотности распределения $p\ (x)$ энтропия H определяется как: $H = -\int_{-\infty}^{\infty} p(x) \log p(x) dx$. Выбор основания логарифма соответствует выбору единицы измерения информации. Единицы измерения, получающиеся при использовании основания два, называются битами. Например, каждое состояние источника с двумя равновероятными состояниями несет один бит информации ($H = 1$).

При любых видах работы с информацией всегда идет речь о ее представлении в виде определенных символических структур (символов, знаков). Информация, представленная в символическом виде и предназначенная для передачи от отправителя к получателю, называется *сообщением*. Для технических целей термин "информация" используется в значении "содержание сообщения". Сообщение $N$ и содержащаяся в нем

информация $I$ связаны друг с другом некоторым отображением – правилом интерпретации $P$, которое представляет собой результат договоренности между отправителем и получателем: $I \xleftrightarrow{P} N$.

Передача информации между людьми всегда происходит посредством сообщений. Автор формирует сообщение (представляет информацию в определенной символической форме); сообщение передается адресату; адресат восстанавливает информацию из сообщения. Для формирования сообщения используются символы, значение которых заранее согласовано между автором и адресатом. В тех случаях, когда для передачи информации используются общепринятые символы или системы символов (как языковые – письменность, речь, формулы и др., так и неязыковые – жесты, интонация, мимика и др.), такое согласование осуществляется неявным образом. Отметим, что сообщение, сформированное автором, может не нести для адресата никакой информации. Эта ситуация возникает, когда правило интерпретации (значение символов) не согласовано заранее между автором и адресатом, или когда сообщение не содержит ничего нового для адресата.

Понятие документа, как и понятие информации, не имеет общепринятого определения. В информатике и технике связи понятие документа отсутствует – с технической точки зрения, это лишь один из возможных типов сообщений, которые используют люди для передачи информации.

В прикладном смысле **документ** – это сообщение, зафиксированное на материальном носителе (в отличие от сигнала – физического процесса, распространяющегося в пространстве и времени, параметры которого содержат сообщение) [10]. Например, по законодательству РФ: "Документ – зафиксированная на материальном носителе информация с реквизитами, позволяющими ее идентифицировать" [181].

Книга, газетная статья, мемориальная доска или рекламный плакат – это документ. Песня на аудиокассете, видеофильм на пленке или компакт-диске, фотография или наскальный рисунок, берестяная грамота или глиняная

табличка, монета или икона – это документ. Скульптура или барельеф, картина маслом или роспись на вазе – это документ. Формат и материал носителя не имеют значения – важен лишь факт фиксации сообщения (информации, выраженной в определенной символической форме) на материальном носителе.

Подробное исследование вопроса о том, что следует считать фиксацией сообщения на материальном носителе, выходит за рамки данной работы. Приведем лишь один из возможных критериев: сообщение можно считать зафиксированным на материальном носителе, если время стабильности параметров носителя, которые содержат сообщение, много больше характерного времени воспроизведения сообщения (представления его в форме, доступной для непосредственного восприятия органами чувств человека).

Понятие электронного документа не имеет общепринятого определения. В прикладном смысле **электронный документ** – это сообщение, зафиксированное на машинном носителе (магнитном диске, магнитной ленте, лазерном диске и др.) с помощью электронных технических средств. Следует различать *аналоговые электронные документы* (например, аудиозапись на магнитной ленте), когда изменение параметров сигнала, содержащего сообщение (в данном примере – колебания плотности воздуха) преобразуется в пропорциональное ему изменение параметров носителя (в данном примере – напряженность магнитного поля), и *цифровые электронные документы*, когда сообщение кодируется в виде последовательности "0" и "1" и эта цифровая последовательность фиксируется на носителе. В настоящее время цифровые электронные документы имеют намного большее значение, чем аналоговые электронные документы. Далее под электронным документом всегда будет пониматься цифровой электронный документ, если не оговорено обратное.

В современных электронных информационных системах основной объем информации хранится не в форме отдельных электронных документов, а в форме структурированных массивов информации (баз данных и т. п.). Один из

механизмов взаимодействия пользователя с такой системой заключается в использовании динамических (виртуальных) документов [73].

**Динамический документ** – это документ, который автоматически формируется системой по запросу пользователя с использованием доступной информации. Набор правил, по которым обрабатывается доступная информация в зависимости от запроса пользователя, определяется *прототипом* динамического документа. Динамические документы, которые выдаются на запросы разных пользователей и даже на повторные запросы одного пользователя, могут иметь существенные отличия. Именно поэтому динамический документ часто называют *виртуальным* – тем самым подчеркивается, что динамический документ не существует в электронной информационной системе в своем законченном виде, а возникает лишь как ответ системы на запрос пользователя. Определение "динамический" отражает не какие-то особые качества или свойства электронного документа, а способ его формирования. Динамический документ – это результат применения набора правил, заданного прототипом, к массиву доступной информации, с параметрами, определяемыми запросом пользователя [1].

Введем следующие обозначения: $\lambda$ – набор параметров, определяемый запросом пользователя; F – набор прототипов f; П – массив доступной информации; $f_\lambda$ – прототип из F, выбранный и преобразованный согласно $\lambda$; $\pi$ – информация из П, выбранная согласно $f_\lambda$; D – динамический документ. Тогда схема формирования динамического документа D по запросу пользователя может быть представлена следующим образом, рис. 1.1:

$$\begin{tikzcd}
\lambda \arrow[r] & F \arrow[dl, "f_\lambda"'] \arrow[dr, "f_\lambda"] & \\
\Pi \arrow[rr, "\pi"] & & f_\lambda(\pi) = D
\end{tikzcd}$$

*Рис. 1.1. Схема формирования динамического документа D.*

## 1.2. История развития электронных документов

Перевод документооборота в электронную форму в ведущих странах мира начался несколько десятков лет назад [10]. Для того, чтобы стал возможен масштабный переход от оборота бумажных документов к обороту их электронных аналогов, потребовалось развитие целого ряда технологий.

**Сканирование документов.** В любой крупной организации за время ее существования накапливается архив бумажных документов, который может включать сотни тысяч наименований. Поток приходящих в организацию или создаваемых бумажных документов также может составлять сотни наименований в день. Очевидно, что переход к безбумажному документообороту был бы невозможен без создания технологий сканирования, позволяющих быстро получать изображение документа в электронном виде (прежде всего, для его последующего распознавания). Производительность современных промышленных сканеров можно считать достаточной для решения любых практических задач сканирования документов [117].

**Автоматическое распознавание, регистрация и индексирование документов.** Использовать электронное изображение документа уже намного удобнее, чем бумажную версию – например, его можно практически мгновенно переслать по электронной почте любому адресату, подключенному к сети Интернет. Однако для того, чтобы получить электронный аналог бумажного документа, который можно свободно редактировать в электронном виде, необходимо автоматически "распознать" электронное изображение, полученное в результате сканирования. Для неструктурированного документа "распознать" означает извлечь текст с сохранением разметки и оформления в отдельный файл стандартного текстового формата либо в отдельное поле базы данных. Для структурированного документа (например, бухгалтерской формы) понятие "распознать" также включает автоматическую идентификацию типа документа (сопоставление с одним из заданных типов) и его структуры (сопоставление полей отсканированного изображения с соответствующими полями таблиц

базы данных). После того, как документ распознан, система может автоматически зарегистрировать его в электронном архиве (занести реквизиты документа в электронную регистрационную карточку документа) и выполнить полнотекстовую индексацию (построить список ключевых слов документа).

Задача распознавания текстов в настоящее время практически решена; успешно решается задача распознавания структурированных документов [108], [112], [125], [140]. Хуже обстоят дела с распознаванием реквизитов неструктурированных документов – автоматически решить, где у такого документа заголовок, где подпись автора, а где – дата выпуска, существующие системы могут только в достаточно простых случаях. Интенсивно развиваются технологии индексации текстов, однако и здесь есть свои трудности: индексация имен собственных, дат, денежных сумм, а также понятий, состоящих из нескольких слов.

**Хранение документов.** Масштабный переход к электронному документообороту был бы невозможен без развития технологий хранения данных, которые в настоящее время позволяют хранить огромные объемы информации на компактных и надежных носителях. Именно колоссальное увеличение емкости и снижение цен на носители информации за последние два десятка лет, а также развитие алгоритмов и технологий сжатия данных позволили к настоящему моменту практически свободно работать с оцифрованным звуком и видео, включая их в традиционные текстовые документы, что в итоге привело к повсеместному признанию мультимедийных (многокомпонентных, составных) документов. Создание электронных архивов документов также вряд ли было бы возможно без интенсивного развития СУБД. Несмотря на то, что традиционные реляционные и объектно-ориентированные базы данных изначально не проектировались для хранения именно документов, их функциональность и гибкость в настоящее время

достаточны для организации электронных архивов документов любого размера[6].

**Редактирование документов.** Переход к безбумажным технологиям был бы невозможен без разработки программного обеспечения, позволяющего создавать и редактировать электронные документы любому человеку со средним уровнем компьютерной грамотности. Прогресс в этой сфере затронул практически каждого пользователя, базовые функции программных продуктов типа MS Word доступны 5-летнему ребенку (что, несомненно, является выдающимся достижением на пути к электронному документообороту). Технологии управления версиями документов [14], в той или иной степени реализованные практически в любом современном программном обеспечении для редактирования документов, позволяют одновременно работать над одним документов нескольким пользователям, вносить и согласовывать исправления, формировать финальный документ на основе последних версий его компонентов, созданных различными пользователями и т. д.

Однако, в этой области есть свои проблемы, прежде всего – проблемы с совместимостью документов разных форматов, которые возникают из-за фактического отсутствия единого открытого стандарта электронных документов и "закрытости" популярных стандартов, таких как MS Word DOC [129] и Adobe PDF [109]. Развитие открытых стандартов, таких как XML [171], HTML [152], TeX [170], безусловно, приближает нас к решению этой проблемы. Однако это существенно разные форматы, предназначенные для решения разных задач (XML – разметка структуры документа для обмена данными между различными приложениями, HTML – разметка для

---

[6] Из популярных СУБД можно отметить IBM DB2 [123], Microsoft SQL Server [129], Oracle [134], Borland Software Corporation InterBase [111], Sybase SQL [143], MySQL [158], PostgreSQL [165], технологию Sun Microsystems JDBC [142] и др.

представления документа в браузере пользователя, TeX – разметка для профессиональной подготовки документа к печати [11]), и тенденций к их объединению пока не наблюдается. Кроме того, в большинстве организаций документы в открытых форматах до сих пор составляют лишь малую часть от всех электронных документов.

**Подтверждение подлинности и целостности документов.** Для того, чтобы документ имел юридическую силу, необходимо, чтобы существовал механизм подтверждения его подлинности (авторства). Бумажным документам юридическую силу придают личная подпись (виза), печать, фирменный бланк с водяными знаками и т. д. Без разработки аналогичных механизмов для электронных документов их сфера применения была бы существенно ограничена; в частности, невозможно было бы перевести в электронную форму бухгалтерию или официальную переписку. Достижения в сфере криптографии, прежде всего развитие технологий шифрования с открытым ключом, позволили создать надежный механизм подтверждения подлинности (авторства) электронных документов – электронную цифровую подпись (ЭЦП) [7].

Принцип визирования документа с использованием ЭЦП заключается в следующем. Пользователь получает в центре сертификации два *ключа* - открытый и закрытый (две последовательности символов, взаимно однозначно соответствующие друг другу). Закрытый ключ, записанный на дискете, хранится у самого пользователя со всеми необходимыми мерами предосторожности (например, в сейфе, подобно печати); открытый ключ может быть доступен большому числу людей. Когда пользователю необходимо завизировать электронный документ, он запускает специальную программу, которая на основе содержания документа и закрытого ключа вычисляет ЭЦП документа (последовательность байтов) и дописывает ее в тот же файл. Электронный документ, завизированный ЭЦП, может быть свободно прочитан любым лицом (если только он дополнительно не зашифрован), однако в него невозможно внести никакие изменения без нарушения подлинности ЭЦП.

Получатель документа с ЭЦП может легко проверить его подлинность и целостность с использованием аналогичной программы и открытого ключа.

Широкому использованию ЭЦП мешают проблемы не технологического, а юридического и психологического характера. Федеральный закон РФ "Об ЭЦП" был принят лишь 10.01.2002 г., и проработка правовых вопросов в этой сфере остается достаточно слабой. Психологически процесс визирования электронного документа с использованием ЭЦП существенно отличается от собственноручной постановки подписи на бумажном документе, в частности, далеко не так очевиден принимаемый при этом объем обязательств. Решать споры типа "моя бабушка случайно нажала какую-то кнопку, а теперь вы отберете у нее дом?" еще предстоит научиться.

**Защита документов.** В традиционном бумажном делопроизводстве для передачи секретных документов используется фельдегерская служба, курьеры и т. п. В электронном делопроизводстве секретные сведения приходится передавать, как правило, по незащищенным каналам связи, и защита документов осуществляется с использованием технологий шифрования с симметричным либо с открытым ключом [16].

При шифровании с симметричным ключом для кодирования и раскодирования документа используется одна и та же секретная последовательность байтов (которая должна быть известна и отправителю, и получателю). Шифрование с открытым ключом принципиально ничем не отличается от описанной выше технологии ЭЦП, но только для шифровки сообщения отправитель использует открытый ключ получателя, а получатель расшифровывает полученное сообщение с использованием своего закрытого ключа (таким образом, каждый, кто знает открытый ключ получателя, может отправить ему зашифрованный документ, который никто, кроме получателя, прочитать не сможет). Шифрование с симметричным ключом на несколько порядков быстрее, чем шифрование с открытым ключом, что может быть существенно при больших объемах документов, но при этом возникает

отдельная задача – как передать получателю документа секретный ключ. В настоящее время часто поступают следующим образом: само сообщение шифруется симметричным ключом, а этот секретный ключ шифруется с использованием открытого ключа.

Задача придания документу юридической силы часто совмещена с задачей защиты содержащейся в документе информации от посторонних лиц. Теоретически, для шифрования и для визирования документа может использоваться одна пара ключей: отправитель шифрует сообщение открытым ключом получателя и визирует своим закрытым ключом, получатель проверяет подлинность и целостность документа с помощью открытого ключа отправителя и расшифровывает документ с помощью своего закрытого ключа. Так и поступали до тех пор, пока не выяснилось, что такая схема позволяет злоумышленнику с использованием специальных приемов достаточно просто вычислить закрытые ключи. В настоящее время для шифрования и визирования документов ЭЦП, как правило, применяют две разные пары ключей [68].

При соблюдении требуемых мер предосторожности (конфиденциальность закрытых ключей) никакие существующие методы и средства не позволяют взломать документы, зашифрованные с использованием современных технологий и завизированные ЭЦП.

**Интернет и электронная почта.** Электронный документооборот был бы в принципе невозможен без широкого распространения быстрых и надежных каналов электронной связи. Масштабное развертывание сети Интернет, активное использование технологий Интернет для построения корпоративных сетей (Интранет) и повсеместное внедрение систем типа электронной почты [30], [69], [74] стали важными факторами для перехода от оборота бумажных документов к электронному документообороту.

Современные системы передачи электронных сообщений позволяют не только передать электронный документ от отправителя к получателю в любой узел всемирной сети Интернет, но и организовать рассылку документов по

списку получателей, назначить документу маршрут движения и организовать контроль за его продвижением, обеспечить автоматический анализ обновлений определенных документов и пересылку обновленных версий (или уведомлений) на заданный адрес и т. д. Технологические проблемы в этой сфере связаны, пожалуй, лишь с увеличением пропускной способности магистральных линий связи и развитием технологий доступа в магистральные сети (*последняя миля*), что необходимо для свободной пересылки аудио- и видео-файлов. Задача передачи текстовых сообщений между пользователями сети Интернет в настоящее время может считаться полностью решенной[7].

**Печать документов.** Строго говоря, возможность печати электронных документов нельзя назвать необходимой составляющей электронного документооборота. Однако на практике процесс перехода от бумажных документов к электронным сопряжен с преодолением многих трудностей юридического и психологического характера, и без развития технологий вывода документов на печать и разработки доступных по цене принтеров для массового использования этот переход вряд ли мог бы состояться за столь короткое время. Сегодня принтер есть не только практически в каждой организации, но и у многих домашних пользователей. Качество печати у современных струйных принтеров вплотную приблизилось к качеству традиционной офсетной полиграфии, чего вполне достаточно для решения любых практических задач документооборота.

**К середине 90-х гг.** прошлого столетия развитие технологий сканирования, автоматического распознавания, регистрации и индексирования документов, устройств хранения данных, реляционных и объектно-ориентированных СУБД, средств редактирования и печати документов,

---

[7] Из популярных систем можно отметить IBM Lotus Notes/Domino [123], Microsoft Exchange Server [129], Novell GroupWise [132], Oracle Collaboration Suite [134] и др.

технологий шифрования, механизмов подтверждения подлинности и целостности документов, повсеместное распространение Интернета и электронной почты позволило в широких масштабах осуществить переход к электронному документообороту [48]. Интеграция перечисленных технологий в единые программные комплексы привела к созданию многофункциональных корпоративных систем управления документами, автоматизирующих различные контуры документооборота [40].

С технической точки зрения, к настоящему моменту задача перехода к электронному документообороту практически решена в таких сферах деятельности, как бухгалтерский учет, складской учет (ERP), логистика (SCM, SRM), учет контактов с клиентами (CRM), управление кадрами и других хорошо формализованных областях. Интенсивно развиваются корпоративные системы для управления неструктурированными документами, в том числе содержащими аудио- и видео компоненты, обеспечивающие работу с полным электронным архивом документов предприятия[8]. Ряд ведущих поставщиков стремится позиционировать свои системы как средства для полной автоматизации всех контуров документооборота предприятия, однако на практике обычно используются несколько систем разных производителей, дополняющих друг друга (например, SAP [138] и Documentum [118]).

---

[8] Из популярных систем можно отметить IBM Lotus [123], Microsoft SharePoint Portal [129], Microsoft Exchange Server [129], Oracle Collaboration Suite [134], Documentum [118], Hummingbird [120], Convera [113], Open Text [133], Vignette [145], Divine [116], из отечественных - "Евфрат-Документооборот" (Cognitive Technologies) [112], ОПТіМА-WorkFlow (Оптима) [103], LanDocs (ЛАНИТ) [100], DocsVision (Digital Design) [115] и др.

## 1.3. Новые возможности электронных документов

Развитие технологий позволило не только автоматизировать все традиционные этапы работы с документами (подготовку, редактирование, согласование, утверждение, регистрацию, доставку адресатам, прохождение документами своего жизненного цикла, контроль исполнения, хранение и поиск в архиве), но и реализовать в системах управления электронными документами совершенно новую функциональность, которая была в принципе недоступна при работе с бумажными документами [1], [10].

**Гиперссылки, аудио- и видео-компоненты.** Принципиальным отличием электронных документов от бумажных является возможность установления связей между документами с использованием гиперссылок, а также вставки аудио- и видео-компонентов. Хотя понятие *гипертекста* появилось еще в середине прошлого столетия, широкое использование гиперссылок в электронных документах началось с возникновением сети WWW в Европейском центре ядерных исследований (CERN) в 1991 году и разработки языка разметки HTML [152] как основного формата представления документов в сети WWW. Сегодня гиперссылки поддерживаются во всех популярных форматах электронных документов: MS Word DOC [129], Adobe PDF [109], TeX [170] и т. д. Гиперссылки освобождают читателя от необходимости самостоятельно искать в сети упомянутый в тексте документ – для того, чтобы он начал загружаться в браузер пользователя, достаточно одного щелчка мышью на соответствующей гиперссылке. Возможность вставки в текстовый документ аудио- и видео-компонентов существенно обогащает спектр приложения электронных документов, в частности, значительно повышает эффективность обучающих программ или электронных учебных пособий.

**Технологии построения прототипов (шаблонов).** Практика хранения электронных документов "целиком", то есть в виде отдельного файла, объекта или поля базы данных, постепенно уходит в прошлое. Все чаще информация хранится в таблицах РСУБД в виде отдельных информационных блоков,

данных об их характеристиках и взаимосвязях с другими блоками. Отдельно хранятся прототипы (шаблоны, формы вывода), определяющие структуру и оформление документов разных типов. Когда от пользователя приходит запрос на определенный документ, система автоматически выбирает требуемый прототип и соответствующий набор информации из РСУБД, генерирует электронный документ и отсылает его пользователю[9]. Такая форма хранения информации позволяет намного более гибко работать с содержанием документов, чем хранение в виде единого файла или объекта [2]. Например, пользователь может получить в виде отдельного документа список всех заголовков и аннотаций документов определенной тематики. Многократно упрощается изменение дизайна документов: чтобы изменить оформление всех документов одного типа, достаточно изменить один прототип.

**Управление жизненным циклом и интеллектуальные агенты.** Управление отдельными этапами *жизненного цикла (document workflow)* документов все чаще осуществляется автоматически, что делает их полноправными участниками бизнес-процессов предприятия [45], [70], [72].

В традиционных системах автоматизации документооборота функцию управления выполняет специальный программный модуль системы (монитор), а документ содержит лишь набор атрибутов. Например, созданный руководителем подразделения документ может быть автоматически направлен на исполнение определенному сотруднику с учетом его загруженности (по результатам анализа очереди документов на исполнение), после чего руководитель сможет получать уведомления о принятии документа на исполнение, ходе работы над документом и результатам выполнения задания.

---

[9] Из распространенных систем работы с шаблонами можно отметить Perl Template Toolkit [161], PHP Smarty Template Engine [164], ASP Template [149], Macromedia ColdFusion MX [127], Sun Microsystems JavaServer Pages Technology [142] и др.

В последнее время все большее распространение получают специальные программы, автономные или встроенные в документ – *интеллектуальные агенты (information agents, intelligent agents, knowledge agents, mobile agents)* [39], [51], [79], которые автоматически анализируют содержание документов и происходящие с ними процессы и по результатам анализа предпринимают определенные действия. Например, такие программы могут выдавать уведомления об изменениях в документах, пересылать обновленные версии документов заинтересованным пользователям по заданному либо динамически формируемому списку, информировать ответственных лиц об устаревании содержащейся в документах информации и т. д.

**Интеллектуальный поиск и анализ документов.** Современные поисковые машины обеспечивают не только поиск документов по их реквизитам – номеру, заголовку, дате выпуска и т. д. (как в обычном бумажном архиве), но и поиск по результатам автоматического анализа всего содержания документа (полнотекстовая индексация). При этом поисковая система не просто выдает пользователю ссылки на все документы, где встречается указанное слово или словосочетание, а анализирует найденные документы и сортирует их таким образом, чтобы наиболее адекватные запросу документы находились вверху списка [28]. Та же система может автоматически вести полную статистику обращений к каждому документу и по запросу пользователя сортировать списки документов по их популярности и индексам цитируемости [29]. Пользователь, имеющий доступ к сети Интернет, более не ограничен архивом документов своего предприятия при поиске интересующей его информации – ему доступен весь огромный объем информации, представленной в свободном доступе в сети Интернет.

**Автоматическое реферирование.** Огромным потенциалом обладают технологии *автоматического реферирования (automatic abstracting, automatic summarization)* [43], [47]. Реферированием называется краткое изложение содержания исходного документа (оригинала) в новом документе (реферате).

Один из методов реферирования заключается в том, что из оригинала извлекаются наиболее значимые фрагменты, которые затем соединяются друг с другом – то есть, реферат представляет собой набор выдержек из оригинала. Более сложные методы предполагают анализ оригинала и изложение его содержания "своими словами". Для этого применяются технологии синтаксического разбора, семантического анализа с использованием сетей, фреймов и т. д. [9], а также генераторы текстов на естественных языках.

Современные системы позволяют успешно решать задачи реферирования отдельных текстов, при этом объем реферата составляет 5% – 30% от объема оригинала [101], [130]. Однако для многих насущных задач в этой сфере до сих пор нет удовлетворительных решений [42]: к ним относятся составление рефератов на основе нескольких источников, реферирование с большой степенью сжатия (1% и менее), реферирование аудио- и видео документов *(audio abstracting, video abstracting)* [31], [44].

**Машинный перевод.** Технологии автоматического перевода текстов на естественных языках долгое время развивались практически независимо от средств автоматизации документооборота [12], [35]. До сих пор системы машинного перевода поставляются преимущественно в виде отдельных программных продуктов [98], [105]. Однако по мере развития систем управления электронными документами машинный перевод, как и автоматическое реферирование, постепенно занимает в них свое место как одно из основных средств работы с документами. Несмотря на то, что основным языком международного общения до сих пор считается английский, повсеместного отказа от национальных языков при публикации документов не наблюдается – наоборот, число документов на основных языках мира постоянно растет. Чтобы отслеживать изменения даже в какой-то одной профессиональной области, может потребоваться мониторинг документов на 5-6 языках, что делает необходимым наличие средств автоматического перевода в корпоративных системах управления документами.

## 1.4. Динамические документы – новый класс электронных документов

В последнее время в сфере систем управления документами наблюдается несколько тенденций. С одной стороны, программные продукты, которые изначально разрабатывались для автоматизации разных аспектов управления документами, постепенно сближаются друг с другом по функциональности, интегрируются с продуктами других производителей. С другой – в повседневный оборот входят такие термины, как *виртуальный (virtual), живой (live, alive), оперативно доступный по запросу (on-line), эволюционирующий (evolving)* или *интеллектуальный (intelligent)* документ. Наконец, все больше компаний начинают позиционировать свои продукты как *системы управления знаниями*. Все эти процессы глубоко взаимосвязаны друг с другом и отражают разные стороны одного явления: в области управления документами происходит качественный переход, связанный с возникновением нового класса электронных документов – *динамических (виртуальных) документов*.

До тех пор, пока основной задачей было управление электронными версиями бумажных документов, в центре внимания находились задачи *автоматизации документооборота* или *автоматизации управления документами*. Перевод оборота документов в электронную форму позволил многократно упростить процесс подготовки и согласования документов, ускорить их доставку адресатам и прохождение документами своего жизненного цикла, усовершенствовать систему хранения и поиска документов [10] – но не добавил ничего принципиально нового в сам процесс передачи информации от одних людей к другим посредством документов. Статичный электронный документ создавался, утверждался, регистрировался, передавался на исполнение, двигался внутри организации и попадал в архив практически точно так же, как и бумажный: разница заключалась лишь в удобстве обращения с ним. Пользователь мог получить "из компьютера" только те

документы, которые были когда-то "в компьютер" занесены – тексты, таблицы, изображения, аудио- или видео-файлы и т. д.

Развитие технологий интеллектуального поиска и анализа документов, СУБД и технологий построения прототипов (шаблонов), автоматического реферирования и машинного перевода, разбора и генерирования текстов на естественных языках позволило реализовать в системах управления электронными документами совершенно новую функциональность, которая была в принципе недоступна при работе с бумажными документами [10]. Электронные информационные системы "научились" не просто выдавать пользователю те документы, которые были когда-то занесены "в компьютер", а автоматически генерировать по запросу пользователя новые документы с использованием доступной информации [1]. Такие документы получили в литературе название *динамических,* или *виртуальных,* документов [73]. В современных информационных системах управление отдельными этапами *жизненного цикла (document workflow)* динамических документов все чаще осуществляется автоматически, что делает их полноправными участниками бизнес-процессов предприятия [45], [70], [72].

Все это позволяет говорить о качественном переходе к информационным системам следующего поколения, ключевым элементом которых являются динамические документы – *эволюционирующие*, *интеллектуальные*, *живые*. Если раньше основные усилия разработчиков были сконцентрированы на автоматизации ввода бумажных документов в систему, развитии технологий хранения и поиска документов в базах данных и совершенствовании средств коллективной работы с документами, то сейчас акцент сместился на развитие технологий интеллектуальной обработки и анализа содержания документов, совершенствование средств поиска требуемых сведений и их представления в удобной для пользователя форме. Все современные системы управления документами в той или иной степени "умеют" автоматически генерировать новые документы на основе доступной информации [33], [71].

Анализ последних тенденций в сфере электронного документооборота показывает, что разработка методов и технологий работы с динамическими документами становится магистральным направлением развития современных информационных систем [1], [32], [39], [73], [150]. Практически все компании, предлагающие системы управления документами, так или иначе работают над этой задачей [113], [118], [120], [123], [129], [133], [134], что в ближайшем будущем приведет к глубокой перестройке существующих бизнес-процессов и схем работы с информацией.

## 1.5. Направления развития динамических документов

Можно ожидать, что развитие методов и технологий работы с динамическими документами будет идти по ряду направлений.

**Во-первых,** необходимо развитие технологий хранения информации и специализированных документно-ориентированных баз данных в соответствии с технологией динамических документов [50]. Необходима дальнейшая стандартизация моделей электронных документов, решение проблем совместимости документов разных форматов, извлечение структурированной информации из неструктурированных документов [166], [155].

**Во-вторых,** необходимо развитие технологий поиска информации, интеллектуального анализа и индексирования электронных документов [37]. Методы синтаксического, семантического и лингвистического анализа текстов должны быть задействованы таким образом, чтобы образ документа в поисковой системе (список ключевых слов, регистрационная карточка и т. п.) в максимальной степени соответствовал реальному содержанию документа [41], [49]. Необходимо развитие технологий сравнительного анализа текстов для автоматического определения первоисточника информации, маршрутов ее распространения и вносимых при этом изменений [52].

**В-третьих,** необходимо совершенствование методик построения и технологий разбора запросов, развитие естественно-языковых интерфейсов

[83]. Сегодня процесс поиска информации в сети является итерационным: пользователь строит запрос, анализирует полученные результаты, самостоятельно переформулирует запрос и т. д. – до тех пор, пока очередная итерация не даст удовлетворительного для него результата. Такая схема более не может считаться эффективной: система должна автоматически анализировать точность, полноту и непротиворечивость запроса пользователя и предлагать ему варианты уточнения запроса, например, с использованием технологий *кластеризации (clustering)* найденных по запросу документов [38], [46], анализа наиболее часто встречающихся запросов, статистики популярности и индексов цитируемости документов [29] и т. д.

**В-четвертых,** необходимо дальнейшее развитие технологий построения прототипов (шаблонов), разработка программного обеспечения для интеллектуального построения прототипов и формирования документов. Система должна автоматически создавать прототипы (формы вывода), которые в максимальной степени соответствуют запросу конкретного пользователя и доступной информации [26]. Необходимо развитие технологий графического представления структурированной информации: построение таблиц и графиков на основе массивов данных, определение трендов (тенденций) и формирование предсказаний (прогнозов) с оценкой их достоверности [34].

**В-пятых,** необходимо развитие технологий автоматического реферирования и машинного перевода. Электронная информационная система должна автоматически выбирать из найденных по запросу данных наиболее важные, критически значимые сведения, факты и утверждения и переводить их на язык запроса. Соответствующее программное обеспечение должно стать основной составляющей интеллектуальных систем управления документами следующего поколения [27], [36].

# 2. Автоматизация подготовки динамических документов

В данной главе представлена технология автоматизированной подготовки динамических документов, основанная на хранении структурированной информации в таблицах РСУБД и применении прототипов (шаблонов). Эта технология описана на примере системы автоматизированной подготовки и публикации документов на корпоративном сайте, которая была разработана автором в сотрудничестве с коллегами [2] и внедрена в компании "Телеком Транспорт" [106] в 2000-2002 гг.

*В разделе 2.1* обоснована необходимость автоматизации подготовки и публикации документов. В *разделе 2.2* сформулированы требования, которым должна удовлетворять система автоматизированной подготовки и публикации документов. *В разделе 2.3* описана технология хранения структурированной информации в таблицах РСУБД, технология конструирования прототипов и схема работы интерпретатора. *В разделе 2.4* представлена методология подготовки основных типов документов и описана архитектура системы автоматизированной подготовки и публикации документов. Наконец, *в разделе 2.5* описаны критерии выбора программного обеспечения для построения информационной системы, основанной на использовании динамических документов.

## 2.1. Необходимость автоматизации работы с документами

Экспоненциальный рост числа документов в корпоративных сетях, наблюдаемый в последние годы, делает все более очевидной необходимость ре-инжиниринга бизнес-процессов (BPR) и автоматизации работы с документами [10]. В данной главе представлена технология автоматизированной подготовки динамических документов, основанная на хранении структурированной информации в таблицах РСУБД и применении прототипов (шаблонов) [2]. Эта

технология описана на примере системы автоматизированной подготовки и публикации документов на корпоративном сайте компании, которая занимается производством или внедрением технологически сложной продукции.

В современных условиях практически любая компания должна осуществлять адекватную информационную поддержку своей деятельности. Под информационной поддержкой далее будет пониматься комплекс мер по подготовке и распространению информации, цель которых – способствовать решению задач, стоящих перед компанией. Как правило, одна из основных задач информационной поддержки – привлекать внимание к предлагаемой компанией продукции.

Многие отечественные компании успешно применяют современные приемы и методы информационной поддержки для решения своих маркетинговых задач (например, в сфере товаров массового спроса). В то же время, есть ряд направлений, где эффективность информационной поддержки все еще остается достаточно низкой. На наш взгляд, наиболее остро проблема обеспечения адекватной информационной поддержки сегодня стоит в сфере технологически сложной продукции – дорогого наукоемкого оборудования, программного обеспечения, комплексных технологических решений.

Чтобы принять решение о целесообразности использования той или иной технологически сложной продукции, потенциальному потребителю необходим значительно больший объем информации о ней, чем в случае товаров массового спроса. Эта информация, как правило, подвергается тщательному анализу, сопоставляется с данными из других источников, сравнивается с информацией о продукции конкурентов. Нередко бывает, что решение принимается в течение нескольких месяцев с участием целого коллектива специалистов.

Успехи в конкурентной борьбе в сфере технологически сложной продукции во многом зависят от того, насколько полно и точно компания удовлетворяет информационные потребности потенциальных потребителей и насколько убедительные доводы она приводит в пользу предлагаемой

продукции. Поэтому для компаний, которые занимаются производством или внедрением технологически сложной продукции, важнейшей составляющей информационной поддержки является создание, распространение и регулярное обновление широкого спектра информационных документов.

Под **информационным документом** далее будет пониматься документ, который привлекает внимание потенциальных потребителей к предлагаемой продукции, содержит значимую для них информацию о предлагаемой продукции и дает убедительные обоснования целесообразности ее использования.

С развитием сети Интернет традиционные печатные информационные документы (каталоги, брошюры, листовки, публикации в средствах массовой информации и т. д.) постепенно теряют роль непосредственного источника информации, и становится скорее средством формирования имиджа компании. Основным источником оперативной информации для потенциальных потребителей технологически сложной продукции все в большей степени становятся электронные информационные документы, в первую очередь – документы, публикуемые на корпоративном сайте компании.

Для подавляющего большинства современных компаний, которые занимаются производством или внедрением технологически сложной продукции, корпоративный сайт является прежде всего системой подготовки и публикации в сети Интернет различных информационных документов: новостей (пресс-релизов), технических описаний продукции, статей о предлагаемых решениях и используемых технологиях, справок о реализованных проектах, портретов компаний-партнеров и т. д.

Как показывает опыт, подготовка и публикация на корпоративном сайте необходимого количества качественных информационных документов является сложной методологической и технологической задачей. В отсутствие единого подхода, без достаточного уровня автоматизации создание и обновление каждого документа требует неоправданно высоких затрат времени и

творческих усилий, и задача в целом решается неудовлетворительно. В то же время, использование динамических документов позволяет построить эффективную систему подготовки и публикации информационных документов на корпоративном сайте [2].

В данной главе представлена технология автоматизированной подготовки динамических документов, описанная на примере системы автоматизированной подготовки и публикации информационных документов на корпоративном сайте. Рассмотрены вопросы стандартизации структуры документов и системы связей между ними, описана технология конструирования прототипов (шаблонов) динамических документов, рассмотрены критерии выбора программного обеспечения, предложена методология подготовки основных типов информационных документов.

## 2.2. Требования к системе подготовки документов

### 2.2.1. Типы документов

Компания, которая занимается производством или внедрением технологически сложной продукции, использует для обеспечения информационной поддержки своей деятельности несколько десятков типов информационных документов. Как правило, компания имеет разную потребность в различных типах документов и использует их с разной интенсивностью: например, технические описания продукции могут исчисляться тысячами наименований, а портрет компании обычно существует лишь в нескольких вариантах.

Та же картина обычно наблюдается и на корпоративном сайте: несколько типов документов являются основными, публикуются в большом количестве и обновляются регулярно (например, новости); другие типы документов для компании оказываются дополнительными, публикуются в значительно меньшем объеме и обновляются реже (например, описания проектов); наконец,

несколько типов документов обычно представлены в единственном экземпляре (например, описание истории компании).

Документы одного типа выполняют одинаковые функции, имеют сходную внутреннюю структуру и способы взаимосвязи с другими информационными документами. Каждый документ, в зависимости от его типа, может включать в себя следующие структурные элементы: один или несколько вариантов заголовка (краткий, полный), один или несколько вариантов текста (резюме, подробное описание), иллюстрации (фотографии, схемы), ключевые слова, дату публикации или обновления, дату устаревания и т. д. [10].

В данной главе мы подробно рассмотрим два наиболее важных типа информационных документов: новости (пресс-релизы) и технические описания продукции. Этот выбор не случаен: как показывает опыт, практически любая компания, которая занимается производством или внедрением технологически сложной продукции, стремится регулярно публиковать новости (пресс-релизы) и поддерживать в актуальном состоянии технические описания для всего ассортимента предлагаемой продукции [2].

Система подготовки и публикации документов должна предусматривать поддержку корпоративного стиля, в частности, единообразие документов одного типа. Поэтому одной из важных задач при разработке системы публикации документов является стандартизация структуры документов разных типов и разработка стандартных форм ввода для их публикации.

### 2.2.2. Классификация документов

Множество документов одного типа можно классифицировать по одному или нескольким признакам, каждый из которых может принимать ряд дискретных значений. Например, новости можно классифицировать по следующим признакам: тип новости Т ("новость компании", "новость сайта", "новость партнера", "новость отрасли" и т. д.), упоминаемые модели продукции М ($М_1$, …, $М_n$), упоминаемые компании-партнеры П ($П_1$, …, $П_m$). Возможность многоуровневой классификации множества объектов информационной системы

по разным признакам делает поиск информации намного более эффективным, гибким и удобным. Например, можно реализовать следующие схемы навигации по информационной системе (возможные последовательности перехода по оглавлениям и документам) для доступа к новости X:

- навигация по типу новости: главная страница сайта → оглавление всех новостей → оглавление всех новостей типа $T_i$ → новость X;
- навигация по моделям продукции: главная страница сайта → оглавления семейств продукции разных уровней → описание модели $M_j$ (со списком ссылок на все новости, в которых упоминается модель $M_j$) → новость X;
- навигация по компаниям-партнерам: главная страница сайта → оглавление всех партнеров → описание партнера $П_k$ (со списком ссылок на все новости, в которых упоминается партнер $П_k$) → новость X.

Многоуровневая классификация множества объектов по сути является лексикографическим упорядочиванием этого множества по определенному набору признаков. В нашем примере каждый документ типа "новость" обладает признаками T, M, П и ID (уникальный идентификатор). На рис. 2.1 представлены два варианта лексикографического упорядочивания множества документов типа "новость" – по наборам признаков {T, ID} и {M, ID}, и два соответствующих им способа многоуровневой классификации.

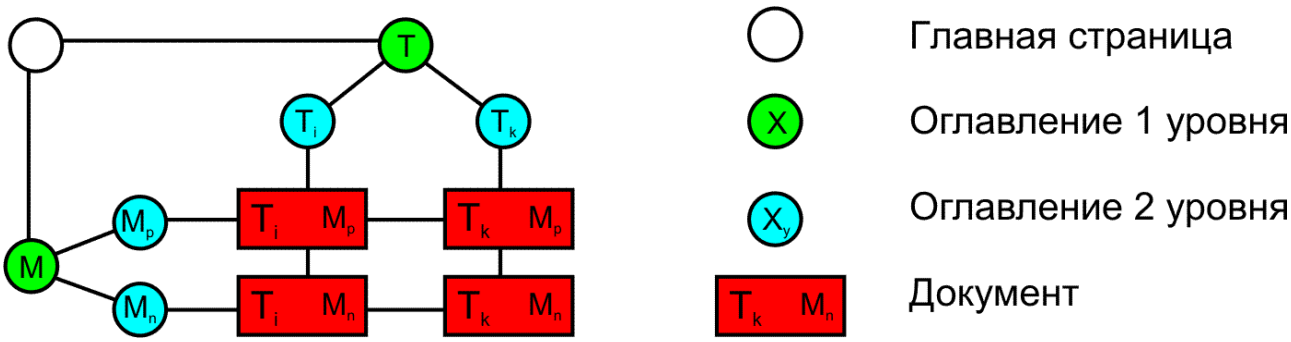



*Рис. 2.1. Многоуровневая классификация множества документов по разным признакам*

Реализация нескольких систем классификации для документов каждого типа значительно усложняет задачу поддержки информационной системы. Добавление в такую систему нового документа, перемещение документа из одного раздела в другой или удаление документа сопряжено с внесением изменений сразу в несколько оглавлений и списков, причем их набор индивидуален для каждого типа документов. Очевидно, что осуществлять такие изменения вручную крайне неудобно.

Следовательно, система подготовки и публикации документов должна предусматривать возможность реализации различных систем классификации документов с автоматическим формированием всех оглавлений и списков. Заметим, что это требование уже само по себе наводит на мысль об использовании базы данных для хранения некоторых элементов документов (заголовок, идентификатор) и автоматическом генерировании списков и оглавлений на основе информации из этой базы данных.

### 2.2.3. Взаимосвязи между документами

Как правило, потенциальный потребитель компании, которая занимается производством или внедрением технологически сложной продукции, интересуется рядом тематически близких документов. Рассматривая возможность приобретения определенной модели технологически сложной продукции, потенциальный потребитель, скорее всего, будет интересоваться не только описанием данной модели, но и описаниями функционально близких моделей, новостями с упоминанием о данной модели, информацией о проектах, в которых использовалась данная модель и т. д.

Сама идея сети Интернет как "всемирной паутины" предполагает, что в такой ситуации каждый документ должен быть снабжен гиперссылками на тематически близкие к нему документы. Каждая новость должна содержать гиперссылку на описание упоминаемой в ней продукции, каждое описание продукции – гиперссылки на все новости, в которых она упоминается и т. д.

Плотная "обвязка" информационных документов гиперссылками делает поиск интересующей информации намного более быстрым, удобным и эффективным.

Однако при таком подходе число гиперссылок пропорционально квадрату от общего числа документов. Учитывая, что общее число документов на корпоративном сайте компании, которая занимается производством или внедрением технологически сложной продукции, может достигать нескольких десятков тысяч, очевидно, что осуществлять "обвязку" документов гиперссылками в требуемом объеме вручную практически невозможно. Этот процесс должен быть в значительной степени автоматизирован.

Это означает, что система подготовки и публикации документов должна предусматривать удобный интерфейс для ввода и изменения информации о взаимосвязях между документами, а также обеспечивать автоматическое формирование гиперссылок на основе имеющейся информации о взаимосвязях между документами. Эта задача может быть эффективно решена только при использовании динамических документов с хранением информации в РСУБД.

Напомним, что такое динамический HTML-документ и чем он отличается от статичного HTML-документа. Традиционный статичный HTML-документ – это отдельная HTML-страница, которая связана гиперссылками с другими документами сайта. Статичные HTML-документы заносятся на сайт, хранятся на сервере и выдаются пользователям в виде готовых HTML-страниц. Чтобы в нескольких статичных HTML-документах появился новый элемент (например, гиперссылка), необходимо вручную внести его в каждый HTML-документ.

Динамические HTML-документы создаются сервером в ответ на запрос пользователя. Структурированная информация, которая используется для построения документов, хранится в таблицах РСУБД. Отдельно на сервере хранятся прототипы документов – в случае сайта это "скелеты" HTML-документов, определяющие формат и дизайн документов разных типов. Когда от пользователя приходит запрос на какой-либо документ, система на основании этого запроса автоматически выбирает требуемый прототип и

соответствующую ему информацию из базы данных, формирует HTML-документ и отсылает его пользователю.

### 2.2.4. Поддержка системы

Поддержка информационной системы, построенной с использованием динамических документов, включает в себя, как минимум, три относительно независимых процесса. Во-первых, это информационное наполнение системы – занесение и редактирование информации в РСУБД. Во-вторых, это разработка и изменение прототипов (шаблонов) документов разных типов. В-третьих, это обеспечение корректного взаимодействия между программным обеспечением системы и пользователя.

На начальном этапе развития электронной информационной системы все эти функции нередко выполняет один человек. Однако по мере развития системы неизбежно возникает необходимость распределить задачу поддержки между *редактором* (который занимается информационным наполнением базы данных), *разработчиком динамических документов* (который создает и редактирует прототипы документов) и *программистом* (который отвечает за организацию корректного взаимодействия между программным обеспечением сервера и пользователя).

Таким образом, еще одно требование к системе подготовки и публикации документов – обеспечивать эффективное разделение работы редактора, разработчика документов и программиста. Редактору необходим удобный интерфейс для работы с базой данных; разработчику документов нужен четкий и прозрачный язык описания прототипов, не перегруженный набором понятных лишь программисту программных кодов; программисту требуется ясный и структурированный программный код, который он мог бы менять независимо от разработчика документов.

### 2.2.5. Список требований

Обобщая сказанное в подразделах 2.2.1-2.2.4, можно сформулировать набор требований, которым должна удовлетворять система подготовки и

публикации документов на корпоративном сайте и вообще любая система автоматизированной подготовки динамических документов:

- стандартизация структуры документов и автоматизация информационного наполнения базы данных за счет использования стандартных форм ввода;
- возможность реализации нескольких систем классификации для документов одного типа с автоматическим формированием оглавлений;
- автоматическая "обвязка" документов гиперссылками на основе информации о взаимосвязях между ними;
- удобный интерфейс для редактирования документов, описания признаков (классификации) документов и взаимосвязей между ними;
- эффективное разделение процессов информационного наполнения базы данных, конструирования прототипов документов и разработки программных модулей.

## 2.3. Технология построения динамических документов

### 2.3.1. Хранение структурированной информации в РСУБД

Структурированная информация, которая используется для построения динамических документов, как правило, хранится в реляционной системе управления базами данных (РСУБД). РСУБД представляет собой набор двумерных таблиц, на пересечении строк и столбцов которых находятся поля данных; таблицы связаны между собой посредством совместно используемых столбцов данных, называемых внешними ключами.

Архитектура таблиц РСУБД, а также источники информации для ее наполнения и методы конвертировании этой информации могут быть самыми разными. Рассмотрим простейший случай, когда наполнение базы данных осуществляется путем конвертирования статичных документов. В этом случае информация из всех документов одного типа заносится в одну таблицу РСУБД, причем каждый документ помещается в одну строку таблицы. Отдельные

структурные элементы документа (заголовок, текст, дата обновления, ссылки на иллюстрации и т. д.) сохраняются в различных полях данных одной строки. Порядок следования полей данных в строке одинаков для всех строк таблицы, таким образом, в каждом столбце таблицы содержатся одинаковые структурные элементы всех документов, табл. 2.1.

| **Первичный ключ** | **Заголовок** | **Текст** | **…** | **Дата обновления** |
|---|---|---|---|---|
| 1 | Заголовок_1 | Текст_1 | … | Дата_обновления_1 |
| 2 | Заголовок_2 | Текст_2 | … | Дата_обновления_2 |
| … | … | … | … | … |

*Таблица 2.1. Пример таблицы РСУБД для хранения документов одного типа.*

Для корректной работы РСУБД каждая таблица должна содержать столбец с уникальными идентификаторами (первичными ключами) документов. Каждой строке таблицы присваивается уникальное значение первичного ключа. Доступ к любому полю данных в любой таблице РСУБД можно получить, зная имя таблицы, первичный ключ и название столбца. В простейшем случае первичным ключом может служить порядковый номер, под которым документ заносится в таблицу.

Каждая таблица может также содержать один или несколько столбцов с внешними ключами – уникальными идентификаторами (первичными ключами) документов из других таблиц РСУБД. Например, для того, чтобы отразить взаимосвязь между документами двух типов, достаточно в одной из соответствующих таблиц РСУБД использовать дополнительный столбец, в котором для документов одного типа хранились бы первичные ключи связанных с ним документов другого типа.

Проектирование базы данных – одна из первых и основных задач на этапе разработки системы подготовки и публикации документов на корпоративном сайте. Необходимо стандартизировать структуру документов каждого типа, разработать систему связей между документами разных типов и на основе этого определить наиболее целесообразную структуру таблиц РСУБД. Примеры

таблиц для некоторых типов документов приведены в следующем разделе, здесь мы коротко рассмотрим основные принципы их построения.

Обычный бумажный документ включает в себя ряд информационных блоков: номер или другой идентификатор, заголовок, резюме, текст с таблицами и иллюстрациями, заключение, дату создания, подпись автора и т. д. В таблице РСУБД могут заноситься как эти информационные блоки, так и различные дополнительные (служебные) элементы: отметка о готовности к публикации, список ключевых слов, указатели на другие документы и т. д.

В отдельных полях необходимо хранить все информационные блоки документа, которые выполняют самостоятельные функции. Это могут быть, например, заголовок, резюме и одна из иллюстраций, используемые для автоматического генерирования оглавлений; даты создания и устаревания, по которым осуществляется сортировка документов, и т. д. В отдельных полях также должны храниться все служебные элементы документа, используемые для поиска, классификации, сортировки документов и других функций.

На способах хранения иллюстраций следует остановиться отдельно. Коммерческие РСУБД позволяют хранить в полях РСУБД достаточно большие объемы данных, в том числе и иллюстрации. При использовании РСУБД с открытым кодом иллюстрации обычно хранятся на сервере отдельно. В любом случае, если только число и расположение иллюстраций в документе не задано жестко, в тексте документа приходится тем или иным образом указывать ссылки на требуемые иллюстрации.

Основной текст документа (абзацы, подзаголовки, списки, таблицы, ссылки на иллюстрации и т. д.), который используется для построения динамических документов как самостоятельный и неделимый информационный блок, целиком хранится в одном поле соответствующей таблицы РСУБД.

### 2.3.2. Интерпретатор и взаимодействие компонентов

Структурированная информация хранится на сервере в базе данных во внутреннем формате РСУБД и пользователям непосредственно недоступна. Поэтому необходимым компонентом информационной системы является специальная программа (*динамический движок)*, которая по запросу пользователя выбирает требуемый прототип документа и соответствующую ему информацию из базы данных, формирует электронный документ и отсылает его пользователю.

Опыт показывает, что такую программу удобно строить из двух модулей, один из которых обрабатывает запрос пользователя к серверу, выбирает прототип и информацию из РСУБД (модуль разбора запроса), а другой на основе полученного набора информации и прототипа формирует электронный документ и отправляет его пользователю (модуль "сборки" документа, или интерпретатор). Схема взаимодействия пользователя, сервера, базы данных и интерпретатора представлена на рис. 2.2.

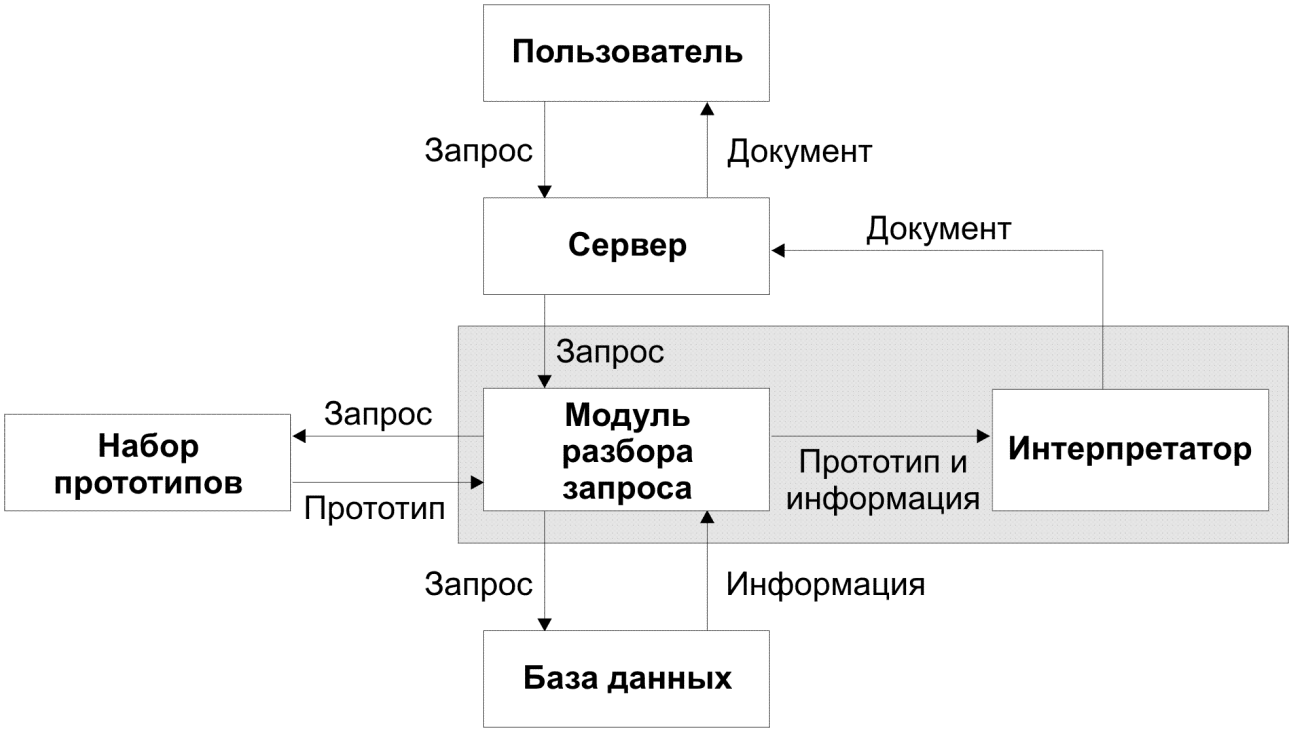



*Рис. 2.2. Схема взаимодействия пользователя, сервера, РСУБД и интерпретатора.*

Отметим, что не все системы автоматизированной подготовки динамических документов построены на основе описанной выше схемы с четким разделением прототипа и интерпретатора. На начальном этапе построения системы часто кажется, что проще встраивать программный код, описывающий взаимодействие с сервером и базой данных, непосредственно в прототип, описывающий конструкцию документа (при этом многие программисты называют такие конструкции также прототипами). Это ошибка, и не только терминологическая – такое пренебрежение в итоге всегда приводит к дополнительным затратам.

При развитии системы, построенной с использованием таких "прототипов", неизбежно возникает ситуация, когда в нагромождении программного кода и языка описания прототипа становится сложно разбираться как программисту, так и разработчику документов, а об их относительно независимой работе не может идти и речи. Любое изменение конструкции документов требует значительных усилий; усложняется задача обучения персонала; возрастает трудоемкость поддержки системы в целом.

### 2.3.3. Разработка прототипов документов

Прототип представляет собой "скелет" документа, в котором с помощью специальных языковых конструкций (*меток*, или *макросов*) указаны места, куда интерпретатор должен вставить данные из РСУБД. В прототипе могут предусматриваться как места для вывода одиночных значений (заголовок новости в прототипе новости), так и места для вывода списков значений, в том числе вложенных, с указанием оформления для каждого элемента списка (список заголовков новостей в прототипе оглавления новостей).

Прототип документа определяет общую структуру документа (взаимное расположение всех элементов) и содержит элементы, общие для документов данного типа: навигационные меню, иллюстрации (логотипы, фон), общие информационные блоки (контактная информация), метаданные (ключевые слова), ссылки на используемые скрипты и таблицы стилей (CSS) и т. д.

Подчеркнем, что прототип не содержит программный код, описывающий взаимодействие с сервером и базой данных, а содержит только стандартный код языка разметки (в случае сайта – HTML) и метки. Все функции обработки данных – замену меток конкретными значениями из РСУБД, определение числа элементов в списках; выбор нужного варианта оформления определенных элементов документа в зависимости от конкретных значений данных, полученных из РСУБД и т. д. – выполняет модуль "сборки" документа. Язык описания прототипа можно рассматривать как язык программирования более высокого уровня, чем язык описания взаимодействия с сервером и базой данных. В этом смысле прототип может рассматриваться как программа на этом языке, модуль "сборки" документа – как ее интерпретатор, разработчик динамических документов – как программист, а программист – как системный программист условной "виртуальной машины".

Рассмотрим принцип построения прототипов и работу интерпретатора на следующем примере. Предположим, поставлена задача создать документ, где бы отображался список всех новостей за текущий месяц, причем требуется, чтобы для новостей типа *"новость компании"* выводилась дата публикации, заголовок и иллюстрация, а для всех остальных типов новостей – только дата и заголовок. Этот документ должен формироваться по запросу пользователя на основе информации из таблицы *Новости* РСУБД.

Пусть также требуется, чтобы заголовок каждой новости в данном списке являлся гиперссылкой на документ с полным описанием новости. Будем считать, что URL документа с полным описанием новости строится по следующей схеме: `http://www.company.ru/?do=fullnew&new_id=X`[10],

---

[10] Фактически, после символа "?" следует инструкция для модуля разбора запроса: указывается требуемый прототип и алгоритм выборки данных (`fullnew`) и требуемый документ (`new_id=X`).

где X – первичный ключ данной новости в таблице *Новости* РСУБД, например, порядковый номер ее публикации.

Таким образом, в прототипе данного документа необходимо описать два варианта представления строки списка новостей, а в интерпретаторе – задать критерии выбора нужного варианта в зависимости от значения поля *тип новости*. Кроме того, модуль разбора запроса должен при получении запроса на данный документ выдавать интерпретатору выборку из таблицы *Новости* РСУБД со следующими полями: *первичный ключ, тип новости, дата публикации, заголовок, ссылка на иллюстрацию.*

Пример прототипа данного документа представлен в прил. 1. Строки стандартного HTML-кода ради простоты пропущены – их заменяют троеточия. Метки, которые заменяются конкретными значениями из РСУБД, ограничены тегами {#…#}; служебные метки для интерпретатора – тегами {##…##}. Отметим, что используемый синтаксис языка описания прототипов условен; в реальности существуют как многочисленные стандартные варианты синтаксиса, так и возможность создать собственный синтаксис.

Предположим, модуль обработки запроса выдал интерпретатору выборку из таблицы *Новости* РСУБД, представленную в табл. 2.2.

| **Первичный ключ** | **Тип новости** | **Дата публикации** | **Ссылка на иллюстрацию** | **Заголовок** |
|---|---|---|---|---|
| 127 | новость сайта | 01.03.03 | news/site/17.jpg | Смена дизайна |
| 128 | новость компании | 05.03.03 | news/company/4.jpg | Расширение ассортимента |
| 129 | новость отрасли | 10.03.03 | news/industry/43.jpg | Прогноз на II квартал |

*Таблица 2.2. Пример выборки из таблицы "Новости".*

Когда управление переходит к интерпретатору, он начинает построчно обрабатывать код прототипа: статичные части прототипа без изменений переписывает в итоговый документ, отдельные метки (например,

{#Текущий_месяц#}) заменяет соответствующими значениями. Когда интерпретатор доходит до списка новостей (о начале которого сигнализирует метка {##Начало_списка_новостей##}, а о конце – метка {##Конец_списка_новостей##}), он считывает его целиком и начинает построчную обработку выборки.

Условием, которое определяет выбор варианта оформления новости, в нашем случае является тип новости. Если поле *тип новости* в обрабатываемой строке выборки имеет значение *"новость компании"*, то интерпретатор выбирает первый вариант оформления, если нет – второй. Выбрав нужный вариант оформления, интерпретатор пишет в итоговый документ соответствующий набор строк прототипа, заменяя при этом метки конкретными значениями из обрабатываемой строки, и переходит к следующей строке выборки.

После того, как обработка выборки закончена, интерпретатор начинает обрабатывать код прототипа дальше, начиная со строки, следующей за служебной меткой {##Конец_списка_новостей##}. Документ в формате HTML, который интерпретатор отправит пользователю, представлен в прил. 2.

### 2.3.4. Преимущества динамических документов

Использование динамических документов с хранением информации в РСУБД и применение описанной выше технологии построения прототипов и схемы работы интерпретатора позволяет построить комплексную систему автоматизированной подготовки динамических документов, которая полностью удовлетворяет всем сформулированным ранее требованиям [2]. Кроме того, эта система позволяет эффективно решать еще целый ряд задач, возникающих при публикации документов в сети Интернет.

**Автоматическая обвязка документов ссылками.** Установить взаимосвязь между документами разных типов позволяет использование внешних ключей. Пусть, например, необходимо обеспечить автоматическую "обвязку" новостей ссылками на описания упоминаемой в ней продукции (и

наоборот). Для этого в таблице *Новости* РСУБД нужно предусмотреть дополнительное поле, в котором редактор мог бы указывать для каждого документа типа "новость" список первичных ключей всех документов типа "описание продукции", которые имеют отношение к этой новости.

Зная первичный ключ документа в таблице, можно извлечь из РСУБД любые элементы этого документа. Например, получив запрос пользователя на определенную новость, динамический движок сможет извлечь из РСУБД выборку первичных ключей и названий всех описаний продукции, которые имеют отношение к данной новости, и автоматически сформировать оглавление всех связанных с новостью описаний продукции, где каждое название будет являться ссылкой на полное описание продукции.

**Автоматическое формирование оглавлений.** Часто требуется предусмотреть возможность различных разбиений совокупности документов на подмножества на основании разных признаков, например: $X = \bigcup x_i, x_i \cap x_j = 0$ и $X = \bigcup y_i, y_i \cap y_j = 0$. Например, может потребоваться сгруппировать описания продукции по сериям, а серии – по типам продукции, и представить несколько вариантов оглавлений: список всех типов продукции, список всех серий одного типа, список всех описаний продукции одной серии. При этом списки и оглавления всех уровней должны формироваться автоматически.

Для этого необходимо ввести в рассмотрение новый объект – *семейство* N-го уровня (например, семейство 1 уровня – тип продукции, семейство 2-го уровня – серия продукции). Объект "семейство" может включать в себя название семейства, краткое описание и другую информацию. Для хранения таких объектов и информации об их иерархии используется отдельная таблица РСУБД, которая в простейшем случае состоит из трех столбцов, прил. 5.

Используя информацию об иерархии семейств и о принадлежности документа к определенному семейству, можно разработать прототипы для автоматического формирования оглавлений любого уровня. В совокупности с автоматической "обвязкой" документов ссылками это кардинально упрощает

классификацию и упорядочивание документов на корпоративном сайте. Например, для переноса документа из одного семейства в другое требуется просто изменить одно значение в соответствующей таблице РСУБД.

**Удобный интерфейс и стандартные формы ввода документов.** Для того чтобы редактор сайта мог осуществлять наполнение РСУБД информацией через обычный браузер, необходимо разработать стандартные формы ввода документов на основе соответствующих прототипов. Технология автоматического генерирования форм ввода по запросу редактора на основе прототипов и информации из РСУБД полностью аналогична описанной выше технологии генерирования документов по запросу пользователя.

**Разделение работы редактора, разработчика документов и программиста.** Формы ввода позволяют редактору заниматься информационным наполнением базы данных независимо от разработчика документов и программиста; четкое разделение прототипов и интерпретатора дает возможность разработчику документов и программисту работать независимо друг от друга. *Независимость* в данном случае означает не полное отсутствие взаимодействий между членами команды, а возможность четкого разграничения их зон ответственности.

**Автоматическое генерирование заголовка и метаданных HTML-документа.** Заголовок HTML-документа может легко формироваться автоматически; например: "`название компании → новости компании → заголовок новости`". Поля метаданных, которые содержат краткое описание документа для роботов поисковых систем, также могут автоматически заполняться в процессе формирования документа интерпретатором с использованием резюме документа, списка ключевых слов, информации об авторе и т. д.

**Регулирование доступа к документам.** Отметка о готовности документа к публикации в простейшем случае принимает два значения: *"готов"* и *"не*

*готов"*. Этот диапазон значений можно легко расширить, введя, таким образом, несколько уровней доступа к документам.

**Надежное решение навигационных задач.** Для удобства пользователя средства навигации часто дублируются: например, ссылки на следующую и предыдущую главы документа ставятся в начале и конце текущей главы документа, в боковых меню и т. д. Ручная простановка ссылок может легко привести к ошибкам. При автоматическом генерировании динамического документа такие ошибки невозможны в принципе, так как все ссылки создаются на основе одной и той же информации из РСУБД.

## 2.4. Архитектура системы подготовки документов

В данном разделе рассмотрена методология подготовки двух основных типов информационных документов – новостей (пресс-релизов) и технических описаний продукции, и описана архитектура системы автоматизированной подготовки динамических документов на примере системы автоматизированной подготовки и публикации информационных документов на корпоративном сайте [2].

В качестве примера рассмотрим корпоративный сайт, на котором представлены два основных типа документов (новости и технические описания продукции) и один дополнительный тип документов (портреты компаний-партнеров), а также отдельные документы (портрет компании, контактная информация компании, схема проезда в офис компании и т. д.).

### 2.4.1. Новости (пресс-релизы)

Новости (пресс-релизы)[11] являются самым популярным жанром информационных документов, у них наиболее широкая аудитория читателей. Роль новостей в конкуренции за потребителя информации исключительно

---

[11] Термины "новость" и "пресс-релиз" (в отношении документа) являются синонимами [2].

высока, не случайно в любом средстве массовой информации новостная (информационная) служба считается одним из основных подразделений. Создание и распространение новостей (пресс-релизов) является важнейшей составляющей информационной поддержки деятельности компании.

Краткие информационные сообщения на сайте используются, прежде всего, для того чтобы привлечь внимание потенциальных потребителей к предлагаемой продукции или определенным направлениям деятельности компании, заинтересовать их, побудить искать более подробную информацию о предмете сообщения. В этом смысле новости (пресс-релизы) похожи на обычные рекламные объявления, и при их создании можно широко опираться на те приемы и методы, которые применяются в рекламном деле.

Типовая новость (пресс-релиз) компании, которая занимается производством или внедрением технологически сложной продукции, состоит из следующих информационных блоков:

- дата публикации;
- заголовок (возможно, два варианта: краткий и полный);
- иллюстрация (возможно, несколько иллюстраций);
- резюме (суть новостного сообщения в одном-двух предложениях);
- основной текст (две части: для специалистов и для широкой аудитории);
- заключение (конкретные инструкции читателю).

Целевую аудиторию новости (пресс-релиза) на корпоративном сайте обычно составляют две основные категории читателей: специалисты, которые профессионально разбираются в предмете информационного сообщения, и широкая аудитория читателей из различных смежных областей, которые имеют общее представление о предмете. В соответствии с принципами *technical writing* (см. пункт 2.4.2), текст новости в данном случае должен включать в себя две части: для специалистов и для широкой аудитории.

Специалисты, которые профессионально разбираются в предмете, как правило, предпочитают самостоятельно интерпретировать полученную

информацию. В части текста, предназначенной для специалистов, нужно четко, с минимумом комментариев изложить факты, составляющие предмет сообщения (характеристики продукции, особенности технологии, научные результаты и т. д.). При этом можно без объяснений употреблять специальные термины, обозначения, сокращения, аббревиатуры.

Читатели, которые имеют лишь общее представление о предмете информационного сообщения, но не разбираются в нем профессионально, не всегда могут самостоятельно сделать значимые выводы и в полной мере осознать смысл новости. В части текста, предназначенной для широкой аудитории, необходимо разъяснить значение фактов, составляющих предмет сообщения, "разжевать" их (например, сравнить с широко известными), повторить эти факты в более простой и доступной форме.

На корпоративном сайте может быть предусмотрен как прототип отдельной новости, так и несколько прототипов списков новостей (список всех новостей одного типа, список всех новостей за определенный период времени и т. д.). Кроме того, ссылки на новости могут выводиться и на других динамических документах: главной странице, оглавлениях, описаниях оборудования, портретах компаний-партнеров и т. д.

В качестве примера рассмотрим случай, когда имеется прототип отдельной новости и прототип списка новостей одного типа, и кроме того, ссылки на новости выводятся на главной странице сайта. Типовой состав таблицы NEWS в РСУБД корпоративного сайта приведен в прил. 3. Типовая схема связей таблицы NEWS, прототипа отдельной новости, прототипа списка новостей одного типа и главной страницы сайта приведена в прил. 7.

### 2.4.2. Технические описания продукции

Основное назначение технических описаний продукции – сообщать пользователю значимые для него сведения о предлагаемой продукции (технические характеристики, схемы применения, конкурентные преимущества). Кроме того, технические описания используются для того,

чтобы убедительно обосновать целесообразность использования данной продукции, побудить пользователя к личному обращению в компанию.

Технические описания продукции являются техническими документами в том понимании, в котором этот термин используется в *technical writing* [18]. **Technical writing** – это наука о создании технических документов, основная функция которых – удовлетворять потребность читателя в информации, требуемой ему для решения прикладных задач. К сожалению, до сих пор нет ни одной книги на русском языке, которая была бы посвящена *technical writing*. До сих пор нет устойчивого термина для обозначения этой области знаний, не считая жаргонного выражения "техническое писательство".

Типовое техническое описание продукции на сайте компании, которая занимается производством или внедрением технологически сложной продукции, состоит из следующих информационных блоков:

- дата публикации;
- заголовок (возможно, два варианта: краткий и полный);
- логотипы компаний-партнеров (например, производителя и поставщика);
- иллюстрация (возможно, несколько иллюстраций);
- резюме (основные характеристики в 1-2 предложениях);
- основной текст:
    - архитектура/конструкция (основные компоненты, блок-схемы);
    - возможности (функциональность, схемы/варианты применения);
    - порядок работы (сборка, настройка, управление, контроль);
- дополнительная информация (соответствие стандартам, сертификаты);
- информация для заказа: (комплект поставки, модификации).

Заголовок технического описания продукции обычно состоит из названия семейства (серии) продукции, к которому относится данная модель, и собственного названия данной модели (например, "Система залпового огня ГРАД"). В качестве иллюстраций могут использоваться фотографии оборудования и его основных узлов (на стенде и в реальных условиях), типовые

схемы применения или использования данной продукции, скриншоты, отражающие отличительные особенности программного обеспечения.

В техническом описании продукции нужно не только сообщить читателю достаточное количество информации, но и провести ее убедительный анализ, привести объективные свидетельства (факты, статистику, примеры, мнения независимых авторитетов), предложить альтернативы. Основные положения и отношения по возможности должны поясняться изображениями (графиками, диаграммами, картинками). Обычно имеет смысл выделить часть текста для специалистов и часть для широкой аудитории (как и в новости).

В качестве примера рассмотрим случай, когда на главной странице сайта выводится полный структурированный список всех типов и серий продукции и, кроме того, для удобства посетителей предусмотрено два дополнительных прототипа оглавлений (список продукции одного типа с разбиением по сериям и список продукции одной серии). Структурированные списки типов и серий продукции на главной странице и страницах оглавлений генерируются динамически на основе дополнительной таблицы TREE.

Типовой состав таблицы PRODUCTS в РСУБД корпоративного сайта приведен в прил. 4. Типовой состав таблицы TREE для двухуровневой классификации (главная страница → тип продукции → серия продукции → техническое описание продукции) приведен в прил. 5. Типовая схема связей таблицы PRODUCTS, прототипа технического описания продукции, прототипов оглавлений и главной страницы сайта приведена в прил. 8.

Отметим, что динамическое генерирование технического описания продукции в формате HTML позволяет легко получить множество ключевых слов этого HTML-документа простым объединением множества ключевых слов технического описания продукции, множества ключевых слов серии продукции и множества ключевых слов типа продукции, что обеспечивает лучшее представление документа в поисковых системах.

### 2.4.3. Портреты компаний-партнеров

Помимо двух основных типов документов – новостей (пресс-релизов) и технических описаний продукции – на корпоративном сайте обычно публикуются также различные дополнительные и вспомогательные документы: портреты компаний-партнеров, справки о реализованных проектах, описания отраслевых выставок, электронные версии публикаций в средствах массовой информации и т. д. Каждый из этих типов документов имеет свою специфику, их суммарная доля на корпоративном сайте чаще всего сравнительно невелика.

Рассмотрим в качестве примера портреты компаний-партнеров, основные функции которых– дать общее представление о компании, кратко описать ее историю, ключевые достижения, основные направления деятельности, текущее состояние и перспективы развития.

Типовой портрет компании-партнера состоит из следующих информационных блоков:

- название компании-партнера;
- логотип компании-партнера (возможно, два варианта разного размера);
- иллюстрация (возможно, несколько иллюстраций);
- резюме (общая характеристика компании в одном-двух предложениях);
- основной текст:
    - история, основные достижения, стратегия;
    - направления деятельности, семейства продукции;
    - анализ конкурентных преимуществ и перспектив;
    - форма партнерских отношений с данной компанией;
- ссылка на сайт компании;
- контактная информация.

Типовой состав таблицы VENDORS в РСУБД приведен в прил. 6. Типовая схема связей таблицы VENDORS, прототипа портрета компании-партнера и главной страницы сайта приведена в прил. 9.

## 2.5. Критерии выбора программного обеспечения

Рассмотрим критерии выбора программного обеспечения (ПО) для построения электронной информационной системы с использованием динамических документов на примере задачи выбора ПО для построения сайта. Под программным обеспечением сайта понимается сочетание четырех компонентов: операционной системы, веб-сервера, РСУБД и динамического движка (интерпретатора и других программных модулей), реализованного с использованием одного или нескольких языков программирования.

Общий критерий выбора ПО может быть сформулирован следующим образом: набор программного обеспечения должен обеспечивать максимальную независимость информационной политики предприятия от ценовой и лицензионной политики поставщиков программного обеспечения с целью обеспечить максимальную информационную безопасность и производительность системы [3]. Этот критерий, в частности, подразумевает, что каждый компонент ПО должен удовлетворять следующим требованиям:

- быть достаточно надежным, но при этом современным, гибким, многофункциональным, открытым к развитию;
- обеспечивать высокую степень безопасности, но в то же время хорошую переносимость на другие программно-аппаратные платформы;
- быть широко используемым, хорошо документированным, признанным в среде профессиональных разработчиков;
- обеспечивать в совокупности с другими компонентами минимизацию издержек на поддержку и модернизацию системы.

Как будет показано ниже, в настоящее время для решения рассматриваемой нами задачи данным критериям наиболее полно соответствует ПО с открытым исходным кодом [3].

**Операционная система (ОС).** В настоящее время в мире существует несколько десятков ОС, которые могут быть использованы как ОС веб-сервера: Microsoft Windows 95/98/2000/ME/NT/XP, Novell Netware 4/5/6, Sun Solaris и

т. д. Малое или среднее российское предприятие обычно выбирает между наиболее распространенными коммерческими ОС (Windows 2000/NT) и несколькими версиями некоммерческих ОС на основе UNIX (FreeBSD, Linux), в т. ч. русифицированными (Red Hat Linux Cyrillic Edition, AspLinux, AltLinux).

Существующие коммерческие операционные системы (Windows, Netware, Solaris), хотя и используются достаточно широко, обладают целым рядом недостатков, которые существенны для малых и средних предприятий. В первую очередь, это высокая цена лицензий, недостаточный уровень безопасности, большое число программных ошибок. Кроме того, использование коммерческих продуктов ставит информационную политику предприятия в зависимость от поставщика программного обеспечения.

Netware и Solaris в России в качестве операционной системы веб-сервера используются редко. Серверные ОС компании Microsoft распространены гораздо шире и являются одними из самых распространенных коммерческих серверных ОС. Основным недостатком Windows являются то, что это ОС с закрытым исходным кодом, имеющая проблемы с безопасностью. Однако простота использования и доступность нелицензионных дистрибутивов делают эту ОС привлекательной для российских предприятий.

ОС на основе UNIX заслуженно занимают первое место среди операционных систем для веб-серверов: на этих ОС работают более 50% всех веб-серверов в мире [174] и более 70% в России [176]. Это гибкие, надежные, бесплатные системы, однако, для их настройки требуется квалифицированный персонал. Выбор таких систем достаточно широк: это FreeBSD, OpenBSD, NetBSD и многочисленные версии Linux (Debian, RedHat, Mandrake, S.u.S.E., TurboLinux, Caldera, Slackware, RedHat Cyrillic Edition, Asplinux, Alt Linux).

В качестве ОС для веб-сервера FreeBSD/Linux имеют следующие преимущества перед Windows: они более устойчивы, более безопасны, менее требовательны к аппаратным ресурсам, более гибко конфигурируются, очень хорошо документированы, для них есть все необходимые приложения, они

распространяются бесплатно (лицензии BSD/GPL). Использование FreeBSD/Linux позволяет предприятию проводить более независимую информационную политику, чем использование Windows.

**Серверное ПО.** Лидером среди серверного ПО является Apache – ПО с открытым кодом, под управлением которого в настоящее время находится более 65% всех веб-серверов в мире [175] и более 85% в России [176]. Серверное ПО компании Microsoft значительно менее популярно: IIS используется менее чем на 25% веб-серверов в мире. Доля всего остального ПО для веб-серверов (свыше 60 разновидностей) в мире составляет менее 10%.

Apache содержит обширный API для расширения с помощью модулей, имеет большое количество подключаемых модулей (в т. ч. позволяющих реализовать функциональность IIS), работает на всех популярных ОС, имеет активную группу разработки и сообщество пользователей, полностью документирован, распространяется свободно [148].

**РСУБД.** Коммерческие РСУБД (Oracle, SQLServer, MS SQL и др.) по функциональности и масштабируемости пока превосходят РСУБД с открытым кодом. Это тщательно продуманные, отлаженные, профессиональные РСУБД; они традиционно ориентированы на биржи, банки и крупные корпорации, где необходима абсолютная надежность хранения и защиты данных. Цена лицензий, стоимость разработки и сопровождения коммерческих РСУБД достаточно высока, также как и стоимость подготовки персонала.

Из РСУБД с открытым кодом наиболее известны две: MySQL и PostgreSQL. MySQL [157], [158] свободно распространяется по условиям лицензии GPL (GNU Public License): пользователям платить не обязательно, но компаниям, которые хотят распространять это программное обеспечение на коммерческих условиях, не нарушая условия GPL, следует приобрести лицензию. PostgreSQL [165] распространяется на условиях, аналогичных лицензии BSD: каждый может свободно получить это программное обеспечение и использовать его, как пожелает.

MySQL и PostgreSQL пока не имеют таких сложных механизмов для обработки запросов, как, например, Oracle. Однако у них есть ряд преимуществ перед коммерческими РСУБД – это простота реализации, широкое сообщество пользователей и разработчиков, наличие подробной и свободно распространяемой документации, высокие темпы развития. Скорость и надежность MySQL и PostgreSQL достаточны для веб-приложений, и они очень популярны среди разработчиков.

**Язык программирования.** Для написания динамических движков сайтов в настоящее время применяются четыре основных языка (технологии) программирования: Perl, PHP, ASP и ColdFusion. ASP – продукт Microsoft, поставляемый бесплатно при условии наличия IIS; ColdFusion – коммерческий продукт компании Allaire (Macromedia). Perl и PHP распространяются свободно; PHP по функциональности практически соответствует ASP, Perl является мощным и гибким языком общего назначения.

В настоящее время практически все профессиональные разработчики, работающие на платформе на основе UNIX, используют встраиваемые в Apache-сервер модули Perl или PHP [162]. PHP изначально разрабатывался как скриптовый язык для веб-приложений; он легко встраивается в HTML-страницы, имеет простой и понятный синтаксис [163]. Perl – многоплановый язык программирования, который может применяться для решения самых разных задач, в частности, задач администрирования сервера [15], [160].

В настоящее время и Perl, и PHP активно развиваются, причем PHP постепенно приближается к Perl по своим возможностям. Однако в PHP до сих пор отсутствуют готовые системы построения прототипов (шаблонов), в то время как на языке Perl существует большое количество таких систем в свободном доступе [161]. В тех случаях, когда задача администрирования сервера решается на Perl, целесообразно использовать Perl и для написания динамического движка.

# 3. Генерирование DTD для массива XML-документов

В данной главе представлена технология записи массива XML-документов без использования информации об их структуре в РСУБД, и автоматического генерирования DTD для этого массива XML-документов. Эта технология является важным элементом системы автоматизированной подготовки динамических документов, так как позволяет автоматизировать создание таблиц РСУБД и их заполнение структурированной информацией. Технология записи XML-документов в РСУБД и генерирования DTD была разработана автором в сотрудничестве с Р. Р. Хуснутдиновым [4], [5] и реализована в виде экспериментальной системы в 2003-2004 гг. *В разделе 3.1* показано, что технология записи массива XML-документов в РСУБД и генерирования DTD для них является одним из ключевых элементов системы автоматизированной подготовки динамических документов. *В разделе 3.2* описаны известные методы записи XML-документов в РСУБД без использования информации об их структуре (такой, как DTD, XML Schema и т. п.). *В разделе 3.3* избранные методы развиты и модернизированы для решения задачи записи массива XML-документов в РСУБД за один проход. *В разделе 3.4* представлена технология генерирования DTD для массива XML-документов, развитая на основе методов и алгоритмов, предложенных в [57]. Наконец, *в разделе 3.5* описана архитектура системы записи массива XML-документов в РСУБД и генерирования DTD для них.

## 3.1. Автоматизация наполнения РСУБД

В процессе развития системы автоматизированной подготовки и публикации динамических документов неизбежно наступает момент, когда ручное занесение информации в базу данных становится неэффективным. Возникает задача автоматического наполнения РСУБД структурированной информацией, извлеченной из электронных документов и баз данных разных

форматов. При всей сложности этой задачи и ее комплексном характере, можно выделить ключевую подзадачу, от решения которой в значительной степени зависит общая эффективность автоматического наполнения РСУБД структурированной информацией. Это автоматическая запись массива XML-документов без использования информации об их структуре в РСУБД и генерирование DTD для этого массива XML-документов.

На сегодняшний день язык разметки XML [172] фактически является стандартом для хранения данных и обмена информацией в самых разных областях научной и коммерческой деятельности, и в обозримом будущем его популярность будет только расти. Сегодня доступны программные продукты для конвертирования электронных документов всех популярных форматов в XML-документы. Все ведущие производители РСУБД разрабатывают приложения, которые позволяют конвертировать XML-документы в таблицы базы данных и обратно: IBM [153], Microsoft [156], Oracle [159], Sybase [169]. Таким образом, задача автоматического наполнения РСУБД структурированной информацией, извлеченной из электронных документов и баз данных разных форматов, фактически сводится к задаче автоматического наполнения РСУБД структурированной информацией, извлеченной из массива XML-документов. Технически это эквивалентно построению набора таблиц РСУБД, наиболее точно соответствующих структуре XML-документов в массиве, и записи XML-документов в эти таблицы РСУБД. Эта задача имеет простое решение, если известна информация о структуре XML документа – DTD или XML Schema.

XML-документ может сопровождаться DTD, однако это условие не является обязательным (в отличие от SGML [168]). На сегодняшний день существует множество XML-документов, которые не имеют DTD: это документы, полученные путем конвертирования из HTML или других открытых форматов, а также конвертированные в XML из специализированных корпоративных систем хранения информации. Поскольку общее число XML-

документов пока еще сравнительно невелико (по отношению к общему объему электронных документов), очень вероятно, что большая часть XML-документов, которые появятся в ближайшем будущем, также будут получены путем автоматического конвертирования из хранилищ данных различных форматов. Скорее всего, в большинстве случаев эти массивы XML-документов не будут иметь соответствующих им DTD.

Таким образом, ключевым элементом системы автоматического наполнения РСУБД структурированной информацией является технология записи массива XML-документов, не имеющих соответствующего им DTD, в базу данных с возможностью их дальнейшей поэлементной обработки (в частности, выборки отдельных элементов согласно заданным критериям), и автоматического генерирования[12] DTD для этого массива XML-документов.

Задача автоматического генерирования DTD для массива XML-документов является далеко не такой простой, как может показаться на первый взгляд. До сих пор нет доступных программных продуктов, которые позволяли бы генерировать DTD для массива XML-документов с приемлемым качеством. Можно отметить несколько систем, которые содержат функцию автоматического или полуавтоматического построения DTD для XML-документа – DDbE [147], Allora [146], а также некоторые XML-редакторы – XML Spy [173], SAXON [167] и др. Однако, насколько можно судить по имеющейся информации, все эти системы реализуют достаточно простые и прямолинейные подходы для генерирования DTD и малоэффективны для решения практических задач. На данный момент в открытой печати подробно описана лишь одна достаточно развитая система, предназначенная для автоматического генерирования DTD: это система XTRACT, разработанная в

---

[12] В англоязычной литературе для обозначения процесса автоматического построения DTD используется целый ряд терминов - “creating”, “building”, “generating”, “extraction”, “mining”.

Bell Laboratories в рамках проекта SERENDIP [57]. Принцип работы этой системы основан на общем алгоритме построения DTD для SGML-документов, описанном в [58] (обобщение исходных последовательностей вложенных элементов, упрощение получившихся регулярных выражений и выбор наилучшего из них). Определенный интерес также представляют работы [59] и [60], где предлагаются методы построения приблизительно эквивалентных регулярных выражений для достаточно длинных строк символов. Общие вопросы выделения логической структуры для массива слабоструктурированных данных рассмотрены в [61], [62], [63], но предлагаемые в этих работах схемы данных существенно отличаются от DTD и не могут применяться для генерирования регулярных выражений.

## 3.2. Методы записи XML-документов в РСУБД

Рассмотрим основные методы записи XML-документов в РСУБД без использования информации об их структуре, описанные на сегодняшний день. Это Edge, Binary, Universal и Normalized Universal [53], [54] и Path [55].

Методы Edge, Binary, Universal и Normalized Universal [53], [54], основаны на следующих предположениях. Рассматривается XML-документ, который не содержит элементов со смешанным содержимым, то есть состоит только из элементов с содержимым из элементов и элементов с содержимым типа `#PCDATA`[13]. Все атрибуты рассматриваются как дочерние элементы. Считается, что все элементы пронумерованы – для простоты предполагается, что отдельно пронумерованы элементы с содержимым из элементов (1…N), и отдельно - элементы (и атрибуты) с содержимым типа `#PCDATA` (или, что эквивалентно, сами секции `#PCDATA`) (v1…vM). XML-документу ставится в соответствие ориентированный ациклический граф, при этом элементам с

[13] Строка символов, не обрабатываемая интерпретатором XML – текст и т. п.

содержимым из элементов соответствуют внутренние вершины графа, секциям #PCDATA – висячие вершины (листья) графа, отношениям "родительский элемент – дочерний элемент" – ребра графа. Ребра, исходящие из одной вершины, нумеруются согласно порядку следования дочерних элементов. Каждое ребро поименовано: имя ребра совпадает с именем дочернего элемента. Пример XML документа и его графа согласно [53] приведен в прил. 10 (левый граф).

Отметим, что между XML-документом и ориентированным графом, который построен согласно правилам, предложенным в [53], нет взаимно однозначного соответствия. Во-первых, в них не вводится различия между атрибутами и дочерними элементами с содержимым типа #PCDATA: и тем, и другим соответствуют листья графа. Во-вторых, отношения между элементами, задаваемые системой ссылок (т. е. атрибутов типа ID и IDREF), отображаются на графе точно так же, как и обычные отношения "родитель - потомок": и тем, и другим соответствуют ребра графа (при этом сами атрибуты типа ID и IDREF на графе не отображаются). Таким образом, при построении графа по описанным выше правилам часть информации о структуре XML-документа теряется, и исходный документ не может быть точно восстановлен по графу.

**Метод Edge.** Простейшая реляционная схема состоит в том, что информация о всех ребрах графа, соответствующего XML-документу, хранится в одной таблице. Этот метод получил название Edge. Каждому ребру графа соответствует строка в таблице *Edge*, в которой записаны: уникальный номер "родителя" (вершины, из которого исходит ребро) – *source*, порядковый номер ребра (среди всех ребер, исходящих из одной вершины) – *ordinal*, имя ребра (т. е. имя дочернего элемента) – *name*, маркер типа связи (показывающий, связывает ли ребро две внутренних вершины графа, или же оно указывает на лист графа, т. е. на значение #PCDATA) – *flag*, уникальный номер "потомка" (внутренней вершины или листа, на который указывает ребро) – *target*. Таким образом, таблица *Edge* имеет следующую структуру: *Edge*

*(source, ordinal, name, flag, target)*. Поле *flag* может содержать не просто булевскую переменную (*"ссылка"/"значение"*), а информацию о типе конкретного значения (например, *"ссылка"/"число"/"строка"*). Пример таблицы *Edge* для рассматриваемого XML-документа (для случая, когда значения хранятся в отдельных таблицах) приведен в прил. 11.

**Метод Binary** (также часто называемый Attribute) заключается в том, что все ребра с одинаковыми именами хранятся в одной таблице. Этот метод основан на схеме хранения данных, предложенной в [56]. По сути, метод Binary соответствует горизонтальной декомпозиции таблицы *Edge* по именам ребер. Таким образом, создается столько таблиц, сколько различных имен элементов и атрибутов встречается в XML-документе. Каждая таблица имеет следующую структуру: $Binary_{name}$ *(source, ordinal, flag, target)*. Все поля в таблицах *Binary* имеют то же значение, что и в таблице *Edge*.

**Метод Universal.** В методе Universal информация о всех ребрах графа хранится в одной таблице, которая имеет следующую структуру (считая, что $name_1, \ldots, name_k$ – все различные имена элементов и атрибутов): *Universal* (*source, ordinal* ($name_1$), *flag* ($name_1$), *target* ($name_1$), *ordinal* ($name_2$), *flag* ($name_2$), *target* ($name_2$),…, *ordinal* ($name_k$), *flag* ($name_k$), *target* ($name_k$)). По существу, таблица *Universal* является полным внешним объединением всех таблиц *Binary*. Пример таблицы *Universal* для рассматриваемого XML-документа (для случая, когда значения хранятся в отдельных таблицах) приведен в прил. 12.

**Метод Normalized Universal** отличается от метода Universal тем, что те элементы и атрибуты родительского элемента, которые имеют одинаковые имена, но принимают разные значения, хранятся не в общей таблице *UnivNorm*, а в отдельных таблицах $Overflow_{name\_1}, \ldots, Overflow_{name\_k}$. Таблицы *Overflow* имеют следующую структуру: $Overflow_{name\_1}$ *(source, ordinal, flag, target);… ;* $Overflow_{name\_k}$ *(source,*

*ordinal, flag, target).* Если элемент (атрибут) с данным именем принимает только одно значение, поле *flag* в таблице *UnivNorm* используется точно таким же образом, что и в предыдущих методах, если этих значений несколько – полю *flag* присваивается значение *"список"*.

**Хранение значений.** В каждом из описанных выше методов – Edge, Binary, Universal и Normalized Universal – может быть применено два варианта хренения значений типа #PCDATA. Первый вариант – хранить значения в отдельных таблицах значений. В этом случае создается столько таблиц значений, сколько различных типов данных представлено в рассматриваемом XML документе. Например, это могут быть таблицы $V_{int}$ *(vid, value)*, $V_{string}$ *(vid, value)* и $V_{date}$ *(vid, value)*. Поле *flag* в этом случае может принимать значения *"число", "строка", "дата"* или *"ссылка"* (а также *"список"* для метода Normalized Universal). Второй вариант – хранить значения в той же таблице, в которой хранится информация о ребрах графа (*inlining – вложение*). Во этом случае в каждой таблице создается столько дополнительных колонок, сколько типов данных представлено в XML-документе. Очевидно, что поле *flag* в этом случае не нужно.

**Метод Path** [55] заключается в том, что в качестве уникального идентификатора каждого элемента (атрибута) выступает строка *"path"*, представляющая полный путь от корневого элемента к данному. Строка *"path"* представляет собой простое объединение имен всех элементов, являющихся родителями данного. Разделителем служит точка, имя корневого элемента идет первым, имя данного элемента - последним. Поскольку элементы с одинаковым именем могут встречаться в XML-документе несколько раз, к каждому имени в конце добавляется его порядковый номер. Например, в рассматриваемом нами XML-документе для элемента <name>Ольга</name> строка *"path"* будет выглядеть так: *".person0.child1.name0"*.

В методе Path для XML-документа строятся две таблицы – *Element* и *Attribute*. Таблица *Element* имеет следующую структуру: *Element (path, value, parent)*. Поле *path* содержит строку *"path"* для данного элемента. Поле *parent* содержит строку *"path"* для элемента, родительского по отношению к данному. Поле *value* содержит данные типа #CDATA или #PCDATA (если данных такого типа в элементе нет, полю *value* присваивается значение *"null"*). Таблица *Attribute* имеет следующую структуру: *Attribute (path, name, value)*. Поле *path* содержит строку *"path"* для элемента, к которому относится данный атрибут. Поле *name* содержит имя атрибута, поле *value* – значение атрибута. Отметим, что атрибуты типа ID и IDREF обрабатываются точно так же, как остальные атрибуты. Примеры таблиц *Element* и *Attribute* для рассматриваемого XML-документа приведены в прил. 13.

## 3.3. Модернизация реляционных схем

В разделе 3.2 было описано девять основных методов записи XML-документов в РСУБД. Их основные характеристики сведены в таблице 3.1.

| **Название** | **Вложение значений** | **Число таблиц** | **Возможность записи за один проход** |
|---|---|---|---|
| Edge | нет | 2 | есть |
| Edge Inline | есть | 1 | есть |
| Binary | нет | ≥ 2 | есть |
| Binary Inline | есть | ≥ 1 | есть |
| Universal | нет | 2 | нет |
| Universal Inline | есть | 1 | нет |
| Normalized Universal | нет | ≥ 2 | нет |
| Normalized Universal Inline | есть | ≥ 1 | нет |
| Path | нет | 2 | есть |

*Табл. 3.1. Сравнительные характеристики различных реляционных схем для записи XML-документов в РСУБД.*

Основным критерием при отборе реляционных схем для дальнейшей доработки и модернизации для нас будет возможность записи XML-документа в РСУБД за один проход. Этому условию удовлетворяют методы Edge, Edge

Inline и Path (которые позволяют сначала создать все таблицы базы данных, а затем заполнять их по мере обработки XML-документа), а также методы Binary и Binary Inline (которые позволяют создавать и заполнять таблицы базы данных параллельно с обработкой XML-документа).

Методы семейства Universal не поддерживают запись за один проход, что делает затруднительным их применение для обработки больших массивов XML-документов. Основные таблицы в этих методах состоят из всех элементов документа, соответственно, они могут быть построены только после обработки всего XML-документа. Кроме того, эти методы обладают большой избыточностью, и в случае больших XML-документов таблицы базы данных будут содержать большое число повторов и пустых полей.

Таким образом, исходя из поставленных целей, для развития мы выбираем 5 методов – Edge, Edge Inline, Path, Binary и Binary Inline [4].

**Построение графа.** Как было указано выше, правила построения графа XML документа, предложенные в [53], обладают рядом недостатков. Во-первых, они неприменимы для XML-документов, которые содержат элементы со смешанным содержимым (т. е. с содержимым из элементов, атрибутов и секций `#PCDATA`). Во-вторых, они не предусматривают различения вложенных элементов и атрибутов родительского элемента (что важно, например, для генерирования DTD документа). В-третьих, они не рассчитаны на обработку массивов XML-документов. Чтобы устранить эти недостатки, мы модернизируем правила построения графа, предложенные в [53].

Модернизированный граф строится следующим образом. Каждому элементу документа ставится в соответствие внутренняя вершина графа, каждому атрибуту и секции `#PCDATA` – висячая вершина (лист) графа. Все ребра графа нумеруются последовательно, начиная с 1 (по правилу обхода "элемент – потомки элемента по порядку – потомки первого из потомков элемента и т. д."). Все элементы также нумеруются последовательно, начиная с 1, по аналогичному правилу. Виртуальному родителю элемента (элементов)

верхнего уровня присваивается номер 0. Модернизированный граф для рассматриваемого XML-документа приведен в прил. 10 (числа меньшего размера – номера ребер, числа большего размера – номера элементов).

**Метод Edge Distributive** является модификацией метода Edge, описанного выше. Информация о всех ребрах графа – связывающих элемент с элементом, атрибутом или секцией #PCDATA – хранится в одной таблице *Edge*, которая имеет следующую структуру: *Edge (id, document_id, parent_id, element_id, order_num, name)*. Поле *id* содержит порядковый номер ребра графа. Поле *document_id* содержит уникальный номер XML-документа в массиве XML-документов (или уникальное имя XML-документа). Поле *order_num* содержит порядковый номер ребра среди всех ребер, исходящих из одной вершины.

Содержимое полей *parent_id*, *element_id* и *name* зависит от того, указывает ли ребро на элемент, атрибут или секцию #PCDATA. Если ребро указывает на элемент, то поле *parent_id* содержит значение *"element_id"* родительского элемента, поле *element_id* – значение *"element_id"* данного элемента, поле *name* – имя данного элемента. Если ребро указывает на атрибут, то поле *parent_id* содержит значение *"element_id"* элемента, которому принадлежит данный атрибут, поле *element_id* – значение *"null"*, поле *name* – имя данного атрибута. Если ребро указывает на секцию #PCDATA, то поле *parent_id* содержит значение *"element_id"* элемента, которому принадлежит данная секция #PCDATA, поля *element_id* и *name* – значения *"null"*.

Правила построения таблицы Edge в методе Edge Distributive, описанные выше, сведены в таблице 3.2.

| | id | document_id | parent_id | element_id | order _num | name |
|---|---|---|---|---|---|---|
| **Элемент** | уникальный id ≥ 1 | уникальное имя XML-документа | "element_id" родительского элемента | "element_id" данного элемента (≥ 1) | порядковый номер ребра | имя элемента |
| **Атрибут** | уникальный id > 1 | уникальное имя XML-документа | "element_id" элемента | null | порядковый номер ребра | имя атрибута |
| **Секция #PCDATA** | уникальный id > 1 | уникальное имя XML-документа | "element_id" элемента | null | порядковый номер ребра | null |

*Табл. 3.2. Правила построения таблицы Edge в методе Edge Distributive.*

Значения атрибутов и секций #PCDATA в методе Edge Distributive хранятся в отдельной таблице *Value*, которая имеет следующую структуру: *Value (id, data)*. Поле *id* содержит уникальный номер *"id"* элемента, атрибута или секции #PCDATA, поле *data* – значение атрибута или секции #PCDATA и значение *"null"* для элементов. Правила построения таблицы *Value* в методе Edge Distributive сведены в таблице 3.3.

| | id | data |
|---|---|---|
| **Элемент** | уникальный id ≥ 1 | null |
| **Атрибут** | уникальный id > 1 | значение атрибута |
| **Секция #PCDATA** | уникальный id > 1 | содержимое секции #PCDATA |

*Табл. 3.3. Правила построения таблицы Value в методе Edge Distributive.*

Примеры таблиц *Edge* и *Value* для рассматриваемого XML-документа приведены в прил. 14.

**Метод Edge Inline** отличается от Edge Distributive только тем, что в нем значения атрибутов и секций #PCDATA хранятся в одной таблице с информацией о ребрах графа. Другими словами, таблицы *Edge* и *Value* объединены в единую таблицу *Edge*, которая имеет следующую структуру: *Edge (id, document_id, parent_id, element_id, order_num, name, data)*. Значения всех полей в методе Edge Inline точно такие же, как и в методе Edge Distributive.

**Метод Binary Distributive** отличается от метода Edge Distributive тем, что информация о ребрах, указывающих на элементы (атрибуты) с одинаковыми именами, хранится в отдельных таблицах. Технически это соответствует горизонтальной декомпозиции таблицы *Edge*, используемой в методе Edge Distributive, по полю *name*. Таким образом, в методе Binary Distributive создается столько таблиц, сколько разных имен элементов (атрибутов) встречается в XML-документе, плюс три дополнительные таблицы, которые будут описаны ниже.

Каждая таблица, полученная в результате декомпозиции таблицы *Edge Distributive* по полю *name*, имеет следующую структуру: *Tag_i (id, document_id, parent_id, element_id, order_num)*. Значения всех полей и политика индексирования таблиц *Tag_1, Tag_2,…, Tag_i* в методе Binary Distributive точно такие же, как и в методе Edge Distributive. Индексы *{id}* и *{element_id}* имеют прозрачную нумерацию по всем таблицам *Tag_i*, соответственно, практически все запросы для методов типа Binary будут иметь вид: *(select clause from Tag_1) UNION … UNION (select clause from Tag_i)*.

Чтобы установить взаимно однозначное соответствие между названиями *"Tag_i"* и реальными именами элементов и атрибутов, создается таблица *Tag_list*, которая имеет следующую структуру: *Tag_list (tag_name, table_name)*. В поле *tag_name* хранится реальное имя элемента или атрибута, в поле *table_name* – название таблицы *"Tag_i"*.

Поскольку секции #PCDATA в наших методах типа Edge обладают значением поля *name* равным *"null"*, для хранения информации о ребрах, указывающих на секции #PCDATA, необходимо создать дополнительную таблицу *Tag_0* с той же структурой, что и у остальных таблиц *Tag_i*.

Наконец, в отдельной таблице *Value* сохраняются значения атрибутов и секций #PCDATA. Таблица *Value* имеет точно такую же структуру, как и в методе Edge Distributive: *Value (id, data).*

Примеры таблиц *Tag_list, Tag_5 (child), Tag_0* и *Value* для рассматриваемого XML-документа приведены в прил. 15.

**Метод Binary Inline** отличается от метода Binary Distributive тем, что значения атрибутов и секций #PCDATA хранятся не в отдельной таблице *Value*, а в тех же таблицах *Tag_i*, в которых содержится информация о ребрах графа. Таблицы *Tag_i* в методе Binary Inline имеют следующую структуру: *Tag_i (id, document_id, parent_id, element_id, order_num, data).*

**Метод Path,** предложенный в [55], не поддерживает возможность обработки массивов XML-документов. Кроме того, используемая в нем схема генерирования строк *"path"* не будет работать корректно в том случае, когда имя элемента заканчивается на число (например, *room1*) или в имени элемента содержится комбинация числа и точки (например, *book1.chapter3*), вероятность чего для реальных XML-документов достаточно высока. Соответственно, мы модернизируем метод Path таким образом, чтобы устранить указанные недостатки. Во-первых, в таблице *Element* создается дополнительное поле – *element_id*. Во-вторых, нумерацию элементов-потомков с одинаковыми именами мы осуществляем в формате *"_#i"*, где *i* – порядковый номер потомка. Значение поля *parent_path* для элемента или элементов верхнего уровня – по определению *"."*. Таким образом, в модернизированном методе Path строится две таблицы: *Element (document_id, path, data, parent_path)* и *Attribute (path, name, data).* Примеры таблиц *Element* и *Attribute* для рассматриваемого XML-документа приведены в прил. 16.

## 3.4. Технология генерирования DTD

Технология автоматического генерирования DTD для массива XML-документов, представленная в данном разделе [5], развита на основе алгоритмов генерирования DTD, предложенных в [57].

Основную сложность в задаче генерирования DTD для массива XML-документов представляет задача построения DTD для элемента с содержимым из элементов. Действительно, DTD документа представляет собой простое объединение DTD отдельных элементов и атрибутов, каждый из которых никак не зависит от DTD других элементов и атрибутов. Построение DTD для атрибутов и элементов с содержимым типа `#PCDATA` осуществляется относительно просто. Таким образом, далее мы будем рассматривать задачу построения DTD для элемента с содержимым из элементов.

Более строго эту задачу можно сформулировать следующим образом. Пусть элемент $X$ встречается в XML-документе (или массиве XML-документов) $n$ раз, которым соответствуют $n$ последовательностей вложенных в него элементов $s_1, s_2, \ldots s_n$. Задача состоит в том, чтобы для множества последовательностей $I = \{s_1, s_2 \ldots s_n\}$ построить регулярное выражение (DTD элемента $X$), описывающее все эти последовательности.

Регулярное выражение – это способ кодировки множества последовательностей символов. Синтаксис регулярных выражений основан на использовании метасимволов **?**, **+**, ***** для обозначения возможного числа повторений символа ("0 или 1", "1 или больше", "0, 1 или больше" соответственно), метасимвола **|** для обозначения дизъюнкции, а также метасимволов **(** и **)** для выделения групп символов. Например, регулярное выражение ***(ab)+(c|d)*** кодирует множество последовательностей ***{abc, abd, ababc, ababd, abababc, abababd …}***. Математически строгое определение регулярного выражения можно найти, например, в [13].

Задачу построения DTD для элемента с содержимым из элементов можно разбить на три этапа: обобщение исходных последовательностей, факторизация полученных регулярных выражений и построение наилучшего DTD на основе полученного множества регулярных выражений.

### 3.4.2. Архитектура системы генерирования DTD

Архитектура системы генерирования DTD представлена в прил. 17. Система состоит из трех модулей: модуля обобщения, модуля факторизации и модуля MDL. Входными данными для модуля обобщения служит множество последовательностей ***I***. Результатом работы модуля обобщения является множество выражений $S_G$, которое включает регулярные выражения, полученные в результате обобщения последовательностей из ***I***, а также все исходные последовательности из ***I***. Множество $S_G$ служит входными данными для модуля факторизации. Результатом работы модуля факторизации является множество выражений $S_F$, который включает регулярные выражения, полученные в результате факторизации выражений из множества $S_G$, а также все выражения из множества $S_G$. Наконец, множество $S_F$ служит входными данными для модуля MDL, который выбирает из $S_F$ подмножество выражений ***S***, которое покрывает все исходные последовательности из множества ***I*** и MDL-стоимость которого минимальна. Итоговый DTD является дизъюнкцией (логическим ИЛИ) всех выражений из ***S***.

Модуль обобщения обрабатывает все последовательности из ***I***, генерируя для каждой из них (при возможности) одно или несколько регулярных выражений с использованием метасимволов ******* и **|**. Например, для ***I = {abab, bbbe}*** модуль обобщения построит регулярные выражения ***(ab)****, ***(a|b)**** и ***b*e***.

Модуль факторизации обрабатывает все выражения из множества $S_G$, генерируя на основе нескольких выражений из $S_G$ (при возможности) новые регулярные выражения с использованием факторизации (т. е. вынесения за

скобки повторяющихся последовательностей символов). Для этого используются соответствующим образом адаптированные алгоритмы факторизации из [64], [75]. Например, факторизация выражений ***b*d*** и ***b*e*** даст ***b*(d|e)***, а выражения ***ac***, ***ad***, ***bc***, ***bd*** будут факторизованы в ***(a|b)(c|d)***. Этап факторизации важен, так как позволяет получить более короткие выражения, которые, возможно, окажутся более предпочтительными на этапе построения наилучшего DTD.

Модуль MDL выбирает из множества выражений $S_F$, полученного в результате работы модулей обобщения и факторизации, подмножество выражений ***S***, которое покрывает все последовательности из набора ***I*** и MDL-стоимость которого минимальна. Для этого используются соответствующим образом адаптированные алгоритмы из [65], [66]. Итоговый DTD представляет собой дизъюнкцию (логическое ИЛИ) всех выражений подмножества ***S***.

### 3.4.3. Модуль обобщения

Алгоритмы обобщения представлены в прил. 19. Процедура GENERALIZE генерирует несколько обобщенных регулярных выражений для каждой последовательности из ***I*** и добавляет их в множество $S_G$ (которое изначально совпадает с ***I***). Для этого последовательно вызываются процедуры DISCOVERSEQPATTERN и DISCOVERORPATTERN с различными параметрами.

**Генерирование выражений вида *(x)*.*** Процедура DISCOVERSEQPATTERN получает на входе последовательность ***s*** и параметр ***r***>***1***. В том случае, если последовательность ***s*** содержит хотя бы одну подпоследовательность вида ***xx…x*** (где ***x*** – один символ или последовательность символов) с числом повторений не менее ***r***, процедура DISCOVERSEQPATTERN возвращает регулярное выражение, которое получено из последовательности ***s*** путем замены этой подпоследовательности на регулярное выражение ***(x)****. Если таких подпоследовательностей несколько, выбирается наиболее длинная из них и процедура повторяется снова.

Процедура DiscoverSeqPattern вызывается из процедуры Generalize с тремя различными значениями параметра ***r***, что позволяет генерировать регулярные выражения с различной степенью обобщения. Например, для последовательности ***aabbb*** процедура DiscoverSeqPattern с ***r=2*** даст выражение ***a*b****, а с ***r=3*** – выражение ***aab****. На этапе выбора по принципу MDL более предпочтительным может оказаться как то, так и другое выражение – в зависимости от того, какое точнее описывает последовательности из ***I***.

**Генерирование выражений вида $(a_1|a_2|\ldots|a_m)^*$.** Процедура DiscoverOrPattern заменят локальные скопления символов $a_1$, $a_2$, …, $a_m$ в последовательности ***s*** на регулярные выражения вида $(a_1|a_2|\ldots|a_m)^*$. Идея состоит в том, что если в последовательности ***s*** есть подпоследовательность, которая представляет собой частое повторение символов из набора $\{a_1, a_2, \ldots, a_m\}$, то эта подпоследовательность с большой вероятностью описывается регулярным выражением вида $(a_1|a_2|\ldots|a_m)^*$.

Для выявления таких локальных скоплений используется процедура Partition. Эта процедура разбивает последовательность ***s*** на подпоследовательности $s_1$, $s_2$, …, $s_n$ таким образом, чтобы расстояние между одинаковыми символами в соседних подпоследовательностях оказалось не меньше ***d*** (где ***d*** – входной параметр процедуры DiscoverOrPattern), рис. 3.1. Процедура DiscoverOrPattern затем просто заменят каждую подпоследовательность $s_i$ на регулярное выражения вида $(a_1|a_2|\ldots|a_m)^*$, где $a_1$, $a_2$, …, $a_m$ – различные символы в подпоследовательности $s_i$.

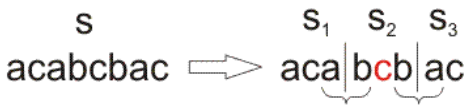



*Рис. 3.1. Принцип работы процедуры Partition.*

Опишем более подробно процедуру Partition. Введем следующие обозначения. Для произвольной последовательности ***s***, выражение ***s[i,j]***

означает подпоследовательность $s$, которая начинается $i$-м символом $s$ и заканчивается $j$-м символом $s$. Процедура PARTITION строит подпоследовательности $s$ по порядку: $s_1$, $s_2$ и так далее. Предположим, построены подпоследовательности $s_1$ … $s_j$. Тогда подпоследовательность $s_{j+1}$ начинается с символа, следующего за последним символом $s_j$, и продлевается вправо до тех пор, пока не будет выполнено условие: ни один символ из $s_{j+1}$ не встречается справа от $s_{j+1}$ на расстоянии от этого символа, меньше или равном $d$. По построению, такой символ не может встретиться слева от $s_{j+1}$.

Отметим, что на вход процедуры DISCOVERORPATTERN поступают выражения, которые получились в результате выполнения процедуры DISCOVERSEQPATTERN. Это позволяет строить регулярные выражения более сложной формы, чем при независимом выполнении этих процедур или при их выполнении в обратном порядке. Например, для входной последовательности $s$=**abcbca** процедура DISCOVERSEQPATTERN с параметром $r$=2 даст $s'$=$aA_1a$, где $A_1$=(**bc**)*, а обработка последовательности $s'$ процедурой DISCOVERORPATTERN с параметром $d$=|$s'$| (число символов в $s'$) даст $s''$=$A_2$, где $A_2$=(**a**|$A_1$)*=(**a**|(**bc**)*)*. Выполнение же процедуры DISCOVERORPATTERN сразу для последовательности $s$ с параметром $d$=|$s$| дало бы более простое выражение (**a**|**b**|**c**|)*. Отметим также, что процедура DISCOVERORPATTERN вызывается с различными значениями параметра $d$, которые выражаются в долях от длины входной последовательности. Это позволяет генерировать регулярные выражения с разной степенью обобщения.

### 3.4.4. Модуль факторизации

На вход модуля факторизации поступает множество регулярных выражений $S_G$, полученное в результате работы модуля обобщения. Модуль факторизации добавляет к этому множеству новые выражения, полученные в результате факторизации (вынесения за скобки повторяющихся последовательностей символов) двух или более выражений из множества $S_G$.

Поскольку факторизованные выражения короче, чем простая дизъюнкция соответствующих выражений из множества $S_G$, они могут оказаться более предпочтительными на этапе построения наилучшего DTD.

Модуль факторизации состоит из двух суб-модулей: первый из них выбирает "перспективные" для факторизации подмножества из множества $S_G$, а второй строит факторизованные формы для этих подмножеств.

**Выбор подмножеств множества $S_G$ для факторизации.** Алгоритм выбора "перспективных" для факторизации подмножеств множества $S_G$ основан на двух идеях. С одной стороны, подмножество $S$ множества $S_G$ является хорошим кандидатом для факторизации, если факторизованная форма выражений из $S$ существенно короче, чем простая дизъюнкция этих выражений. Например, подмножество $S_1 = \{abcd, abce\}$ более перспективно для факторизации, чем $S_2 = \{abcd, aefg\}$, так как факторизация $S_1$ даст регулярное выражение $abc(d|e)$, а факторизация $S_2$ – $a(bcd|efg)$.

С другой стороны, регулярные выражения тем более перспективны для факторизации, чем меньше пересечение между множествами последовательностей из исходного множества $I$, которые описываются этими выражениями. Например, факторизация регулярных выражений $b^*$ и $ab^*$, которые описывают непересекающиеся множества последовательностей из $I$, с большей вероятностью позволит улучшить финальный DTD, чем факторизация регулярных выражений $a^*b^*$ и $aab^*$, полученных в результате обобщения одной и той же входной последовательности из $I$.

Таким образом, "перспективное" для факторизации подмножество $S$ множества $S_G$ должно удовлетворять двум требованиям. Во-первых, каждое выражение в $S$ должно иметь общий префикс / суффикс с несколькими другими выражениями в $S$. При этом, чем больше выражений в $S$ имеют общие префиксы / суффиксы и чем больше длина общих префиксов / суффиксов, тем лучшего результата можно ожидать от факторизации подмножества $S$. Во-

вторых, пересечение между подмножествами последовательностей из ***I***, которые описываются различными регулярными выражениями из ***S***, должно быть достаточно мало.

Чтобы строго определить второй критерий, введем обозначения:

- ***покрытие(D)*** – подмножество последовательностей из множества ***I***, которые описываются регулярным выражением ***D***;
- ***перекрытие(D,D')*** = ***|покрытие(D) ∩ покрытие(D')| / |покрытие(D) U покрытие(D')|*** - отношение числа последовательностей, описываемых одновременно обеими выражениями ***D*** и ***D'***, к общему числу последовательностей, описываемых выражениями ***D*** и ***D'***.

Строгая формулировка второго критерия выглядит следующим образом: подмножества ***S*** должны строится так, чтобы для любой пары ***D*** и ***D'*** из подмножества ***S*** ***перекрытие(D,D')*** < ***δ***, где ***δ*** – достаточно малая величина, заданная пользователем.

Чтобы строго определить первый критерий, введем обозначения:

- ***pref(D)*** – множество префиксов ***D*** (т. е., выражений, полученных отсечением правой части регулярного выражения ***D***);
- ***suf(D)*** – множество суффиксов ***D*** (т. е., выражений, полученных отсечением левой части регулярного выражения ***D***);
- ***psup(p,S)*** – число выражений в **S**, для которых ***p*** – префикс;
- ***ssup(s,S)*** – число выражений в ***S***, для которых ***s*** – суффикс;
- ***рейтинг(D,S)*** = ***max [{|p|*psup(p,S): p*** принадлежит ***pref(D)} U {|s|*ssup(s,S): s*** принадлежит ***suf(D)}].***

Функция ***рейтинг(D,S)*** - это просто максимальное произведение длины префикса / суффикса выражения ***D*** на число выражений в ***S*** с таким же префиксом / суффиксом. Функция ***рейтинг(D,S)*** вводится для того, чтобы в

численном виде выразить степень схожести между префиксами/суффиксами выражения ***D*** и префиксами / суффиксами остальных выражений **S**.

Строгая формулировка первого критерия выглядит следующим образом: подмножества ***S*** должны строится так, чтобы для каждого ***D*** из ***S*** значения ***рейтинг(D,S)*** и ***рейтинг(D,*** $S_G$***)*** были как можно больше.

Выбор подмножеств ***S*** из множества $S_G$, удовлетворяющих обоим критериям, осуществляется процедурой FactorSubsets, прил. 18. Эта процедура выбирает ***k*** таких подмножеств (где ***k*** – параметр, задаваемый пользователем), каждое из которых затем подвергается факторизации путем вызова процедуры Factor. Процедура Factor возвращает дизъюнкцию вида $F_1 \mid F_2 \mid \ldots \mid F_m$ для каждого подмножества ***S***. Каждое из выражений $F_i$ добавляется к $S_F$.

Процедура FactorSubsets работает следующим образом. Сначала (шаги 4-7) из множества $S_G$ выбирается подмножество ***SeedSet*** из ***k*** выражений, которые имеют наибольшие ***рейтинги*** по отношению к $S_G$ и минимальные ***перекрытия*** друг с другом. Затем (шаги 9-14) каждое из этих выражений используется для построения подмножества ***S*** для факторизации (т. е. генерируется ***k*** подмножеств). Это построение осуществляется итерационно: на первом шаге ***S*** состоит только из начального выражения, каждое следующее выражение добавляется в ***S*** из $S_G$ по правилу: ***рейтинг*** выражения по отношению к ***S*** максимален и его ***перекрытие*** с выражениями из ***S*** меньше $\delta$.

**Факторизация подмножества регулярных выражений.** Алгоритмы факторизации множества регулярных выражений представляют собой соответствующим образом адаптированные алгоритмы из [75]. Процедура факторизации представлена в прил. 20. Ее ключевым элементом является процедура Divide(S,V), которая делит множество ***S*** на множество ***V*** и возвращает частное ***Q*** и остаток ***R***. Деление в данном случае означает поиск таких множеств ***Q*** и ***R***, что $S = V \circ Q \cup R$, где $V \circ Q$ – это множество выражений, полученных в результате присоединения каждого выражения из ***Q***

справа к каждому выражению из ***V***. Например, для множества ***S = {b***, ***c***, ***ab***, ***ac***, ***df***, ***dg***, ***ef***, ***eg}*** и делителя ***V = {d***, ***e}*** процедура DIVIDE(S,V) вернет частное ***Q = {f***, ***g}*** и остаток ***R ={b***, ***c***, ***ab***, ***ac}***.

### 3.4.5. Модуль MDL

Результатом работы модуля факторизации является множество регулярных выражений $S_F$. Напомним, что $S_F$ – это объединение множества последовательностей из ***I***, множества выражений $S_G$, полученных путем обобщения последовательностей из ***I***, и множества выражений, полученных путем факторизации двух и более выражений из $S_G$. Модуль MDL осуществляет выбор такого подмножества выражений ***S*** из множества $S_F$, что финальный DTD (дизъюнкция выражений из ***S***) покрывает все последовательности из исходного множества ***I*** и при этом его MDL-стоимость минимальна. MDL-стоимость DTD складывается из числа битов, необходимых для того, чтобы закодировать данный DTD – ***A***, и числа битов, необходимых для того, чтобы закодировать все последовательности из ***I*** на основе данного DTD – ***B***. Сначала мы опишем схему кодирования, которая применяется для вычисления значений ***A*** и ***B***, а затем – алгоритм выбора подмножества ***S*** из множества $S_F$.

**Схема кодирования.** Пусть ***Σ*** – множество символов, которые встречаются в последовательностях набора ***I***. Пусть ***M*** – множество метасимволов **|**, *******, ***+***, ***?***, ***)***, ***(***. Тогда DTD будет строкой из элементов множества ***Σ U M***. Пусть длина DTD – ***n***. Тогда число битов, необходимых для кодирования данного DTD, вычисляется по формуле: $n*]\log_2(|\Sigma \cup M|)[$. Здесь $|\Sigma \cup M|$ - число элементов в ***Σ U M***, $]x[$ – ближайшее к ***x*** целое число, большее или равное ***x***.

Смысл этой формулы очевиден. Чтобы закодировать $|\Sigma \cup M|$ различных элементов в двоичной форме, необходимо минимум $]\log_2(|\Sigma \cup M|)[$ двоичных разрядов. Два элемента можно закодировать с использованием одного разряда (***0*** и ***1***), три или четыре элемента – с использованием двух

разрядов (***00***, ***01***, ***10***, ***11***) и т. д. Соответственно, длина DTD в двоичной кодировке вычисляется как произведение числа элементов DTD на число разрядов, необходимых для кодирования каждого элемента.

Пусть, например, ***Σ={a***, ***b}***. Тогда длина DTD ***a*b**** в битах будет 4*]$\log_2$(2+6)[ = 4*3 = 12, длина DTD ***(ab|abb)(aa|ab*)*** – 16*3 = 48, и т. д.

Число битов ***B***, необходимых для того, чтобы закодировать исходный набор последовательностей ***I*** на основе данного DTD, вычисляется как сумма чисел битов, необходимых для кодирования каждой последовательности из ***I***. Суть алгоритма кодирования отдельной последовательности на основе данного DTD состоит в том, что сначала последовательность кодируется строкой индексов (чисел ***0***, ***1***, ***2***, ***3***…), после чего каждый индекс по определенному правилу кодируется последовательностью битов. Принцип кодирования последовательности строкой индексов основан на следующих соображениях:

- последовательность ***a*** кодируется на основе DTD ***a*** пустой строкой ***ε***
- последовательность ***b*** кодируется на основе DTD ***a|b|c*** индексом ***1*** (который обозначает позицию ***b*** в дизъюнкции ***a|b|c***, начиная с ***0***)
- последовательность ***ccc*** кодируется на основе DTD ***c**** индексом ***3*** (который обозначает число повторений символа ***c***)

Формально, алгоритм кодирования последовательности ***s*** на основе данного DTD ***D*** описывается следующим образом.

Обозначим последовательность индексов, которая кодирует последовательность ***s*** на основе регулярного выражения ***D***, как ***seq(D,s)***. Для вычисления ***seq(D,s)*** рекурсивно применяются правила 1-4:

1. ***seq(D,s)*** = ***ε*** если ***D=s*** (где ***ε*** – пустая строка). Это правило применяется в том случае, когда регулярное выражение ***D*** не содержит метасимволов, т. е. является просто последовательностью символов из ***Σ***.
2. $seq(D_0 \ldots D_k, s_0 \ldots s_k) = seq(D_0, s_0) \ldots seq(D_k, s_k)$. Это правило применяется в том случае, когда регулярное выражение ***D*** можно

представить в виде объединения выражений $D_0 \ldots D_k$, а последовательность $s$ – в виде объединения подпоследовательностей $s_0 \ldots s_k$ таким образом, что каждая подпоследовательность $s_i$ соответствует выражению $D_i$.

3. $seq(D_0|\ldots|D_m,s) = i\ seq(D_i,s)$. Это правило применяется, когда регулярное выражение $D$ представляет собой дизъюнкцию выражений $D_0 \ldots D_m$, и последовательность $s$ соответствует выражению $D_i$.
4. Наконец, $seq(D^*,s_1 \ldots s_k) =$
   - $k\ seq(D,s_1) \ldots seq(D,s_k)$, если $k>0$
   - $0$, если $k=0$

Правило 4 применяется в том случае, когда последовательность $s$ можно представить в виде объединения подпоследовательностей $s_1 \ldots s_k$ таким образом, что каждая подпоследовательность $s_i$ соответствует регулярному выражению $D$. Это же правило применяется для $seq(D+,s_1 \ldots s_k)$ и $seq(D?,s_1 \ldots s_k)$: в первом случае значение $k$ всегда больше $0$, во втором случае $k$ может принимать только значения $0$ и $1$.

После того, как построена последовательность индексов, каждый индекс представляется в виде последовательности битов. Индексы $0$ и $1$ представляются битами 0 и 1 соответственно. Каждый индекс $k > 1$ представляется в виде последовательности битов по следующему правилу. Сначала вычисляется число двоичных разрядов, необходимых для представления данного индекса в двоичной форме: $]\log_2(k+1)[$. Затем строится последовательность длиной $2*]\log_2(k+1)[+1$, где первые $]\log_2(k+1)[$ символов – это единицы, число которых обозначает число двоичных разрядов, необходимых для представления данного индекса в двоичной форме; потом идет $0$ в качестве разделителя; и затем – последовательность из $]\log_2(k+1)[$ нулей и единиц, представляющих уже сам индекс $k$ в двоичной форме. Таким образом, $0 \leftrightarrow 0$, $1 \leftrightarrow 1$, $2 \leftrightarrow 11010$, $3 \leftrightarrow 11011$, $4 \leftrightarrow 1110100$, $5 \leftrightarrow 1110101$ и т. д.

Очевидно, что схема кодирования должна подразумевать возможность однозначного восстановления исходной последовательности, то есть выполнения декодирования: последовательность битов → последовательность индексов + DTD → исходная последовательность. Нетрудно заметить, что предложенная в XTRACT схема кодирования допускает различные варианты восстановления последовательности индексов из последовательности битов (например, последовательность битов 11011 может означать как 11011 ↔ ***3***, так и 1 1 0 1 1 ↔ ***1 1 0 1 1***). Чтобы устранить эту неоднозначность, мы модернизируем метод кодирования (см. пункт 3.4.6).

**Построение DTD с минимальной MDL-стоимостью.** Задача выбора из множества регулярных выражений $S_F$ такого подмножества $S$, которое покрывает все последовательности из исходного набора $I$ и при этом имеет минимальную MDL-стоимость, может быть строго сформулирована следующим образом:

$$S = \min_{\substack{\text{по всем} \\ F \subset J}} \left( \sum_{\substack{\text{по всем} \\ j \in F}} c(j) + \sum_{\substack{\text{по всем} \\ i \in C}} \min_{\substack{\text{по всем} \\ j \in F}} [d(j,i)] \right)$$

Здесь введены следующие обозначения: $J$ – множество регулярных выражений $S_F$; $F$ – подмножество регулярных выражений из $S_F$; $j$ – регулярное выражение из $S_F$; $C$ – исходный набор последовательностей $I$; $i$ – последовательность из $I$; $c(j)$ – длина последовательности битов, кодирующей регулярное выражение $j$; $d(j,i)$ – длина последовательности битов, кодирующей последовательность $i$ на основе регулярного выражения $j$ (если регулярное выражение $j$ не покрывает последовательность $i$, то $d(j,i)$ полагается равной ∞). Алгоритмы решения такого рода задач хорошо исследованы в литературе [65], [66]. Мы использовали адаптированный алгоритм из [65]. Итоговый DTD представляет собой дизъюнкцию всех регулярных выражений из $S$.

### 3.4.6. Развитие алгоритмов генерирования DTD

Мы развили описанные выше алгоритмы генерирования DTD и построили на из основе систему автоматического генерирования DTD для массива XML-документов [5]. Наиболее важные изменения и модернизации описаны ниже.

**Во-первых,** в модуле обобщения были изменены процедуры DiscoverSeqPattern и DiscoverOrPattern. В системе XTRACT эти процедуры выполняют замены вида ***xxx…x*** → ***(x)**** и ***xyxzyz*** → ***(x|y|z)****. Очевидно, что такие замены некорректны: если следовать синтаксису DTD, то должны выполняться замены вида ***xxx…x*** → ***(x)+*** и ***xyxzyz*** → ***(x|y|z)+***. Преобразование выражений вида ***(x)+*** в выражения вида ***(x)**** в нашей системе осуществляется на этапе факторизации: точно так же, как дизъюнкции вида ***(a|1)*** заменяются на ***a?***, дизъюнкции вида ***(a+|1)*** заменяются на ***a****.

**Во-вторых,** в модуль обобщения была добавлена новая процедура DiscoverPlusPattern. Дело в том, что в реальных DTD часто встречаются выражения вида $(a_1 \cdot a_2 \cdot \ldots a_n \cdot)+$, где *"·"* означает *?*, *+* или *** (так называемые *"+"*-выражения), но система XTRACT практически не способна распознавать их. Например, обработка последовательности ***abcabbaccabc*** в системе XTRACT даст регулярное выражение ***(a|b*|c*)****, а не более точное ***(ab*c*)+***. Для генерирования *"+"*-выражений мы разработали собственный алгоритм (см. 3.4.7).

**В-третьих,** была устранена неоднозначность в схеме кодирования системы XTRACT, описанная выше. Для того, чтобы обеспечить взаимно однозначное соответствие между последовательностью индексов и последовательностью битов, мы осуществляем кодирование индексов ***0*** и ***1*** по общему правилу – то есть, ***0*** кодируется в виде последовательности битов 100, ***1*** – в виде последовательности битов 101. Например, последовательность индексов ***1 4 0 1 7*** кодируется в нашей системе в виде последовательности

битов 101 1110100 100 101 1110111. Такой способ кодирования позволяет однозначно восстановить исходную последовательность индексов.

**В-четвертых,** была реализована возможность эмпирической настройки схемы кодирования. Пользователь нашей системы имеет возможность задавать весовые коэффициенты для метасимволов при вычислении MDL-стоимости DTD, а также устанавливать весовой коэффициент MDL-стоимости самого DTD ***A*** по отношению к MDL-стоимости кодирования ***B***.

### 3.4.7. Генерирование "+"-выражений

Очевидно, что для произвольной последовательности ***s***, которая содержит ***n*** различных символов, можно построить ***n!*** различных ***"+"***-выражений, каждое из которых будет соответствовать ***s***. Например, для ***s=abaab (n=2)*** это будут ***"+"***-выражения ***(a+b)+*** и ***(b*a*)+***, первое из которых позволяет восстановить ***s*** за два повторения ***(ab|aab)***, а второе – за три ***(a|baa|b)***. В нашей системе мы ограничились построением одного ***"+"***-выражения для каждой последовательности из ***I*** – такого, для которого число повторений минимально. Алгоритм состоит из двух этапов: сначала мы строим шаблон ***"+"***-выражения, то есть конструкцию вида $(a_1 \cdot a_2 \cdot \ldots a_n \cdot)+$, после чего в этом шаблоне подставляем вместо точек соответствующие метасимволы (***?***, ******* или ***+***). Принцип работы нашего алгоритма построения ***"+"***-выражений мы опишем на примере последовательности ***abccddbaccbccddbac*** *(**n**=4)*.

Прежде всего, из исходной последовательности ***s*** мы исключаем все повторяющиеся символы (***xx…x*** заменяем на ***x***). Затем выбираем все комбинации длины ***n*** и ***n***−*1*, которые не содержат повторяющихся символов, и дополняем все комбинации длины ***n***−*1* до комбинаций длины ***n*** (***bac*** → ***bacd*** и т. д.). В нашем примере мы получим следующее множество комбинаций ***K***: ***abcd***, ***bcda***, ***cdba***, ***dbac***, ***bacd***, ***acbd***, ***bcda***, ***cdba***, ***dbac***.

Введем понятия *циклического сдвига* и *группы циклических сдвигов*. Циклическим сдвигом комбинации $k=a_1a_2\ldots a_i$ будем называть операцию

разбиения комбинации ***k*** на две произвольные части и их перестановки (например, ***abcd*** → ***ab cd*** → ***cdab***). Группой циклических сдвигов комбинации ***k*** будем называть множество всех комбинаций, которые могут быть получены из ***k*** операцией циклического сдвига. Группу циклических сдвигов будем обозначать $\{k\}_ц$. Например, $\{abcd\}_ц$ = ***{abcd, bcda, cdab, dabc}***.

Из множества комбинаций ***K*** мы выбираем все подмножества комбинаций, принадлежащих к одной группе циклических сдвигов. В нашем примере эта процедура будет выполнена за три шага: комбинации ***abcd***, ***bcda***, ***bcda*** принадлежат группе $\{abcd\}_ц$, комбинации ***cdba***, ***dbac***, ***bacd***, ***cdba***, ***dbac*** – группе $\{cdba\}_ц$, комбинация ***acbd*** – группе $\{acbd\}_ц$. Затем мы выбираем ту группу, комбинации из которой входят в множество ***K*** наибольшее число раз (в нашем примере - $\{cdba\}_ц$). Наконец, из этой группы мы выбираем ту комбинацию, которая начинается с того же символа, что и исходная последовательность ***s***, и на ее основе строим шаблон ***"+"***-выражения для ***s***. В нашем примере будет выбрана комбинация ***acdb***, которой соответствует шаблон ***(a·c·d·b·)+***.

Построение ***"+"***-выражения на основе полученного шаблона выполняется по следующему алгоритму. Сначала исходная последовательность ***s*** разбивается на минимальное число подпоследовательностей, каждая из которых соответствует одному повторению шаблона. В нашем примере: ***abccddbaccbccddbac*** → ***a__b|_ccddb|acc_b|_ccddb|ac__***. Затем для каждого символа вычисляется число его повторений в каждой такой подпоследовательности. В нашем примере: ***a*** – (1, 0, 1, 0, 1), ***c*** – (0, 2, 2, 2, 1), ***d*** – (0, 2, 0, 2, 0), ***b*** – (1, 1, 1, 1, 0). Затем в шаблон ***"+"***-выражения вместо точек подставляются метасимволы по правилу: если числа повторений – только 1, то подставляется пустая строка; если только 0 и 1, то подставляется ***?***; если 0, 1 и N>1, то подставляется *******; если только 1 и N>1, то подставляется ***+***.

В нашем примере выполнение процедуры DiscoverPlusPattern для последовательности ***abccddbaccbccddbac*** даст регулярное выражение ***(a?c*d*b?)+***, которое наряду с другими будет добавлено в множество $S_G$.

## 3.5. Система обработки XML-документов

Методы записи XML-документов в РСУБД Edge Distributive, Edge Inline, Binary Distributive, Binary Inline и Path, а также алгоритмы генерирования DTD, описанные выше, были реализованы нами в системе DTDXtract. Эта система состоит из модуля обработки XML-документов, РСУБД, модуля генерирования DTD и интерфейса пользователя.

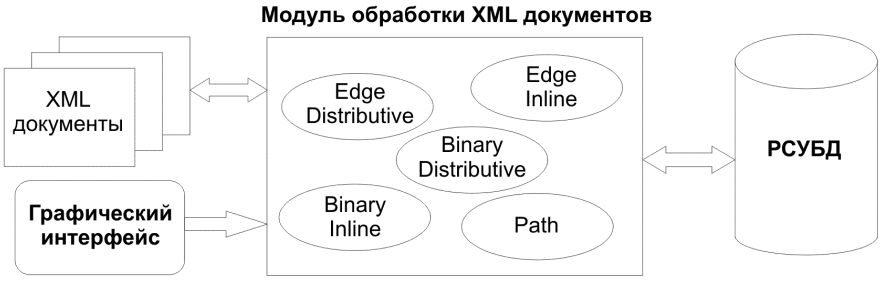



*Рис. 3.2. Архитектура модуля обработки XML-документов.*

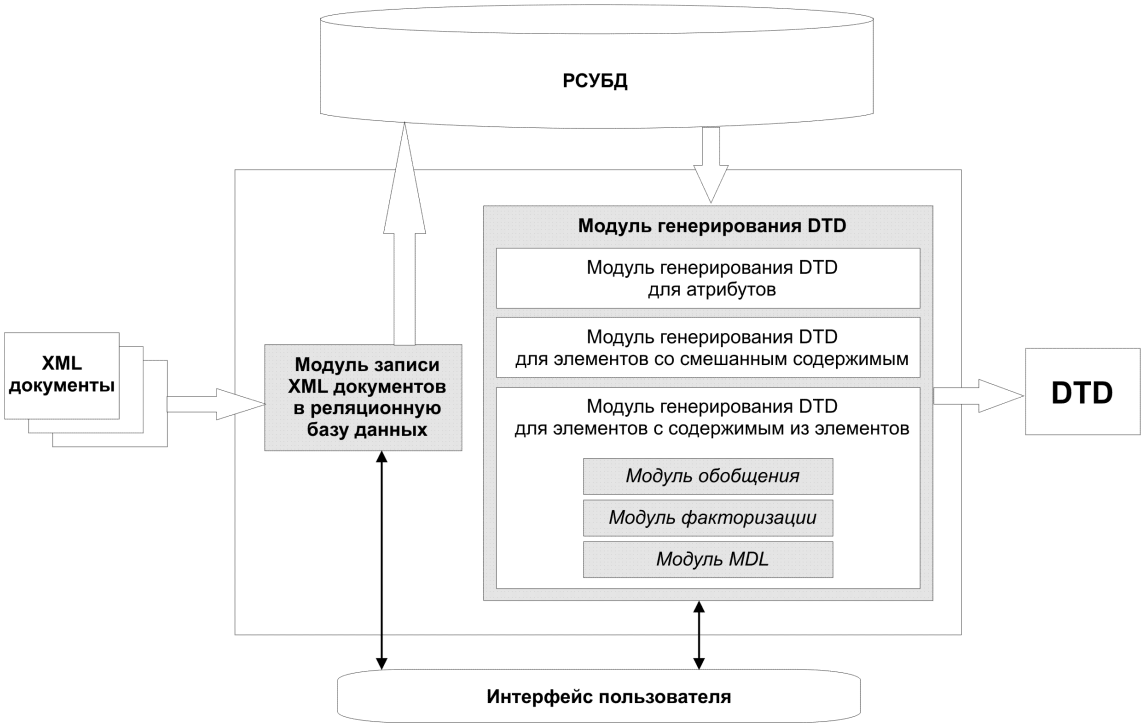



*Рис. 3.3. Архитектура модуля генерирования DTD.*

Модуль обработки XML-документов позволяет записывать XML-документы в РСУБД и извлекать их из РСУБД с использованием любого из указанных методов, рис. 3.2. Модуль генерирования DTD позволяет генерировать DTD для массива XML-документов, сохраненных в РСУБД, рис. 3.3. Модуль генерирования DTD включает в себя модуль генерирования DTD для атрибутов, модуль генерирования DTD для элементов со смешанным содержимым и модуль генерирования DTD для элементов с содержимым из элементов (который включает модули обобщения, факторизации и MDL).

В качестве РСУБД использовалась MySQL (версия 4.0.5a-beta-max) [151], установленная на рабочей станции SUN spark (512 Мб RAM) под ОС Solaris 2.7. Интерфейс пользователя был реализован на Java в среде Java 2 Standard Edition (версия 1.4.1_01) [154] на рабочей станции Pentium IV 2,4 ГГц (1024 Мб RAM) под ОС Windows 2000 (SP4). Интерфейс пользователя позволяет выбирать XML-документы для записи, указывать метод и параметры записи, задавать рабочие параметры модулей генерирования DTD.

Производительность модуля обработки XML-документов была протестирована при использовании каждого из пяти методов для записи и извлечения трех массивов XML-документов – объемом 150 Кб, 1 Мб и 10 Мб. Результаты тестирования представлены в таблицах 3.4 и 3.5.

| **Размер массива** | **Edge Distributive** | **Edge Inline** | **Path** | **Binary Distributive** | **Binary Inline** |
|---|---|---|---|---|---|
| **150 Кб** | 0,410 | 0,289 | 0,486 | 2,510 | 1,982 |
| **1 Мб** | 2,86 | 2,15 | 3,72 | 22,34 | 17,45 |
| **10 Мб** | 32,9 | 24,1 | 43,4 | 250,6 | 201,3 |

*Табл. 3.4. Время записи массива XML-документов для разных методов, сек.*

| **Размер массива** | **Edge Distributive** | **Edge Inline** | **Path** | **Binary Distributive** | **Binary Inline** |
|---|---|---|---|---|---|
| **150 Кб** | 0,410 | 0,310 | 0,657 | 236 | 187 |
| **1 Мб** | 2,43 | 2,21 | 12,83 | 1957 | 1453 |
| **10 Мб** | 38,4 | 27,4 | 160 | – | – |

*Табл. 3.5. Время извлечения массива XML-документов для разных методов, сек.*

Наилучшим методом как для записи, так и для извлечения XML-документов оказался метод Edge Inline. Наихудшие результаты показали методы типа Binary, поскольку из-за громоздких реляционных схем приходится создавать запросы с использованием конструкций типа `UNION`, которые значительно замедляют работу. Время извлечения массива XML-документов объемом 10 Мб для методов типа Binary определить не удалось из-за крайне низкой скорости работы. Производительность метода Path сравнима с методами типа Edge при записи XML-документов, но заметно отстает при извлечении XML-документов, особенно для больших массивов XML-документов. Это связано с тем, что в методе Path при извлечении необходимо получать элементы и атрибуты отдельно и по одному, а в методах Edge мы получаем все эти данные с помощью одного запроса `SELECT`.

Тестирование модуля генерирования DTD проводилось на массивах XML-документов разного объема. В одних случаях массив XML-документов имел сопровождающий DTD, и сгенерированный DTD сравнивался с оригинальным. В других случаях массив XML-документов не имел сопровождающего DTD, и качество сгенерированного DTD оценивалось субъективно. Тестирование показало высокую эффективность подсистемы генерирования DTD и подтвердило ее способность генерировать краткие и точные DTD. Результаты сравнительного тестирования систем DTDXtract, XTRACT и DDbE представлены в таблице 3.6. Видно, что процедура распознавания ***"+"***-выражений, реализованная нами, позволила повысить качество итогового DTD по сравнению с системами XTRACT и DDbE.

| **Оригинальный DTD** | **Система DDbE** | **Система XTRACT** | **Система DTDXtract** |
|---|---|---|---|
| a\|b\|c\|d\|e | a\|b\|c\|d\|e | a\|b\|c\|d\|e | a\|b\|c\|d\|e |
| (a\|b\|c\|d\|e)* | (a\|b\|c\|d\|e)* | (a\|b\|c\|d\|e)* | (a\|b\|c\|d\|e)* |
| a*b?c?d? | - | a*b?c?d? | a*b?c?d? |
| (a(bc)+d)* | (a\|b\|c\|d)+ | (a(bc)*d)* | (a(bc)+d)* |
| (ab?c*d?)* | (a\|b\|c\|d)+ | - | (ab?c*d?)* |

*Табл. 3.6. Результаты сравнительного тестирования систем DTDXtract, XTRACT и DDbE (имена элементов для простоты заменены символами).*

# 4. Интерактивное повествование в виртуальном окружении

Данная глава посвящена новому типу динамических документов – интерактивному повествованию в виртуальном окружении. Это перспективное направление развития компьютерных технологий, которое находится на стыке электронных информационных систем, компьютерных игр, обучающих программ, виртуальных тренажеров и интерактивных моделей. Технология интерактивного повествования в виртуальном окружении описана на примере обучающей системы "Виртуальный Планетарий", разработкой которой автор занимается в сотрудничестве с коллегами [6] в настоящее время. *В разделе 4.1* представлен новый тип динамических документов – интерактивное повествование в виртуальном окружении. *В разделе 4.2* рассмотрены основные методы и технологии интерактивного повествования. *В разделе 4.3* дан обзор основных технологий виртуального окружения. *В разделе 4.4* рассмотрена технологическая платформа Avango – программное обеспечение для разработки интерактивных приложений в виртуальном окружении. Наконец, *в разделе 4.5* описана архитектура и принципы интерактивного повествования в виртуальном окружении на примере обучающей системы "Виртуальный Планетарий".

## 4.1. Интерактивное повествование – новый тип динамических документов

На заре развития человечества, в древности и в средние века вплоть до изобретения книгопечатания основным способом передачи информации между людьми было устное повествование. Конечно, важнейшие законы и государственные акты высекали на камне, вырезали на бересте, писали на пергаменте и на папирусе. Священные книги надежно хранились за стенами монастырей и переписывались тысячами монахов. Даже многие научные и художественные сочинения тех времен, чудом уцелев в огне пожарищ, дошли

до нас в рукописном виде. Однако главной и часто единственной целью фиксации информации на материальном носителе была не передача ее современникам, а сохранение информации в виде неизменного – официального, канонизированного, авторского – текста. Таким образом, хотя изобретение письменности теряется в сумраке веков, создание, хранение, воспроизведение и использование *документов* тысячелетиями оставались уделом крайне узкой прослойки населения.

Изобретение книгопечатания и радио, фотографии и телевидения, вычислительных машин и сетей, Интернет и Всемирной паутины ознаменовало формирование **информационного сообщества** – "новой исторической фазы развития цивилизации, в которой главными продуктами производства являются информация и знания" [180]. В частности, совершенствование средств тиражирования и распространения документов стали этапами более чем трехсотлетней информационной революции, одним из основных признаков которой стало постепенное угасание значимости устного повествования и переход к обороту документов как основному способу обмена информацией между людьми. В процессе этого перехода и особенно в последние два десятка лет само понятие документа испытало существенную трансформацию, прежде всего благодаря появлению и развитию целого комплекса новых технологий, связанных с обработкой и представлением информации.

Вплоть до 80-х годов прошлого столетия практически все документы были статичными: однажды созданные, они более не менялись ни со временем, ни в зависимости от желаний читателя. Стремительное развитие вычислительной техники и сетей передачи данных в конце прошлого столетия породило новый объект – *электронный документ* – с совершенно новым уровнем функциональности [10]. Возможность обновлять и дополнять документ сколь угодно часто без существенных усилий позволила говорить о *живых (alive)* и *эволюционирующих (evolving)* документах. Устранение технических препятствий для использования в документе аудио и видео

компонентов привело к появлению *мультимедийных (multimedia)* документов, сделав читателя одновременно слушателем и зрителем. Возможность оперативного доступа к документу из любой точки мира в свое время казалась столь удивительной, что возникло понятие *он-лайн (on-line)* документа. Изобретение гиперссылок позволило читателю активно влиять на процесс получения интересующей его информации – документы стали *интерактивными (interactive)*. Все большую популярность приобретают *интеллектуальные агенты (information agents, intelligent agents, knowledge agents, mobile agents)* – специальные программы, которые автоматически анализируют содержание документов и происходящие с ними процессы и по результатам этого анализа предпринимают определенные действия. Наконец, документы стали управлять отдельными этапами своего *жизненного цикла (document workflow)*, что привело к появлению понятия *интеллектуального (intelligent)* документа.

Дальнейшее развитие технологий хранения, поиска и автоматической обработки информации привело к качественному переходу в области управления документами. Электронные информационные системы "научились" не просто выдавать пользователю те документы, которые были когда-то кем-то занесены "в компьютер", а автоматически генерировать по запросу пользователя новые документы с требуемыми характеристиками. Уже в середине 90-х гг. прошлого столетия в лексикон разработчиков систем электронного документооборота вошло понятие *динамического (dynamic)*, или *виртуального (virtual)* документа. Этот термин означает электронный документ, который автоматически создается системой по запросу пользователя на основе доступной информации. Именно *динамические документы* постепенно становятся ключевым объектом современных информационных систем, что ведет к глубокой перестройке существующих бизнес-процессов и схем работы с информацией [1].

Несмотря на колоссальное развитие возможностей обмена информацией, связанное с появлением электронных документов, вплоть до начала нового тысячелетия оставался целый ряд сфер человеческой деятельности, где отказ от личного взаимодействия был затруднителен или вовсе невозможен. Прежде всего, это те области практического знания, где передача профессионального мастерства и тренировка определенных навыков требует использования наглядных трехмерных моделей: различные направления искусства, прикладная наука и техника, архитектура, медицина и многое другое. Применение видео и аудио компонентов, интерактивных схем и компьютерной анимации существенно расширило возможности передачи информации посредством документов, и все же изучение тонкостей театральной постановки, порядка обслуживания авиационного двигателя или методики проведения хирургической операции до последнего времени осуществлялось преимущественно в процессе личного взаимодействия. Кроме того, применение документов как средства передачи информации было ограничено во всех процессах обучения, где существенную роль играет возможность рассказчика адаптироваться под слушателя, воспринимать его реакцию и оперативно менять глубину и акцент изложения, оставаясь при этом в рамках заданного сценария.

Активное развитие технологий *виртуального окружения (virtual environment)*, *искусственного интеллекта (artificial intelligence, AI)* и *интерактивного повествования (interactive storytelling)* позволило существенно расширить функциональность динамических документов и устранить последние барьеры на пути к их использованию в качестве основного средства обмена информацией во всех сферах человеческой деятельности. Эти технологии позволяют полностью погрузить *пользователя* (читателя, слушателя, зрителя и участника разворачивающегося действия) в трехмерную модель изучаемого явления или предметной области, анализировать его реакцию и варьировать процесс "развертывания" документа в зависимости от поведения пользователя. Современный динамический документ, построенный с

использованием технологии интерактивного повествования в виртуальном окружении, столь непохож на традиционный бумажный документ, что в его отношении с трудом удается применять привычные термины, такие, как *автор, читатель, документ*. Интерактивное повествование в виртуальном окружении напоминает одновременно театральную постановку, компьютерную игру и обучающую программу. Неудивительно, что в оборот быстро входят новые термины, большей частью заимствованные из театрального и компьютерного лексикона: разработчик, сцена, камера, свет, сюжет, персонаж.

Сегодня статичные документы постепенно утрачивают свою роль средства обмена информацией между людьми и происходит возврат к проверенной тысячелетиями практике *повествования* на новом технологическом уровне. Из модного увлечения вчерашних разработчиков систем управления знаниями, компьютерных игр и трехмерных тренажеров *интерактивное повествование* становится новой точкой роста современных информационных систем. В последние годы постоянно растет интерес к технологиям интерактивного повествования, увеличивается число публикаций и тематических конференций, посвященных различным аспектам этой технологии, усиливается внимание к ней со стороны военных и образовательных ведомств различных стран, космической промышленности, авиа- и машиностроительных корпораций, медицинского сообщества и индустрии развлечений. Все это свидетельствует о том, что именно интерактивное повествование в виртуальном окружении становится прорывным направлением в развитии электронных информационных систем.

## 4.2. Методы интерактивного повествования

Интерактивное повествование – это новый жанр компьютерных приложений, который находится на стыке традиционных информационных систем, построенных с использованием баз данных и динамических документов, компьютерных игр, образовательных программ, виртуальных

тренажеров и интерактивных моделей. Основные сферы применения интерактивного повествования – образование, обучение и тренировка. Само понятие *интерактивного повествования (interactive storytelling, virtual storytelling)* возникло сравнительно недавно, однако интерес к этому предмету в последние годы испытывает стремительный рост [19]. На данный момент есть по меньшей две международные конференции, целиком посвященные интерактивному повествованию [90], [92]; в качестве одного из разделов этот предмет присутствует в ведущих конференциях по виртуальному окружению [88], [86], [94], [95], [93], мультимедийным и обучающим системам [96], [91], [87], компьютерным играм и развлечениям [85], [89], [84]. В сети Интернет можно найти сотни научных публикаций, посвященных интерактивному повествованию, ссылки на многие из них есть в программах указанных конференций.

В рамках введенной выше понятийной системы, интерактивное повествование является одной из разновидностей динамического документа. Целью интерактивного повествования является передача пользователю определенной информации[14], которая представлена в электронном виде и может быть получена пользователем в процессе интерактивного взаимодействия с системой, причем пользователь может оказывать существенное влияние на процесс повествования. Соответственно, в нашей классификации интерактивное повествование – это электронный, динамический, интерактивный документ.

В то же время, интерактивное повествование существенно отличается и от традиционной информационной системы, построенной с использованием динамических документов, и от компьютерной игры, и от интерактивной модели, что позволяет рассматривать его как отдельный класс динамических

---

[14] Это отличает его от компьютерной игры, основная цель которой – развлечение игрока.

документов и новый жанр компьютерных приложений. Отличие интерактивного повествования от обычного динамического документа или образовательной программы заключается в широком применении современных технологий виртуального окружения и искусственного интеллекта. Благодаря этому информирование, обучение и тренировка пользователя при интерактивном повествовании происходят не столько посредством дискретной выдачи статичных блоков информации (текстов, аудио и видео компонентов), сколько непрерывно в процессе изучения пользователем виртуального мира и его взаимодействия с виртуальными персонажами. Это позволяет реализовать в интерактивном повествовании новые возможности обучения и тренировки, недоступные для традиционных динамических документов.

От виртуального тренажера или интерактивной модели интерактивное повествование отличается наличием *сюжета*. Сюжет направляет повествование по определенной траектории, предоставляя пользователю относительную свободу в выборе траектории и способов ее прохождения. Основной задачей при разработке приложений в жанре интерактивного повествования является достижение баланса между собственно повествованием (narrative), то есть передачей пользователю определенной информации по заданному сценарию, и интерактивностью (interactivity), то есть возможностью пользователя влиять на развитие повествования и процесс получения этой информации. Эти характеристики во многом конкурируют между собой: чем жестче регламентирован процесс передачи информации пользователю, тем меньше к него возможностей влиять на этот процесс; чем больше свободы у пользователя – тем сложнее направить его внимание на требуемую информацию. Эта проблема получила в литературе название *нарративного парадокса (narrative paradox)*, или *парадокса совмещения развлечения и образования (edutainment paradox)*. Методы решения нарративного парадокса и соотношение между *повествованием* и *интерактивностью* (*narrativity* / *interactivity*) являются основными характеристиками интерактивного

повествования, на основе которых может быть проведена содержательная классификация этого жанра компьютерных приложений.

Рассмотрим понятие сюжета более подробно. В общем случае, в интерактивном повествовании можно выделить участки, которые не несут значимой информации (необходимые для создания иллюзии свободы пользователя), и значимые события, которые собственно и формируют повествование. *Событие* – это предусмотренный разработчиком акт взаимодействия системы с пользователем, имеющий определенное значение в контексте информирования, обучения или тренировки пользователя. Особый тип событий составляют *узлы* – ситуации явного или неявного выбора пользователем одного из нескольких вариантов дальнейшего повествования, в зависимости от которого он в дальнейшем испытывает различные последовательности событий. К набору узлов и событий необходимо добавить *точку входа* пользователя в документ (начало повествования) и *точку выхода* пользователя из документа (конец повествования)[15]. Тогда можно определить *сюжетную линию* как последовательность узлов, событий и участков между ними от точки входа до точки выхода, по которой может пройти пользователь в процессе интерактивного повествования. Объединение всех сюжетных линий и является *сюжетом* интерактивного повествования. Если сюжет состоит из одной сюжетной линии (не имеет узлов), говорят о *линейном сюжете*, иначе – о *нелинейном сюжете*. Основные элементы сюжета изображены на рис. 4.1.

---

[15] У интерактивного повествования может быть несколько точек входа и точек выхода.

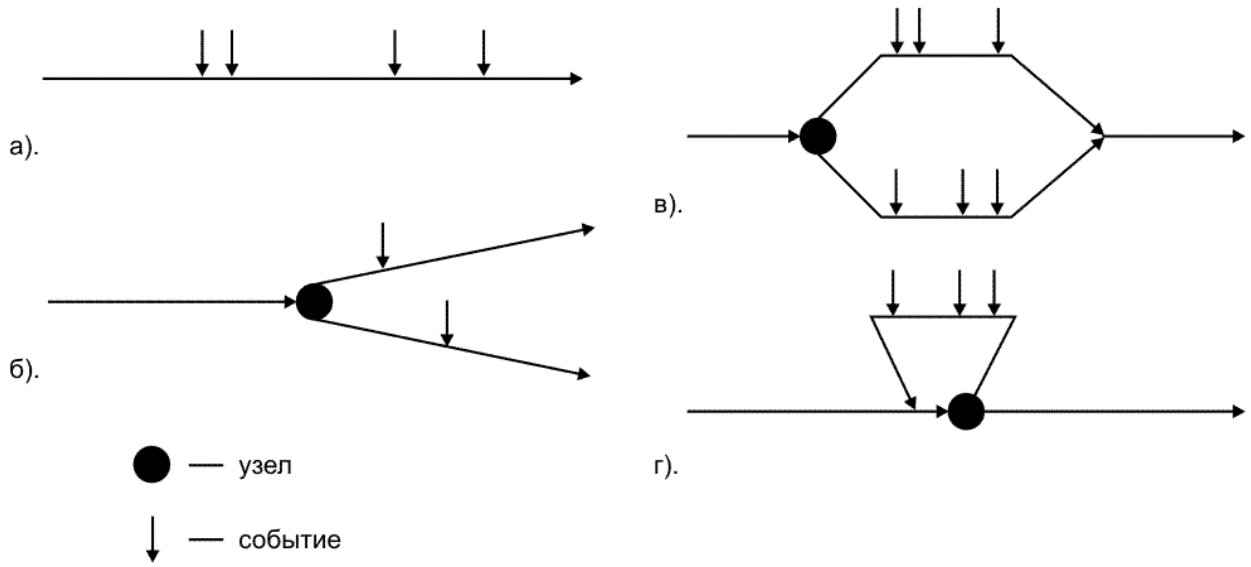



*Рис. 4.1. Основные элементы сюжета: а) линия; б) развилка; в) пучок; г) петля.*

Еще одним понятием, на котором следует остановиться более подробно, является понятие пользователя. До сих пор мы называли так человека, на которого направлено интерактивное повествование. Однако термин *пользователь (user)* не всегда адекватно отражает роль человека в интерактивном повествовании. Пользователем обычно называют человека, который выполняет работу или решает задачу с помощью компьютерной системы; этот термин слабо отражает аспекты получения информации, обучения и тренировки. Разные авторы употребляют термины *игрок (player)*, *зритель (spectator), публика (audience), персонаж (character), актер (actor)* и другие, которые хорошо подходят в одних случаях и совершенно неуместны в других. Игрок играет ради собственного развлечения; зритель в интерактивном повествовании часто является одновременно читателем, слушателем и действующим лицом; персонажем сложно назвать генерала, ведущего боевые действия с виртуальным противником… На наш взгляд, термин *пользователь*, несмотря на указанные недостатки, лучше всего подходит для обозначения человека, на которого направлено интерактивное повествование, и применим практически к любому приложению этого жанра.

В рамках введенной терминологии можно дать следующее определение **интерактивного повествования:** это динамический документ с нелинейным сюжетом, построенный с использованием технологий виртуального окружения и искусственного интеллекта и предназначенный для информирования, обучения или тренировки пользователя.

В зависимости от решаемых задач, соотношения *narrativity* / *interactivity* и предпочтений разработчика такой динамический документ может называться десятками разных способов. В литературе можно встретить такие названия, как *виртуальное повествование (virtual storytelling), интерактивная история (interactive story), нелинейное интерактивное повествование (non-linear interactive narrative), интеллектуальное виртуальное окружение (intelligent virtual environment), виртуальное развлекательно-обучающее окружение (virtual edutainment environment), интерактивная обучающая система (interactive educational system), повествовательная система (narrative system), тренировочная система в виртуальном окружении (VE training system), тренажер в виртуальном окружении (VE simulator), контекстно-зависимая игра (context-dependent game), драматургическая игра (dramaturgical gameplay), интерактивная драма (interactive drama)* и т. д. и т. п.

## 4.3. Технологии виртуального окружения

Последние десятилетия характеризуются катастрофическим ростом объемов информации, которую необходимо обрабатывать для поддержания научно-технического прогресса. Важной вехой на пути преодоления этого "кризиса данных" стал отчет Национального научного фонда США (NSF) "Визуализация в научных вычислениях" [20], подчеркнувший важность интерактивной визуализации больших массивов данных и обративший внимание научной общественности на знаменитый афоризм Хемминга: "Целью вычислений являются не числа, а понимание (постижение, проникновение в суть, интуиция, insight)". В процессе развития визуализации как научной

дисциплины было осознано, что человек тем лучше проникает в суть исследуемого явления, чем более полно он может "погрузиться" в модель этого явления и чем более естественно для него организована манипуляция данными в этой модели. Так сформировалась технология виртуального окружения, называемая также технологией виртуальной реальности.

Выражение "виртуальная реальность" (virtual reality, VR) предложил Ярон Ланье[16] (Jaron Lanier) [179] в начале 80-х гг. прошлого столетия. Одно из популярных определений этого выражения звучит следующим образом: "виртуальная реальность – это синтезированное компьютером, интерактивное, трехмерное окружение, в которое погружен человек" [21]. Это определение выделяет три основных характеристики виртуальной реальности. Во-первых, виртуальная реальность представляет собой трехмерное окружение (сцену, модель), сформированное (синтезированное) компьютером. Во-вторых, виртуальная реальность интерактивна: взаимодействие системы с пользователем происходит в удобной, естественной для человека форме в режиме реального времени. В-третьих, пользователь погружен в виртуальную реальность, то есть, восприятие человеком реального мира в виртуальной реальности частично или полностью блокируется.

Выражение "виртуальная реальность" получило широкое распространение в популярной литературе, однако оно плохо подходит для использования в качестве научного термина. Вслед за многими зарубежными авторами мы отдаем предпочтение термину *виртуальное окружение (virtual environment, VE)*, который в специальной литературе употребляется как более точный синоним "виртуальной реальности". Поскольку виртуальное окружение – это, прежде всего, технология взаимодействия человека и компьютерной

---

[16] Основатель VPL Research – одной из первых компаний, которая начала продавать системы виртуальной реальности.

системы, мы также считаем необходимым дать более строгое определение этого термина, чем приведенное в предыдущем абзаце. **Виртуальное окружение** – это технология человеко-машинного взаимодействия, которая обеспечивает погружение пользователя в трехмерную интерактивную модель изучаемого явления или предметной области и предоставляет естественный интуитивный интерфейс для взаимодействия с этой моделью.

На практике системами виртуального окружения называют широкий спектр приложений с разным соотношением реальных и виртуальных объектов и разной степенью погружения пользователя в виртуальное окружение. Между системами "чистого" виртуального окружения и системами, построенными целиком в реальном мире, располагается целый ряд приложений, в отношении которых Поль Милграм (Paul Milgram) предложил использовать термин *смешанное окружение (mixed reality, MR)*, рис. 4.2 [22]. С одной стороны, это приложения, где реальные объекты дополнены трехмерными компьютерными моделями – *дополненная реальность (augmented reality, AR)* [177], [178]. С другой стороны, это системы виртуального окружения, в которые внедрены объекты или элементы реального мира – *дополненная виртуальность (augmented virtuality, AV)*. Термин *дополненная виртуальность* не получил широкого распространения; он применяется сегодня лишь в отношении достаточно узкого класса приложений виртуального окружения, где в синтезированную компьютером трехмерную модель внедрены видеоизображения реальных людей или объектов. Напротив, технологии *дополненной реальности* в настоящее время фактически стали самостоятельным направлением развития систем виртуального окружения.


*Рис. 4.2. Диаграмма Поля Милграма (reality–virtuality continuum).*

Технологии дополненной реальности заключаются в том, что трехмерная интерактивная модель, сформированная компьютером на основании реальных данных, накладывается на изображение объекта таким образом, что пользователь воспринимает объект и модель как единое целое. При этом основной задачей является не погружение пользователя в виртуальное окружение, а представление ему дополнительной информации о реальном объекте в удобной для восприятия и интерактивной манипуляции данными форме. Например, на изображение атомного реактора может быть наложена трехмерная картина распределения температуры в реакторе, построенная на основе показаний датчиков и обновляемая в режиме реального времени с возможностью изменения масштаба, степени детализации и т. д. Технологии дополненной реальности намного менее требовательны к производительности систем визуализации, чем технологии "полного" виртуального окружения, что дает существенную экономию при разработке приложений.

Сфера приложений технологий виртуального окружения включает сегодня как традиционные фундаментальные дисциплины – физику, математику, астрономию, медицину, – так и множество прикладных направлений. Это, прежде всего, космонавтика и исследования Солнечной системы, аэрогидродинамика (расчеты динамики течения потоков), океанология и геофизика (инженерия землетрясений), металлообработка (авто- и авиаиндустрия), сопротивление материалов (моделирование эластичных объектов), исследование и конструирование оболочек (корпусов подводных лодок и ядерных реакторов), анализ столкновений и разрушений (моделирование аварий и катастроф), робототехника и телемеханика (удаленное управление машинами и механизмами), оперативная медицина и биомедицинская инженерия (хирургия, протезирование и диагностика), промышленный и потребительский дизайн, моделирование боевых действий и чрезвычайных ситуаций и многое другое.

При создании современных систем виртуального окружения широко используется техника распределенных систем и параллельных вычислений. Разработкой методов визуализации, в том числе с применением технологий виртуального окружения, занимаются во многих лабораториях мира. Значительные результаты получены в США (НАСА, Военно-морская исследовательская лаборатория, Ливерморская национальная лаборатория, все Национальные суперкомпьютерные центры и др.), Европе (Фраунгоферовский институт машинной графики, Дармштадт и Национальный исследовательский центр по информационным технологиям, Санкт-Августин – Германия; ИНРИА – Франция; Женевский университет и Лозаннская высшая политехническая школа – Швейцария; Резерфордовская лаборатория и Университет Манчестера – Великобритания и др.), азиатских странах (Сингапур, Китай, Япония и др.).

## 4.4. Технологическая платформа Avango

В любой современной системе виртуального окружения можно выделить следующие четыре основные технологические подсистемы:

- подсистема управления базой данных модели, которая описывает виртуальные объекты и сцены (геометрия, текстуры поверхности, источники освещения и др.);
- подсистема графического преобразования текущего состояния модели в базе данных в трехмерное визуальное представление модели *(рендеринг – rendering)*;
- подсистема стереоскопического проецирования визуального представления модели на экран, которая создает иллюзию погружения пользователя в трехмерную виртуальную среду и блокирует при этом его восприятие реального мира;
- подсистема локализации и слежения за положением и ориентацией головы и глаз пользователя *(трекинг – tracking)*.

Кроме того, в состав современных систем виртуально окружения часто входят дополнительные подсистемы, которые позволяют усилить иллюзию погружения в виртуальную трехмерную модель и упростить манипуляцию данными в этой модели:

- подсистема синтеза звуковых эффектов, чувствительная к положению и ориентации пользователя в пространстве;
- подсистема генерации силовых и тактильных ощущений, создающая иллюзию прикосновения к виртуальным объектам;
- устройства ввода для построения гибкого интерфейса манипулирования данными (указки, панели управления, кинетические сенсоры и др.);
- методика взаимодействия с виртуальными объектами и персонажами, которая подменяет пользователю привычные способы взаимодействия в реальном мире.

Одним из наиболее развитых программных пакетов для создания распределенных интерактивных приложений в виртуальном окружении является система Avango, разработанная во Фраунгоферовском институте медиакоммуникаций (Санкт-Августин, Германия) [151], [23]. Это среда программирования, которая имеет открытый исходный код и работает под управлением операционной системы Linux. Система Avango предоставляет гибкие возможности для разработки приложений виртуального окружения.

Для описания объектов виртуальной модели в системе Avango используется язык программирования C++. Кроме того, Avango поддерживает язык программирования высокого уровня Scheme, который позволяет оперировать со структурированными данными (строки, списки, векторы). Все объекты Avango могут быть описаны на Scheme. Для преобразования текущего состояния модели в трехмерное визуальное представление в Avango используется графическая система OpenGL Performer. Эта система обеспечивает связь с аппаратным обеспечением и выполняет ресурсоемкие задачи рендеринга, такие, как стирание невидимых граней, переключение

степени детализации и т. д. Трехмерное визуальное представление модели генерируется с частотой не менее 20 раз в секунду и привязано к точке зрения пользователя.

В Avango могут быть определены объекты двух типов – узлы и датчики. *Узлы (nodes)* используются для описания объектов модели, которые могут отображаться в визуальном представлении. Совокупность узлов приложения составляет иерархическую структуру, которая называется *графом сцены (scene graph*) – по аналогии с театральной сценой, объединяющей все предметы и действующих лиц спектакля. *Сенсоры (sensors)* обеспечивают взаимодействие модели с реальным миром, но не могут быть включены в граф сцены и не отображаются в визуальном представлении. Данные, генерируемые внешним устройством (клавиатура, мышь, указка и т. д.), записываются в поля сенсора, после чего посредством взаимосвязей между сенсорами и узлами вводятся в граф сцены.

Информация о состоянии каждого объекта представляется в виде совокупности *полей (fields)* – иначе говоря, объекты Avango являются *полевыми контейнерами (fieldcontainers)*. Между полями данных различных объектов Avango могут быть установлены *взаимосвязи (fieldconnections)* – если меняется значение *поля-источника (source field)*, то такое же значение записывается в *поле-адресат (destination field)*. Механизм взаимосвязей между полями позволяет задать дополнительные отношения между объектами, которые не могут быть описаны в рамках графа сцены. Взаимосвязи обеспечивают ввод данных из реального мира в виртуальную модель и позволяют организовать интерактивное взаимодействие пользователя с моделью.

Комплекс взаимосвязей между полями различных объектов Avango формирует *граф потока данных (dataflow graph)*. В определенном смысле граф потока данных ортогонален графу сцены – если граф сцены служит для иерархического описания объектов модели (на столе стоит ваза, в вазе лежит апельсин…), то граф потока данных отражает интерактивное взаимодействие

пользователя с объектами модели (персонаж подошел к столу и положил апельсин в карман…). Полный *просчет графа потока данных (dataflow graph evaluation)* осуществляется на каждом *шаге рендеринга (rendering frame)*.

Преобразование текущего состояния модели в трехмерное визуальное представление требует высокой скорости доступа к объектам. Для этого все объекты, участвующие в процессе рендеринга, должны находиться в локальной памяти этого процесса. Поэтому для создания *распределенных приложений (distributed application)* – то есть приложений, которые поддерживают одновременное выполнение нескольких процессов рендеринга на одном графе сцены – необходимо обеспечить дублирование объектов для использования одного объекта в нескольких процессах.

Для создания распределенных приложений в Avango используется два осноных механизма – *распределенная общая память (distributed shared memory, DSM)* и потоковый интерфейс. DSM – это сегмент локальной памяти, который доступен одновременно нескольким процессам на одной машине. Каждый процесс обладает локальной копией сегмента DSM, все копии синхронизированы друг с другом. В сегменте DSM могут быть созданы *группы распределеных объектов (distribution group)*. Каждый процесс Avango может подключиться к одной или нескольким таким группам.

Объекты Avango могут быть *локальными (local object)* или *распределенными (distributed object)*. Локальный объект существует в адресном пространстве одного процесса, распределенный объект копируется в соответствующую группу и далее в адресное пространство всех процессов, подключенных к этой группе. Для копирования объектов в системе Avango используется универсальный *потоковый интерфейс (streaming interface)*, который позволяет записывать информацию о текущем состоянии объекта в *последовательный поток данных (stream)* и затем реконструировать объект из потока.

Возможность создания *распределенных приложений (distributed application)* является ключевым отличием Avango от аналогичных систем, предназначенных для разработки приложений виртуального окружения. Поддержка нескольких процессов рендеринга на одном графе сцены позволяет тестировать и отлаживать одновременно несколько модулей работающего приложения. Это существенно повышает эффективность коллективной работы и сокращает время разработки сложных приложений виртуального окружения.

## 4.5. Обучающая система "Виртуальный Планетарий"

Рассмотрим архитектуру и принципы построения интерактивного повествования в виртуальном окружении на примере обучающей системы "Виртуальный Планетарий", которая разрабатывается в настоящее время в Институте физико-технической информатики. Эта система основана на трехмерной модели Солнечной системы в виртуальном окружении, разработанной специалистами Института физико-технической информатики и Фраунгоферовского института медиакоммуникаций (Санкт-Августин, Германия) [24], [25]. Обучающая система "Виртуальный Планетарий" включает в себя следующие структурные элементы: модель Солнечной системы в виртуальном окружении (*модель*), РСУБД с информацией об объектах Солнечной системы (*РСУБД*), набор прототипов (шаблонов) для построения меню (*меню*) и программный модуль, который обеспечивает взаимодействие между пользователем и элементами системы (*программный модуль*).

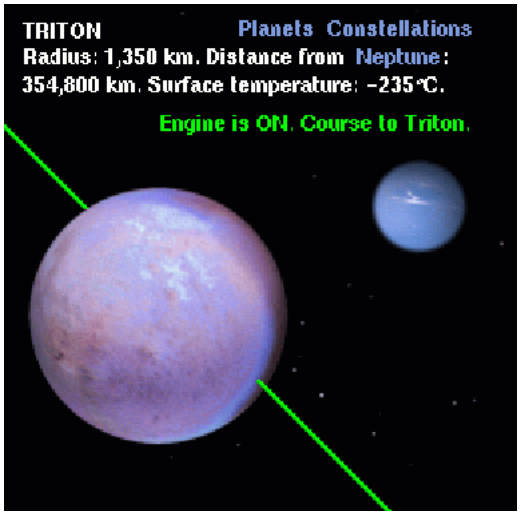



*Рис. 4.3. Фрагмент модели Солнечной системы в виртуальном окружении: Нептун и его спутник Тритон.*

*Рис. 4.4 Аватар "Виртуальный экскурсовод".*

*Модель* обеспечивает следующую функциональность: задает координаты и внешний вид основных объектов Солнечной системы (Солнце, 9 планет, 32 спутника), звезд и созвездий (3200 звезд), обеспечивает визуализацию объектов в виртуальном трехмерном пространстве в соответствии с их координатами и текстурами поверхности в зависимости от положения и ориентации "космолета" (в котором находится пользователь), обеспечивает перемещение космолета по виртуальному пространству (в автоматическом режиме – подлет к объекту в соответствии с координатами космолета по умолчанию для данного объекта, в ручном режиме – приближение к объекту, удаление от объекта, облет вокруг объекта), рис. 4.3. В дальнейшем планируется расширить модель, прежде всего, добавить новые объекты (спутники, кометы, пояса астероидов) и обеспечить возможность демонстрации основных астрономических явлений (солнечное и лунное затмение).

*РСУБД* содержит информацию о всех объектах модели. Во-первых, это данные о положении объекта в общей иерархической структуре объектов (уникальный номер объекта, название объекта, тип объекта, уникальный номер родительского объекта, порядковый номер среди всех потомков того же родителя). Во-вторых, это сведения познавательного характера,

представленные в виде отдельных информационных блоков (общая характеристика, история исследований, мифы и легенды и т. д.). Каждый информационный блок представлен в двух вариантах: текст и аудиофайл; в дальнейшем планируется дополнить систему модулем автоматического синтеза речи и оставить только текстовые информационные блоки. Часть таблицы РСУБД с информацией об иерархии объектов, представлена в табл. 4.1.

| **Уникаль-ный номер объекта** | **Название объекта** | **Тип объекта** | **Уникаль-ный номер родитель-ского объекта** | **Порядко-вый номер среди потомков того же родителя** |
|---|---|---|---|---|
| 1 | Вселенная | вселенная | 0 (корень) | 1 |
| 2 | Млечный путь | галактика | 1 | 1 |
| 10 | Солнечная | солнечная система | 2 | 1 |
| 20 | Солнце | звезда | 10 | 1 |
| 25 | Юпитер | планета | 10 | 6 |
| 26 | Сатурн | планета | 10 | 7 |
| 30 | Ио | спутник | 25 | 6 (условно) |
| 41 | Пояс Копейра | пояс астероидов | 10 | 2 (условно) |
| 50 | Комета Галлея | комета | 10 | 15 (условно) |
| 100 | Центавр | созвездие | 2 | 56 (условно) |
| 500 | Альфа Центавра | звезда | 100 | 1 |

*Табл. 4.1. Часть таблицы РСУБД с информацией о иерархии объектов.*

Предложенная система иерархии нуждается в некоторых пояснениях. Строгая научная иерархия (вселенная – галактика – звездная система – звезды и планеты) имеет два существенных недостатка. Во-первых, в ней нет места созвездиям, так как они не являются физическими объектами. Хотя такое понятие, как созвездие, постепенно исчезает из современной астрономии, отказаться от созвездий в обучающей системе не представляется возможным. Во-вторых, научная иерархия не выделяет особую роль солнечной системы. Между тем, очевидно, что для людей солнечная система – объект совершенно иного ранга, чем другие звездные системы (также, как и Солнце – не просто звезда). Иерархия объектов в нашей системе построена таким образом, чтобы

устранить указанные недостатки. Созвездия считаются дочерними объектами нашей галактики и родительскими объектами звезд нашей галактики. Солнечная система рассматривается как уникальный объект, который имеет в качестве родительского объекта нашу галактику, а в качестве дочерних – Солнце и планеты.

Предлагаемая архитектура РСУБД позволяет автоматически строить меню для любого объекта на основе прототипа (шаблона), единого для каждого типа объектов, и легко формировать различные сценарии для автоматического путешествия по модели Солнечной системы. Можно выбирать из РСУБД все объекты одного типа, располагать по порядку всех потомков одного родителя, строить различные подборки информационных блоков и т. д. При необходимости РСУБД может быть легко дополнена как новыми объектами, так и дополнительными полями для всех объектов. Также в дальнейшем можно расширить набор типов объектов – в дополнение к вселенной, галактикам, созвездиям, звездам, солнечной системе, планетам, спутникам, кометам и поясам астероидов ввести квазары, туманности, звездные системы и т. д.

*Меню* накладывается на изображение трехмерной модели солнечной системы и обеспечивает навигацию пользователя в виртуальном окружении. С точки зрения пользователя меню выглядит как приборная доска для управления космолетом, в котором находится пользователь. Меню состоит из трех панелей, которые имеют разные функции.

*Общая панель* располагается внизу и содержит основные элементы управления, которые должны быть доступны пользователю в любой момент. Прежде всего, это кнопки "в начало", "наверх" и "выход". При нажатии кнопки "в начало" космолет перемещается в начальное состояние (у Земли) и выводится основное меню (меню солнечной системы). При нажатии кнопки "наверх" космолет перемещается к объекту, родительскому для данного объекта, и выводится соответствующее меню. При нажатии кнопки "выход" система завершает работу. Также на общей панели могут располагаться

переключатель между автоматическим и ручным режимами, различные индикаторы и т. д. Общая панель отображается постоянно при работе пользователя с системой.

*Навигационная панель* располагается слева и содержит список дочерних объектов того объекта, у которого в данный момент находится космолет (для планеты – список спутников, для созвездия – список звезд и т. д.). При щелчке мышью на определенном пункте панели навигации (например, названии спутника) космолет перемещается в виртуальном пространстве к выбранному объекту и на экран выводится новое меню, соответствующее данному объекту.

*Информационная панель* располагается справа и содержит список текстов и аудио-файлов, соответствующих отображаемому объекту. При щелчке мышью на определенном пункте информационной панели соответствующий информационный блок выводится пользователю (отображается на "информационном табло" или проигрывается аудиосистемой). Навигационная и информационная панели меню отображаются в ручном режиме управления и не отображается в автоматическом.

Технически меню реализовано как набор прототипов (шаблонов), в которые программный модуль подставляет требуемую информацию из РСУБД. Для каждого типа объектов создается свой прототип меню. Когда пользователь выбирает конкретный объект, программный модуль находит нужный прототип, подставляет в него информацию, соответствующую выбранному объекту, и передает готовое меню системе визуализации. Например, прототип меню объекта "планета" включает в себя список спутников на навигационной панели и список ссылок на информационные блоки на информационной панели. Когда пользователь выбирает конкретную планету, программный модуль на основе этого прототипа автоматически формирует меню со списком спутников этой планеты и ссылками на информационные блоки, которые доступны в РСУБД.

В дальнейшем интерфейс взаимодействия пользователя с системой планируется расширить. Кнопочное меню ("приборная доска для управления

космолетом"), будет дополнено виртуальным экскурсоводом – персонажем, полностью синтезированным системой, рис. 4.4. Такого виртуального персонажа в системах интерактивного повествования в виртуальном окружении принято называть *аватаром (avatar)*. Реализация аватара представляет собой более сложную задачу, чем создание кнопочного меню. Для этого требуется ввести в систему модуль искусственного интеллекта (AI), который будет отвечать за поведение аватара и его взаимодействие с пользователем. В то же время, аватар существенно усиливает иллюзию погружения в виртуальное окружение и делает взаимодействие пользователя с системой более удобным и естественным.

*Программный модуль* отслеживает управляющие сигналы, поступающие от пользователя (в ручном режиме) или со стороны запущенного скрипта (в автоматическом режиме), и выполняет соответствующие действия. При щелчке мышью в виртуальном пространстве управление передается модели, которая обеспечивает перемещение космолета в пространстве вокруг объекта. При щелчке мышью на общей или навигационной панели управление передается модели, которая обеспечивает перемещение космолета в выбранную точку виртуального пространства, и интерпретатору (одному из компонентов программного модуля), которая строит новое меню. При щелчке мышью на информационной панели выдается соответствующий текст или аудио-файл.

Перечислим основные возможности описанной выше системы:

1. Представляет в виртуальном окружении основные объекты Солнечной системы (Солнце, планеты и спутники), а также звезды и созвездия.
2. Обеспечивает навигацию по виртуальной Солнечной системе ("полет") в ручном режиме или автоматически по заданному сценарию.
3. Содержит информацию о всех объектах в виде набора информационных блоков (тексты, аудио-файлы), который легко изменить или дополнить.
4. Представляет в наглядной форме основные астрономические явления (солнечное и лунное затмения, смена сезонов года на Земле и т. п.).

5. Имеет простой интерфейс для редактирования РСУБД и создания сценариев (скриптов) путешествий в автоматическом режиме.

Таким образом, обучающая система "Виртуальный Планетарий" представляет собой классический пример интерактивного повествования в виртуальном окружении. Пользователь может выбирать маршрут путешествия по виртуальной солнечной системе и карте звездного неба, читать или прослушивать интересующую его информацию о наблюдаемых объектах и явлениях. При этом свобода перемещения пользователя в виртуальном окружении, с одной стороны, ограничена рамками сюжета, с другой – достаточна для создания иллюзии самостоятельного путешествия в космолете по космическому пространству.

Принципы построения интерактивного повествования в виртуальном окружении, описанные на примере системы "Виртуальный Планетарий", могут использоваться для создания информационных, обучающих и тренировочных систем в самых разных областях. В частности, технология интерактивного повествования может эффективно применяться для создания инструкций по эксплуатации и документации к технологически сложным изделиям, в том числе, "двойного" назначения. Ведущие предприятия во многих странах мира сегодня активно внедряют системы CALS, PLCS, PLM [102]. Методы интерактивного повествования в виртуальном окружении могут существенно расширить возможности таких систем.

# Заключение

Основные результаты данной работы представлены ниже. Описание каждого результата состоит из краткой формулировки, основных тезисов, комментария с оценкой научной новизны и практической значимости и списка публикаций, в которых был отражен данный результат.

**1. Исследован новый класс электронных документов – динамические документы.**

- Динамический документ – это документ, создаваемый системой по запросу пользователя на основе доступной информации.
- Динамические документы обладают намного более широкой функциональностью, чем традиционные статичные электронные документы.
- Концепция динамических документов широко применяется для построения современных электронных информационных систем.

Понятие динамического документа появилось в зарубежной литературе около 10 лет назад [73]. Однако, до сих пор не был проведен содержательный анализ этого понятия и связанного с ним комплекса методов и технологий. В данной работе концепция динамических документов впервые представлена в целостном, логически связанном виде. Описана история развития электронных документов, рассмотрены возможности электронных документов, показан механизм возникновения нового класса электронных документов – динамических документов, описаны характеристики динамических документов, исследованы их возможности и преимущества.

Данные результаты отражены в публикации [1].

**2. Разработана технология автоматизированной подготовки динамических документов.**

- Технология автоматизированной подготовки динамических документов основана на хранении структурированной информации в РСУБД и использование прототипов.
- С использованием разработанной технологии построена система автоматизированной подготовки и публикации документов на корпоративном сайте, которая была внедрена в эксплуатацию в компании "Телеком Транспорт" в 2000-2002 гг. и успешно функционирует в настоящее время.

Представленная технология автоматизированной подготовки динамических документов по своей архитектуре близка к технологии построения так называемых *динамических* сайтов. Однако существующие технологии построения динамических сайтов разработаны и описаны, как правило, либо с точки зрения программиста, либо с точки зрения дизайнера (верстальщика). В первом случае объектом исследования являются программные продукты и языки программирования, а целью – создание на их основе новых программных модулей, интеграция различных программных продуктов друг с другом, разработка новых алгоритмов и приемов программирования. Во втором случае объект исследования – это языки разметки (HTML и др.), а цель – наиболее эффективное отображение информации на экране монитора с учетом характеристик компьютеров и программного обеспечения пользователей.

Технология, представленная в данной работе, разработана и описана с точки зрения разработчика (конструктора) динамических документов, цель которого – наиболее эффективная организация информационного взаимодействия между электронной информационной системой и ее пользователями. Объектом исследования являются динамические документы – новый класс электронных документов, которые предоставляют намного более

широкие возможности управления информацией, чем традиционные статичные электронные документы. Результатом исследования является новая технология работы с информацией, основанная на использовании динамических документов. Эта технология может применяться для построения электронных информационных систем самых разных типов – корпоративных сайтов, баз знаний, экспертных систем и т. д. В частности, на ее основе разработана технология интерактивного повествования в виртуальном окружении, описанная в данной работе.

Технология автоматизированной подготовки динамических документов, представленная в данной работе, может использоваться для построения электронных информационных систем разной функциональности и масштаба. Она представляет интерес для разработчиков современных электронных информационных систем, которых не удовлетворяет функциональность статичных электронных документов и которые стремятся расширить возможности работы с информацией. Эта технология может использоваться для построения корпоративных информационных систем, баз знаний, систем управления знаниями, корпоративных сайтов, обучающих программ, экспертных систем, публичных информационных порталов и т. д.

Данные результаты отражены в публикации [2].

**3. Разработана технология записи массива XML-документов в РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов.**

- Технология записи массива XML-документов в РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов позволяет автоматизировать занесение структурированной информации в таблицы РСУБД и тем самым существенно повысить эффективность автоматизированной подготовки динамических документов.

- Технология записи массива XML-документов в таблицы РСУБД без использования информации об их структуре и автоматического генерирования DTD для этого массива XML-документов была разработана и реализована в виде экспериментальной системы в 2003-2004 гг.

Представленная технология записи массива XML-документов в РСУБД без использования информации об их структуре и генерирования DTD для этого массива XML-документов является новой. В литературе описан ряд алгоритмов записи отдельного XML-документа в РСУБД без использования информации о его структуре [53], [54], [55]. Также в литературе описан алгоритм построения DTD для отдельного XML-элемента [57]. В данной работе задача генерирования DTD для массива XML-документов впервые рассмотрена как часть более общей задачи автоматического занесения структурированной информации в РСУБД электронной информационной системы. Разработанная технология записи массива XML-документов в РСУБД и генерирования DTD для этого массива XML-документов позволяет автоматизировать наполнение РСУБД информацией и тем самым существенно повысить эффективность автоматизированной подготовки динамических документов. Эта технология реализована в виде экспериментальной системы, которая может использоваться как для решения прикладных задач, так и для дальнейших научных разработок.

Технология записи массива XML-документов в РСУБД без использования информации о их структуре и генерирования DTD для этого массива XML-документов, описанная в данной работе, представляет интерес для разработчиков электронных информационных систем, которым необходимо автоматизировать наполнение РСУБД структурированной информацией. Эта задача неизбежно возникает при развитии любой электронной информационной системы, когда ручное занесение информации в систему становится неэффективным и перестает удовлетворять возросшим требованиям к объему и качеству структурирования информации. Представленная технология

генерирования DTD для массива XML-документов в комплексе с системами автоматического поиска информации и конвертерами информации из документов и баз данных в формат XML обеспечивает эффективное решение задачи автоматического наполнения РСУБД структурированной информацией. Структурированная информация из таблиц РСУБД может быть легко использована для автоматизированного построения динамических документов.

Данные результаты отражены в публикациях [4], [5].

**4. Исследован новый тип динамических документов – интерактивное повествование в виртуальном окружении, и разработана технология интерактивного повествования в виртуальном окружении.**

- Интерактивное повествование в виртуальном окружении – это новый жанр компьютерных приложений, который находится на стыке электронных информационных систем, компьютерных игр, обучающих программ, виртуальных тренажеров и интерактивных моделей.
- Технология интерактивного повествования в виртуальном окружении основана на интеграции технологий динамических документов и виртуального окружения на технологической платформе Avango.
- На основе представленной технологии интерактивного повествования в виртуальном окружении разработана экспериментальная обучающая система "Виртуальный Планетарий".

Технология интерактивного повествования в виртуальном окружении, представленная в данной работе, является новой. В мире есть несколько десятков коллективов, которые занимаются разработкой методов и технологий интерактивного повествования в виртуальном окружении [90], [92]. Однако, как и в любой новой предметной области, понятие *интерактивного повествования* по-разному трактуется разными исследователями. Этот факт в сочетании с широким спектром систем и технологий виртуального окружения приводит к тому, что каждый коллектив фактически разрабатывает свою

технологию интерактивного повествования в виртуальном окружении, которая существенно отличается от других разработок. Представленная технология интерактивного повествования в виртуальном окружении основана на интеграции технологий динамических документов и виртуального окружения на технологической платформе Avango [23]. Это новый подход, который ранее не рассматривался и не был описан другими исследователями.

Технология интерактивного повествования в виртуальном окружении, описанная в данной работе, представляет интерес для разработчиков электронных информационных, обучающих и тренировочных систем. Эта технология основана на технологической платформе Avango, которая имеет открытый исходный код и распространяется свободно [23]. Стоимость системы виртуального окружения на Linux-кластере персональных компьютеров сегодня вполне доступна для крупных отечественных научных центров, ВУЗов, промышленных и добывающих корпораций [93]. Учитывая, что стоимость разработки приложений виртуального окружения на базе программного обеспечения с открытым исходным кодом на порядок меньше, чем стоимость фирменных систем с аналогичной функциональностью, можно предположить, что круг потенциальных пользователей предложенной технологии интерактивного повествования в виртуальном окружении достаточно широк. Среди возможных применений данной технологии – создание инструкций по эксплуатации и документации к технологически сложным изделиям, в том числе, "двойного" назначения, в рамках концепций CALS, PLCS, PLM [102].

Данные результаты отражены в публикации [6].

# Приложения

```
<html><head><title>Список последних новостей</title>……</head>
<body>……
<h1> Список новостей за {#Текущий_месяц#} месяц</h1>
<table>
{##Начало_списка_новостей##}
{##Вариант_оформления_новости_1##}
<tr>
<td>{#Дата_публикации_новости#}</td>
<td><img src=http://www.company.ru/{#Ссылка_на_иллюстрацию_новости#}></td>
<td><a href=" http://www.company.ru/?do=fullnew&new_id={#Первичный_ключ#}">
                                                                    {#Заголовок_новости#}</a></td>
</tr>
{##Конец_варианта_оформления_новости_1##}
{##Вариант_оформления_новости_2##}
<tr>
<td>{#Дата_публикации_новости#}</td>
<td></td>
<td><a href=" http://www.company.ru/?do=fullnew&new_id={#Первичный_ключ#}">
                                                                    {#Заголовок_новости#}</a></td>
</tr>
{##Конец_варианта_оформления_новости_2##}
{##Конец_списка_новостей##}
</table>
……  </body></html>
```

*Прил. 1. Прототип динамического документа.*

```
<html><head><title>Список последних новостей</title>……</head>
<body>……
<h1> Список новостей за Март месяц</h1>
<table>
<tr>
<td>01.03.03</td>
<td></td>
<td><a href=" http://www.company.ru/?do=fullnew&new_id=127">Смена дизайна</a></td>
</tr>
<tr>
<td>05.03.03</td>
<td><img src=http://www.company.ru/news/company/4.jpg></td>
<td><a href=" http://www.company.ru/?do=fullnew&new_id=128">Расширение ассортимента</a></td>
</tr>
<tr>
<td>10.03.03</td>
<td></td>
<td><a href=" http://www.company.ru/?do=fullnew&new_id=129">Прогноз на II квартал</a></td>
</tr>
</table>
…… </body></html>
```

*Прил. 2. Документ, сформированный на основе прототипа.*

| Столбец таблицы | Содержимое ячейки | Источник информации |
|---|---|---|
| N_ID<br>(первичный ключ) | Уникальный идентификатор новости (пресс-релиза) | Генератор последовательных номеров РБД |
| N_TYPE | Тип новости (один из четырех):<br>Новость компании<br>Новость партнера<br>Новости отрасли<br>Новость сайта | Вводится редактором через форму ввода с использованием ниспадающего меню |
| N_NAME | Краткий вариант заголовка новости, например, для узких меню | Вводится редактором через форму ввода |
| N_TITLE | Полный вариант заголовка новости, например, для оглавлений | Вводится редактором через форму ввода |
| N_SUMMARY | Резюме новости (15-25 слов) | Вводится редактором через форму ввода |
| N_CONTENT | Основной текст новости (3-4 абзаца) и заключение | Вводится редактором через форму ввода |
| N_PHOTO | Ссылка на основную иллюстрацию новости (формат JPG, GIF) | Вводятся автоматически (при загрузке файлов на сервер через форму ввода), либо вводятся редактором через форму ввода (при загрузке файлов через FTP) |
| N_ICON | Ссылка на дополнительную иллюстрацию новости, например, миниатюрную иллюстрацию для списков новостей (формат JPG, GIF) | |
| N_UPD_DATE | Дата публикации новости | Генератор даты РБД |
| N_EXP_DATE | Дата условного устаревания новости | Вычисляется автоматически, например, по формуле: N_UPD_DATE + 1 год |
| N_KEYWORDS | Список ключевых слов новости | Вводится редактором через форму ввода, через запятую |
| N_AUTHOR | Автор новости | Вводится редактором через форму ввода |
| READY | Отметка о готовности новости к публикации | Выставляется редактором через форму ввода |
| V_ID<br>(внешний ключ) | Список упоминаемых в новости компаний-партнеров | Вводится редактором через форму ввода с использованием списка V_NAME, который строится на основе таблицы VENDORS |
| P_ID<br>(внешний ключ) | Список упоминаемых в новости моделей ТСП | Вводится редактором через форму ввода с использованием списка P_NAME, который строится на основе таблицы PRODUCTS |

*Прил. 3. Типовой состав таблицы NEWS в РСУБД.*

| Столбец таблицы | Содержание ячейки | Источник информации |
|---|---|---|
| P_ID (первичный ключ) | Уникальный идентификатор технического описания продукции | Генератор последовательных номеров РБД |
| P_NAME | Краткий вариант названия ТСП (например, для узких меню) | Вводится редактором через форму ввода |
| P_TITLE | Полный вариант названия ТСП (например, для оглавлений) | Вводится редактором через форму ввода |
| P_UPD_DATE | Дата последнего обновления технического описания продукции | Генератор даты РБД |
| P_EXP_DATE | Дата условного устаревания технического описания продукции | Вычисляется автоматически, например, по формуле: P_UPD_DATE + 2 года |
| P_PHOTO | Ссылка на основную иллюстрацию технического описания продукции (формат JPG, GIF) | Вводятся автоматически (при загрузке файлов на сервер через форму ввода) либо вводятся редактором через форму ввода (при загрузке файлов через FTP) |
| P_ICON | Ссылка на дополнительную иллюстрацию технического описания продукции, например, миниатюрную иллюстрацию для оглавлений нижнего уровня (формат JPG, GIF) | |
| P_SUMMARY | Резюме технического описания продукции (20-30 слов), например, для оглавлений нижнего уровня | Вводится редактором через форму ввода |
| P_CONTENT | Основной текст технического описания продукции | Вводится редактором через форму ввода |
| P_KEYWORDS | Список ключевых слов технического описания продукции | Вводится редактором через форму ввода, через запятую |
| READY | Отметка о готовности технического описания продукции к публикации | Выставляется редактором через форму ввода |
| V_ID (внешний ключ) | Идентификатор компании-партнера (например, для извлечения логотипа компании-партнера из описания компании-партнера) | Вводится редактором через форму ввода с использованием списка V_NAME, который строится на основе таблицы VENDORS |
| F_ID (внешний ключ) | Идентификатор родителя (серии продукции), определяющий, какое оглавление серии будет содержать ссылку на данное описание продукции | Вводится редактором через форму ввода с использованием списка F_NAME, который строится на основе таблицы TREE |

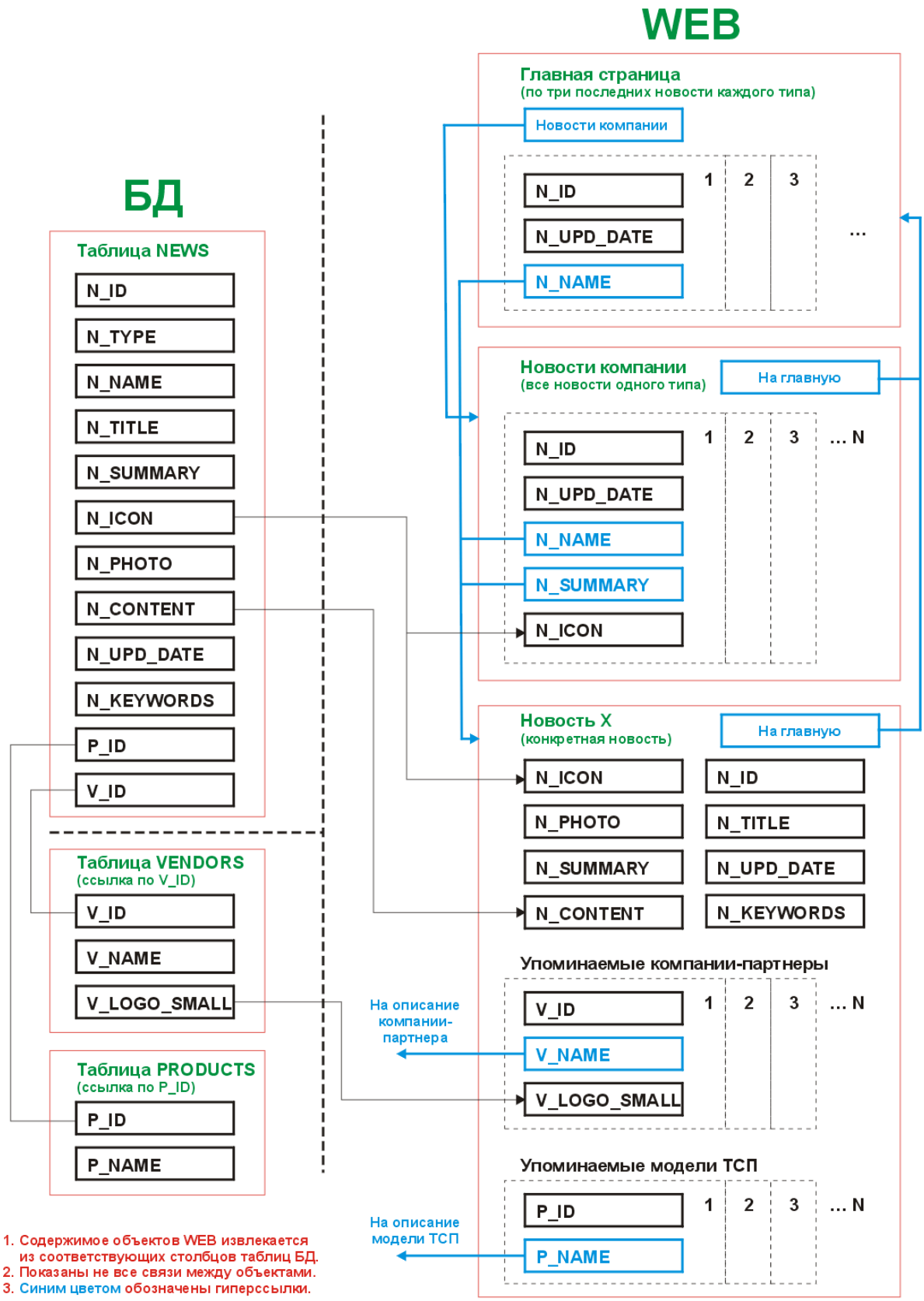
*Прил. 4. Типовой состав таблицы PRODUCTS в РСУБД.*

| Столбец таблицы | Содержимое ячейки | Источник информации |
|---|---|---|
| F_ID (первичный ключ) | Уникальный идентификатор типа или серии продукции | Генератор последовательных номеров РБД |
| F_PARENT | Уникальный идентификатор родителя – типа продукции (для серий) или пустое значение (для типов). | Выбирается редактором в ниспадающем меню формы ввода, которое строится на основе столбцов F_ID, F_NAME |
| F_NAME | Краткий вариант названия типа или серии продукции (например, для узких меню) | Вводится редактором через форму ввода |
| F_TITLE | Полный вариант названия типа или серии продукции | Вводится редактором через форму ввода |
| F_CONTENT | Описание типа или серии продукции | Вводится редактором через форму ввода |
| F_KEYWORDS | Список ключевых слов для данного типа или серии продукции | Вводится редактором через форму ввода, через запятую |
| READY | Отметка о готовности типа или серии продукции к публикации | Выставляется редактором через форму ввода |

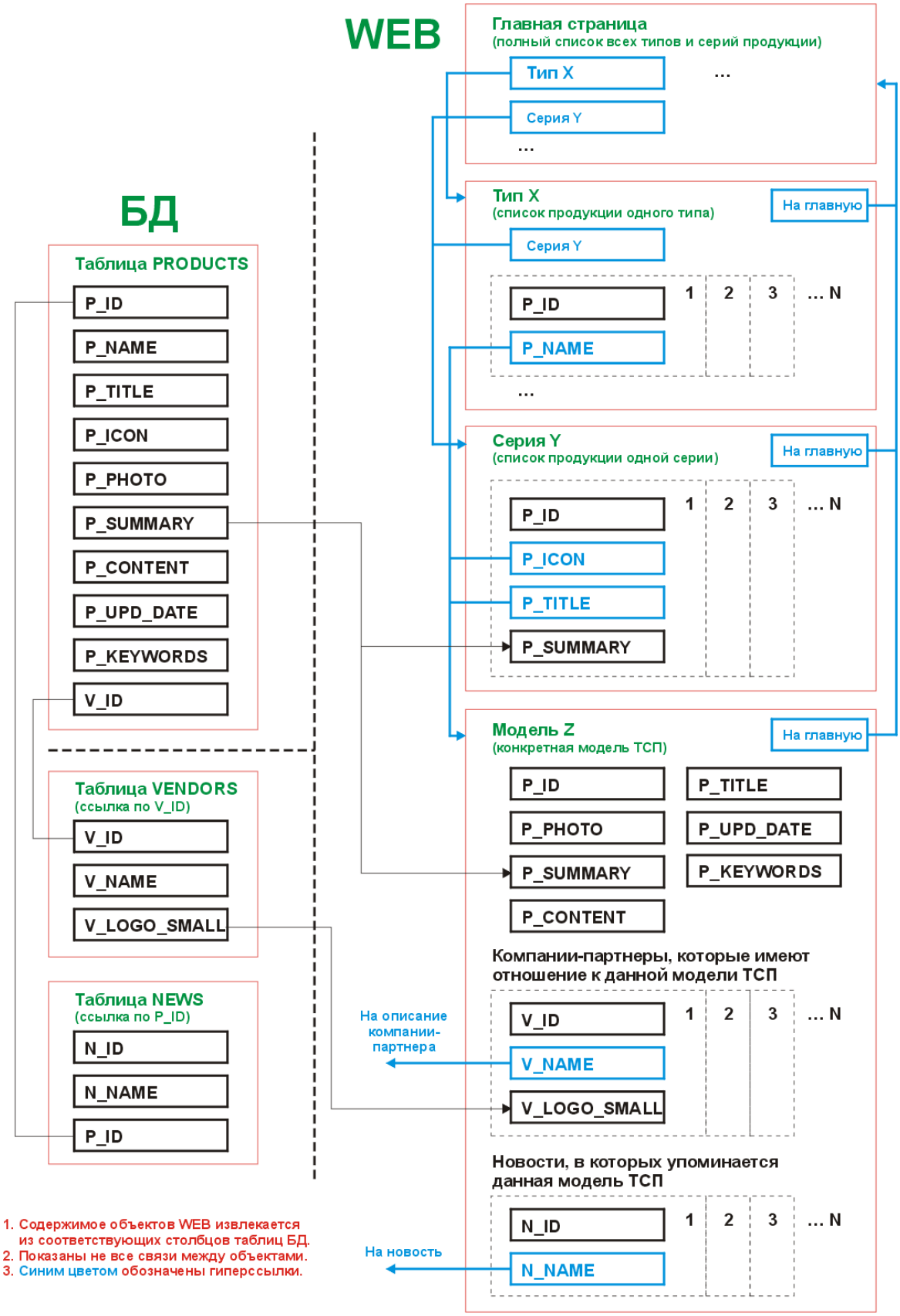
*Прил. 5. Типовой состав таблицы TREE в РСУБД.*

| Столбец таблицы | Содержимое ячейки | Источник информации |
|---|---|---|
| V_ID (первичный ключ) | Уникальный идентификатор описания компании-партнера | Генератор последовательных номеров РБД |
| V_NAME | Название компании-партнера | Вводится редактором через форму ввода |
| V_LOGO_BIG | Ссылка на логотип компании-партнера большого размера (формат JPG, GIF) | Вводятся автоматически (при загрузке файлов на сервер через форму ввода), либо вводятся редактором через форму ввода (при загрузке файлов через FTP) |
| V_LOGO_SMALL | Ссылка на логотип компании-партнера малого размера (формат JPG, GIF) | |
| V_PHOTO | Ссылка на иллюстрацию к портрету компании-партнера | |
| V_SUMMARY | Резюме портрета компании-партнера | Вводится редактором через форму ввода |
| V_CONTENT | Основной текст портрета компании-партнера | Вводится редактором через форму ввода |
| V_SITE_LINK | Ссылка на сайт компании-партнера | Вводится редактором через форму ввода |
| V_CONTACT | Контактная информация менеджера, который отвечает за работу с данной компанией-партнером | Вводится редактором через форму ввода |
| V_KEYWORDS | Список ключевых слов портрета компании-партнера | Вводится редактором через форму ввода, через запятую |
| READY | Отметка о готовности описания компании-партнера к публикации | Выставляется редактором через форму ввода |

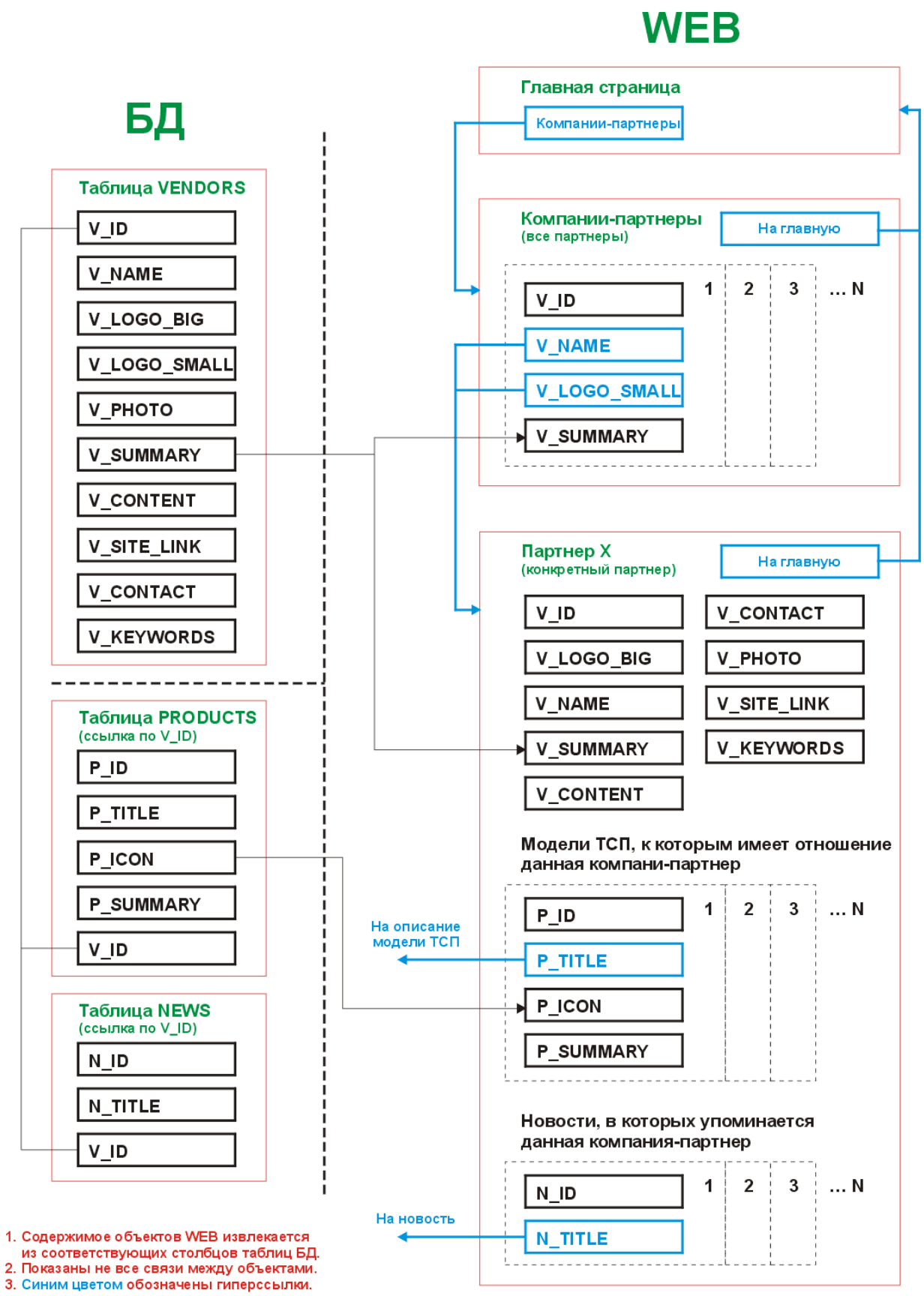
*Прил. 6. Типовой состав таблицы VENDORS в РСУБД.*




*Прил. 7. Типовая схема связей таблицы NEWS и прототипов документов.*

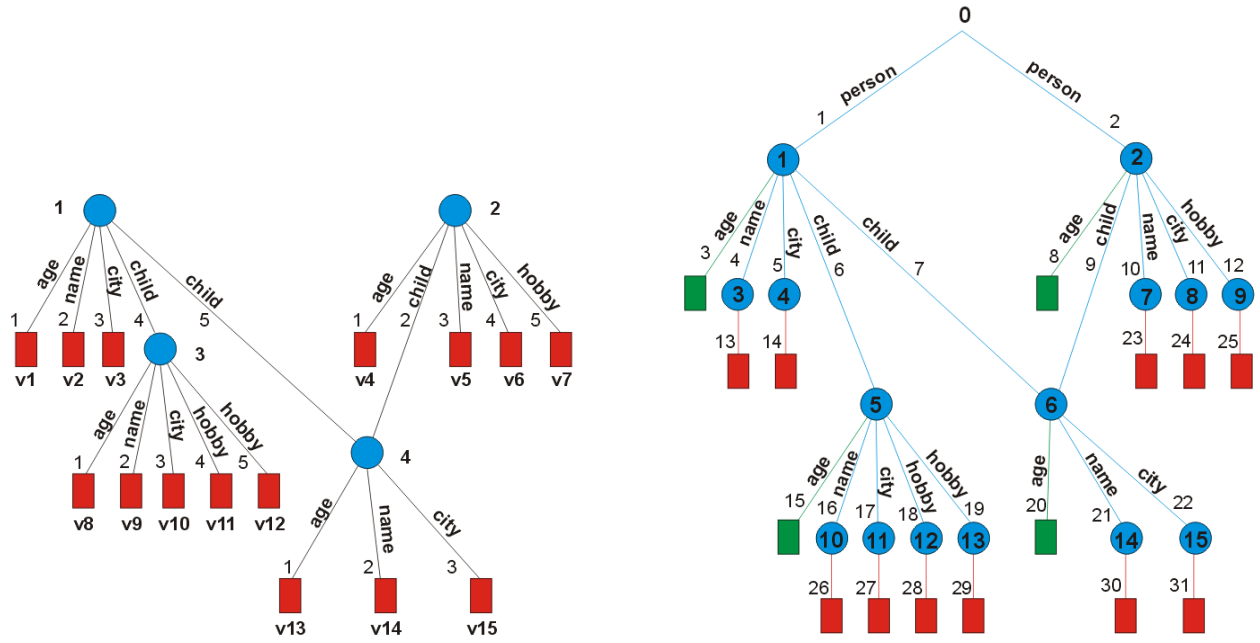



*Прил. 8. Типовая схема связей таблицы PRODUCTS и прототипов документов (поля таблицы TREE на схеме не показаны).*




*Прил. 9. Типовая схема связей таблицы VENDORS и прототипов документов.*

| | Номер значения #PCDATA | Порядковый номер дочернего элемента (атрибута) | |
|---|---|---|---|
| `<person id="1" age="55">` | **v1** | 1 | |
| `  <name>`**Иван**`</name>` | **v2** | 2 | |
| `  <city>`**Москва**`</city>` | **v3** | 3 | |
| `  <child id="3" age="22">` | **v8** | 4 | 1 |
| `  <name>`**Николай**`</name>` | **v9** | | 2 |
| `  <city>`**Казань**`</city>` | **v10** | | 3 |
| `  <hobby>`**Плавание**`</hobby>` | **v11** | | 4 |
| `  <hobby>`**Альпинизм**`</hobby>` | **v12** | | 5 |
| `  </child>` | | | |
| `  <child id="4" age="7">` | **v13** | 5 | 1 |
| `  <name>`**Ольга**`</name>` | **v14** | | 2 |
| `  <city>`**Москва**`</city>` | **v15** | | 3 |
| `  </child>` | | | |
| `</person>` | | | |
| `<person id="2" age="38" child="4">` | **v4** | 1,2 | |
| `  <name>`**Мария**`</name>` | **v5** | 3 | |
| `  <city>`**Москва**`</city>` | **v6** | 4 | |
| `  <hobby>`**Рисование**`</hobby>` | **v7** | 5 | |
| `</person>` | | | |




*Прил. 10. XML-документ и соответствующие ему графы. Слева – граф документа согласно [53], справа – модернизированный граф.*

| Edge | | | | |
|---|---|---|---|---|
| **source** | **ordinal** | **name** | **flag** | **target** |
| 1 | 1 | age | число | v1 |
| 1 | 2 | name | строка | v2 |
| 1 | 3 | city | строка | v3 |
| 1 | 4 | child | ссылка | 3 |
| 1 | 5 | child | ссылка | 4 |
| … | … | … | … | … |

| $V_{int}$ | |
|---|---|
| **vid** | **value** |
| v1 | 55 |
| v4 | 38 |
| v8 | 22 |
| v13 | 7 |

| $V_{string}$ | |
|---|---|
| **vid** | **value** |
| v2 | Иван |
| v3 | Москва |
| v5 | Мария |
| v6 | Москва |
| … | … |
| v15 | Москва |

*Прил. 11. Таблица Edge и отдельные таблицы значений ($V_{int}$ и $V_{string}$).*

| Universal | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **source** | **…** | **ordinal (name)** | **flag (name)** | **target (name)** | **…** | **ordinal (hobby)** | **flag (hobby)** | **target (hobby)** |
| 1 | … | 2 | строка | v2 | … | null | null | null |
| 1 | … | 2 | строка | v2 | … | null | null | null |
| 2 | … | 3 | строка | v5 | … | 5 | строка | v7 |
| 3 | … | 2 | строка | v9 | … | 4 | строка | v11 |
| 3 | … | 2 | строка | v9 | … | 5 | строка | v12 |
| 4 | … | 2 | строка | v14 | … | null | null | null |

*Прил. 12. Таблица Universal.*

| Element | | |
|---|---|---|
| **path** | **value** | **Parent** |
| .person0 | null | root |
| .person0.name0 | Иван | .person0 |
| .person0.city0 | Москва | .person0 |
| .person0.child0 | null | .person0 |
| .person0.child0.name0 | Николай | .person0.child0 |
| … | … | … |
| .person1.hobby0 | Рисование | .person1 |

| Attribute | | |
|---|---|---|
| **path** | **name** | **value** |
| .person0 | id | 1 |
| .person0 | age | 55 |
| .person0.child0 | id | 3 |
| .person0.child0 | age | 22 |
| … | … | … |
| .person1 | id | 2 |
| .person1 | age | 38 |
| .person1 | child | 4 |

*Прил. 13. Таблицы Element и Attribute для метода Path.*

| Edge | | | | | |
|---|---|---|---|---|---|
| **id** | **document_id** | **parent_id** | **element_id** | **order_num** | **name** |
| 1 | family1 | 0 | 1 | 1 | person |
| 2 | family1 | 0 | 2 | 2 | person |
| 3 | family1 | 1 | null | 1 | age |
| 4 | family1 | 1 | 3 | 2 | name |
| 5 | family1 | 1 | 4 | 3 | city |
| 6 | family1 | 1 | 5 | 4 | child |
| 7 | family1 | 1 | 6 | 5 | child |
| 8 | family1 | 2 | null | 1 | age |
| 9 | family1 | 2 | 6 | 2 | child |
| … | … | … | … | … | … |
| 30 | family1 | 14 | null | 1 | null |
| 31 | family1 | 15 | null | 1 | null |

| Value | |
|---|---|
| **id** | **data** |
| 1 | null |
| 2 | null |
| 3 | 55 |
| 4 | null |
| 5 | null |
| 6 | null |
| 7 | null |
| 8 | 38 |
| 9 | null |
| … | … |
| 30 | Ольга |
| 31 | Москва |

*Прил. 14. Таблицы Edge и Value в методе Edge Distributive.*

| Tag_5 | | | | |
|---|---|---|---|---|
| **Id** | **document_id** | **parent_id** | **element_id** | **order_num** |
| 6 | family1 | 1 | 5 | 4 |
| 7 | family1 | 1 | 6 | 5 |
| 9 | family1 | 2 | 6 | 2 |

| Tag_list | |
|---|---|
| **tag_name** | **table_name** |
| person | tag_1 |
| age | tag_2 |
| name | tag_3 |
| city | tag_4 |
| child | tag_5 |
| hobby | tag_6 |

| Tag_0 | | | | |
|---|---|---|---|---|
| **Id** | **document_id** | **parent_id** | **element_id** | **order_num** |
| 13 | family1 | 3 | null | 1 |
| 14 | family1 | 4 | null | 1 |
| 23 | family1 | 7 | null | 1 |
| 24 | family1 | 8 | null | 1 |
| 25 | family1 | 9 | null | 1 |
| 26 | family1 | 10 | null | 1 |
| … | … | … | … | … |
| 31 | family1 | 15 | null | 1 |

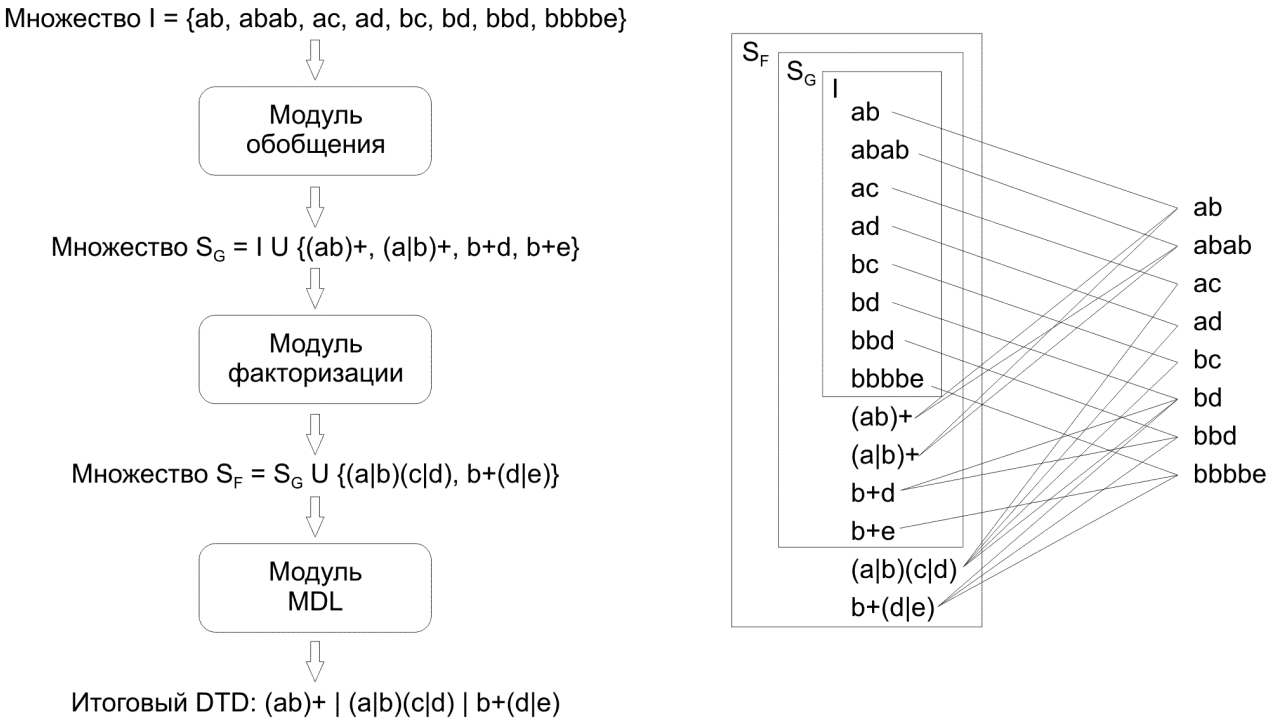
| Value | |
|---|---|
| **id** | **data** |
| 1 | null |
| 2 | null |
| 3 | 55 |
| 4 | null |
| 5 | null |
| 6 | null |
| 7 | null |
| 8 | 38 |
| 9 | null |
| … | … |
| 30 | Ольга |
| 31 | Москва |

*Прил. 15. Таблицы Tag_list, Tag_5, Tag_0 и Value для метода Binary Distributive.*

| Element | | | |
|---|---|---|---|
| **document_id** | **path** | **data** | **parent_path** |
| family1 | .person_#0 | null | . |
| family1 | .person_#1 | null | . |
| family1 | .person_#0.name_#0 | Иван | .person_#0 |
| family1 | .person_#0.city_#0 | Москва | .person_#0 |
| family1 | .person_#0.child_#0 | null | .person_#0 |
| family1 | .person_#0.child_#1 | null | .person_#0 |
| … | … | … | … |
| family1 | .person_#0.child_#1.city_#0 | Москва | .person_#0.child_#1 |

| Attribute | | |
|---|---|---|
| **path** | **name** | **data** |
| .person_#0 | id | 1 |
| .person_#0 | age | 55 |
| .person_#1 | id | 2 |
| .person_#1 | age | 38 |
| .person_#1 | child | 4 |
| .person_#0.child_#0 | id | 3 |
| .person_#0.child_#0 | age | 22 |
| .person_#0.child_#1 | id | 4 |
| .person_#0.child_#1 | age | 7 |

*Прил. 16. Таблицы Element и Attribute для модернизированного метода Path.*




*Прил. 17. Архитектура системы генерирования DTD.*

**процедура** FACTORSUBSETS($S_G$)
**начало**
1. **для каждого** выражения D из $S_G$
2. вычислить *рейтинг(D, $S_G$)*
3. $S_F$ := S' := $S_G$; SeedSet := Ø
4. **для** i := от 1 до k
5. пусть D – выражение из S' с максимальным значением *рейтинг(D,S')*
6. SeedSet := SeedSet ∪ D
7. S' := S' – {D': *перекрытие(D,D')* ≥ δ}
8. **для каждого** выражения D в множестве SeedSet
9. S := {D}
10. S' := $S_G$ – {D': *перекрытие(D,D')* ≥ δ}
11. **пока** S' не станет пустым
12. пусть D' – выражение из S' с макс. значением *рейтинг(D',S)*
13. S := S ∪ D'
14. S' = S' – {D'': *перекрытие(D',D'')* ≥ δ}
15. F := FACTOR(S) /* примечание: F = $F_1$ | $F_2$ | … | $F_m$ */
16. $S_F$ := $S_F$ ∪ {$F_1$, $F_2$, …$F_m$}

**конец**

*Прил. 18. Процедура FACTORSUBSETS.*

```
процедура GENERALIZE(I)
начало
1.      для каждой последовательности s в I
2.              добавить s в S_G
3.              для r := 2,3,4
4.                      s' := DISCOVERSEQPATTERN(s,r)
5.                      для d := 0,1*|s'|,0,5*|s'|,|s'|
6.                              s'' := DISCOVERORPATTERN(s',d)
7.                              добавить s'' в S_G
конец
процедура DISCOVERSEQPATTERN(s,r)
начало
1.      повторять
2.              пусть x – подпоследовательность в s, которая последовательно
3.                      повторяется максимальное число раз (≥ r)
4.              заменить все последовательные повторения x в s
5.                      новым подстановочным символом A_i = (x)*
6.      до тех пор, пока (s не содержит ≥ r последовательных повторений
7.              любой подпоследовательности x)
8.      вернуть s
конец
процедура DISCOVERORPATTERN(s,d)
начало
1.      s_1, s_2, …, s_n := PARTITION(s,d)
2.              для каждой подпоследовательности s_j из s_1, s_2, …, s_n
3.                      пусть набор различных символов в s_j – это a_1, a_2, …, a_m
4.                      если (m>1)
5.                              заменить подпоследовательность s_j в последовательности s
6.                              новым подстановочным символом A_i = (a_1|a_2|…|a_m)*
7.      вернуть s
конец
процедура PARTITION(s,d)
начало
1.      i := start := end := 1
2.      s_i = s[start, end]
3.      пока (end < |s|)
4.              пока (end < |s| и любой символ из s_i встречается справа от s_i
5.                              на расстоянии от этого символа, меньше или равном d)
6.                      end := end+1;
7.                      s_i := s[start, end];
8.              если (end < |s|)
9.                      i := i+1;
10.                     start := end+1;
11.                     end := end+1;
12.                     s_i := s[start, end];
13.     вернуть s_1, s_2, …, s_i
конец
```

*Прил. 19. Алгоритм обобщения.*

**процедура** FACTOR(S)  /* S – множество выражений, который нужно факторизовать */
**начало**
1. DivisorSet := FINDALLDIVISORS(S)
2. **если** (DivisorSet = Ø)
3. **вернуть** дизъюнкцию (логическое ИЛИ) выражений из S
4. DivisorList := Ø
5. **для каждого** делителя V из множества DivisorSet
6. Q, R := DIVIDE(S,V)
7. добавить (V, Q, R) в множество DivisorList
8. найти самый короткий триплет ($V_i$, $Q_i$, $R_i$) в множестве DivisorList
9. **вернуть** (FACTOR($V_i$)FACTOR($Q_i$) | FACTOR($R_i$))

**конец**

**процедура** FINDALLDIVISORS(S)
**начало**
1. DivisorSet := Ø
2. **для каждой** последовательности символов s, такой, что s – суффикс
3. для двух или более выражений в наборе S
4. DivisorSet := DivisorSet ∪ {{p:ps принадлежит S}}
5. вернуть DivisorSet

**конец**

**процедура** DIVIDE(S,V)
**начало**
1. **для каждой** последовательности символов p из множества V
2. $q_p$ := {s: ps принадлежит S}
3. Q := пересечение всех $q_p$ для p, принадлежащих V
4. R := S – V º Q
/* Примечание: V º Q – это множество последовательностей, полученных путем присоединения каждой последовательности из Q справа к каждой последовательности из V */
5. **вернуть** Q, R

**конец**

*Прил. 20. Алгоритм факторизации.*

# Глоссарий

**ИФТИ** – Институт физико-технической информатики

**МФТИ** – Московский физико-технический институт

**ОС** – операционная система

**ПО** – программное обеспечение

**РСУБД** – реляционная система управления базами данных

**СУБД** – система управления базами данных

**API** – Application Programming Interface – набор функций, предоставляемый для использования в прикладных программах

**CALS** – первоначально Computer Aided Logistic Support, в настоящее время Continuous Acqusition and Life cycle Support, система управления электронной документацией и обмена информацией о технологически сложном изделии на всех этапах его жизненного цикла

**CRM** – Customer Relationship Management, учет контактов с клиентами

**CSS** – Cascading Style Sheets, каскадные таблицы стилей

**DTD** – Document Type Definition, определение типа документа

**ERP** – Enterprise Resources Planning, планирование ресурсов предприятия

**HTML** – HyperText Markup Language, язык разметки гипертекста

**IEEE** – Institute of Electrical and Electronics Engineers, Институт инженеров по электротехнике и электронике

**IIS** – Internet Information Services (серверное ПО компании Microsoft)

**NSF** – National Science Foundation, Национальный научный фонд США

**PLCS** – Product Life Cycle Support, то же, что CALS

**PLM** – Product Life Management, то же, что CALS

**BPR** – Business Process Reengineering, ре-инжиниринг бизнес-процессов

**SCM** – Supply Chain Management, управление цепочками поставок

**SRM** – Supplier Relationship Management, то же, что SCM

**SGML** – Standard Generalized Markup Language, стандартный обобщенный язык разметки

**URL** – Uniform Resource Locator, "адрес" документа в сети Internet (например, http://www.mail.ru/index.htm).

**XML** – Extensible Markup Language, расширенный язык разметки

# Список литературы

**Публикации автора по теме диссертации**

[1] Леонов А. В. Динамический документ – ключевой объект современных информационных систем // Сборник трудов 3-й международной конференции VEonPC'2003 "Системы виртуального окружения на Linux-кластерах персональных компьютеров". – М., 2003. – С. 150-169.

[2] Леонов А. В., Бахбух Б. М., Лудинов В. В., Петренко И. И. Публикация динамических документов рекламно-информационного характера на корпоративном сайте // Исследовано в России. – 2003. – С. 1148-1185.

[3] Леонов А. В., Бахбух Б. М. Построение корпоративной сети малого или среднего предприятия с использованием операционной системы Linux // Сборник трудов 3-й международной конференции VEonPC'2003 "Системы виртуального окружения на Linux-кластерах персональных компьютеров". – М., 2003. – С. 141-149.

[4] Леонов А. В., Хуснутдинов Р. Р. Построение оптимальной реляционной схемы для хранения XML документов в РСУБД без использования DTD / XML Schema // Программирование. – 2004. – N 6. – С. 30-48.
Leonov A., Khusnutdinov R. Construction of an Optimal Relational Schema for Storing XML Documents in RDBMS without Using DTD/XML Schema // Programming and Computer Software. – 2004. – N 6 (30). – P. 323-336.

[5] Леонов А. В., Хуснутдинов Р. Р. Исследование и разработка системы генерирования DTD для XML-документов // Программирование. – 2005. – N 4. – принята в печать редколлегией журнала.
Леонов А. В., Хуснутдинов Р. Р. Исследование и разработка системы генерирования DTD для XML-документов // Исследовано в России. – 2004. – С. 2515-2537.

[6] Байгозин Д. А., Батурин Ю. М., Гёбель М., Клименко С. В., Леонов А. В., Никитин И. Н., Никитина Л. Д. Интерактивное повествование в виртуальном окружении: обучающая система "Виртуальный Планетарий" // Вычислительные методы и программирование. – 2004. – Т. 5. – N 2. – С. 192-205.

**Книги, публикации, отчеты**

[7] Баричев С. Г., Серов Р. Е. Основы современной криптографии. – М.: Горячая линия - Телеком, 2002. – 175 с.

[8] Бауэр Ф., Гооз Г. Информатика / Пер. с нем. – М.: Мир, 1976. – 484 с.

[9] Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем. – СПб: Питер, 2000. – 384 с.

[10] Электронные документы в корпоративных сетях / С. В. Клименко, И. В. Крохин, В. М. Кущ, Ю. Л. Лагутин. – М.: Эко-Трендз, 1999. –

271 с.

[11] Кнут Д. Е. Все про TeX / Пер. с англ. – Протвино: АО RDTeX, 1993. – 592 с.

[12] Марчук Ю. Н. Основы компьютерной лингвистики. – М.: Сигнал, 1999. – 265 с.

[13] Новак Л. Г., Кузнецов С. Д. Свойства схем данных XML // Труды Института системного программирования РАН. – М., 2003. http://utc.jinr.ru/internet/xml/xml_sv/XML_sv.shtml.htm

[14] Плискин Е. Л. Управление версиями в системах коллективного создания документов // Сборник трудов ИСА РАН "Развитие безбумажной технологии в организационных системах". – М., 1999.

[15] Холзнер С. Perl: специальный справочник / Пер. с англ. – СПб: Питер, 2000. – 640 с.

[16] Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Пер. с англ. – М.: Триумф, 2002. – 816 с.

[17] Шеннон К. Математическая теория связи / Пер. с англ. В. Ф. Писаренко. – В кн.: Шеннон К. Работы по теории информации и кибернетике. – М.: ИЛ, 1963. – 832 с.

[18] Lannon J. M. Technical Writing. 7th ed. – New York: Longman, 1996.

[19] Göbel M. et al. On Creating Virtual Reality Stories And Interactive Experiences // Proc. GraphiCon. – 2000.

[20] McCormick B. H., DeFanti T. A., Brown M. D. Visualization in Scientific Computing // Computer Graphics. – 1987. – Vol. 21. – No. 6.

[21] Aukstakalnis S., Blatner, D. Silicon Mirage – The Art and Science of Virtual Reality. – Berkeley, CA: Peachpit Press. – 1992.

[22] Milgram P., Kishino F. A Taxonomy of Mixed Reality Visual Displays // IEICE Transactions on Information Systems E77-D (12). – 1994.

[23] Tramberend H. Avango: A Distributed Virtual Reality Framework // Proc. IEEE Virtual Reality 1999. – 1999.

[24] Klimenko S., Nikitin I., Burkin V., Göbel M., Hasenbrink F., Tramberend H. Virtual Planetarium in CyberStage // Proc. 6th Eurographics Workshop on Virtual Environments. – 2000.

[25] Klimenko S., Nielson G., Nikitina L., Nikitin I., Strassner J. Virtual Planetarium: Learning Astronomy in Virtual Reality // Proc. ED-MEDIA'2004. – 2004.

[26] Abe Y., Suzuki J., Tashiro G., Yamamoto Y. Persona: a Framework to provide Adaptive Presentation for Web Documents. – IPSJ Summer Programming Symposium.

[27] Adams K. C. The Web as Database: New Extraction Technologies and Content Management. – ONLINE, March 2001.

[28] Brin S., Page L. The Anatomy of a Large-Scale Hypertextual Web Search Engine. – Computer Science Department, Stanford University. – 1998.

[29] Chakrabarti S., van den Berg M., Dom B. Focused crawling: a new approach to topic-specific Web resource discovery. – Amsterdam, Netherlands: Computer Networks. – 1999.

[30] Coffman K. G., Odlyzko A. M. Growth of the Internet // Optical Fiber Telecommunications IV B: Systems and Impairments, I. P. Kaminow and T. Li, eds. – Academic Press. – 2002. – P. 17-56.

[31] Cooper M., Foote J. Automatic Music Summarization via Similarity Analysis. – 2002.

[32] Costello D. For Knowledge, Look Within // Knowledge Management Magazine. – September 2000.

[33] Fikes R., Jenkins J., Frank, G. JTP: A System Architecture and Component Library for Hybrid Reasoning. Knowledge Systems Laboratory. – 2003.

[34] Harney J. Predictive analytics: forecasting future trends from existing data. – KMWorld Magazine. – January 2003. – Volume 12. – Issue 1.

[35] Hutchins W. J. Machine translation: past, present, future. – Chichester (UK): Ellis Horwood. – 1986. – 382 p.

[36] Hutchins W. J. Machine translation today and tomorrow // Computerlinguistik: was geht, was kommt? Festschrift für Winfried Lenders, hrsg. Gerd Willée, Bernhard Schröder, Hans-Christian Schmitz. – Sankt Augustin: Gardez! Verlag. – 2002. – P. 159-162.

[37] Khoussainov R., Kushmerick N. Optimising Performance of Competing Search Engines in Heterogeneous Web Environments. – 2003.

[38] Kitsuregawa M., Wang Y. Link Based Clustering of Web Search Results. – 2001.

[39] Klusch M., Bergamaschi S., Edwards P., Petta, P. Intelligent Information Agents: The AgentLink Perspective. – Austrian Research Institute for Artificial Intelligence, Vienna, Austria. – 2003.

[40] Koulopoulos T. M., Frappaolo C. Electronic Document Management Systems: A Portable Consultant. – New York: McGraw-Hill, Inc. – 1995.

[41] Lahtinen T. Automatic indexing: an approach using an index term corpus and combining linguistic and statistical methods. – 2000.

[42] Mani I. Automatic Summarization. – Amsterdam, The Netherlands: John Benjamins Publishing Co. – 2001. – 285 p.

[43] Mitkov R. Anaphora resolution. – Longman. – 2002.

[44] Ngo C.W., Pong T.C., Zhang H.J. Recent Advances in Content Based Video Analysis. – International Journal of Image and Graphics. – 2002.

[45] Prior C. Workflow and Process Management. – 2003.

[46] Rüger S. M., Zervas G. The Curse of Dimensionality and Document

Clustering. – 1999.

[47] Saggion H. Automatic Abstracting: towards a Text Based Generation.

[48] Sutton M. J. D. Document Management for the Enterprise: Principles, Techniques, and Applications. – New York: Wiley Computer Publishing. – 1996.

[49] Williamson B., Miller L. The semantic web: A touch of intelligence for the internet? – 2003.

[50] Woods E. Knowledge management 2002-2003: the end of the beginning. – KMWorld Magazine. – January 2003. – Volume 12. – Issue 1.

[51] Wooldridge M., Jennings N. Intelligent Agents: Theory and Practice. – Knowledge Engineering Review. – June 1995. – No 2. – Volume 10.

[52] Zha H., Ji X. Poster session: Correlating multilingual documents via bipartite graph modeling // Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval. – 2002.

[53] Florescu D., Kossmann D. A performance evaluation of alternative mapping schemes for storing XML data in a relational database // Rapport de Recherche No. 3680 INRIA, Rocquencourt, France. – May 1999.

[54] Florescu D., Kossmann D. Storing and querying XML data using an RDBMS // Bulletin of the IEEE Computer Society Technical Committee on Data Engineering. – 1999.

[55] Chen Q., Khan L, Rao Y. A Comparative Study of Storing XML Data in Relational and Object-Relational Database Management Systems // Proceedings of the International Conference on Internet Computing. – Las Vegas, Nevada, USA. – 2002. – P. 277–282.

[56] Zwol R. v., Apers P., Wilschut A.. Modelling and querying semistrictured data with MOA // Workshop on Query processing for semistructured data and non-standard data formats. – 1999.

[57] Garofalakis M., Gionis A., Rastogi R., Seshadri S., Shim K. XTRACT: Learning Document Type Descriptors from XML Document Collections // Data Mining and Knowledge Discovery. – 2003. – N 7. – P. 23-56.

[58] Shafer K. E. Creating DTDs via the GB-Engine and Fred. – 1995.

[59] Brazma A. Efficient Identification of regular expressions from representative examples // Proc. of the Ann. Conf. on Computational Learning Theory (COLT). – 1993.

[60] Kilpelainen P., Mannila H., Ukkonen E. MDL learning of unions of simple pattern languages from positive examples // Proc. of the European Conf. on Computational Learning Theory (EuroCOLT). – 1995.

[61] Fernandez M., Suciu D. Optimizing regular path expressions using graph schemas // Proc. of the Intl. Conf. on Database Theory (ICDT). – 1997.

[62] Goldman R., Widom J. DataGuides: Enabling query formulation and optimization in semistructured databases // Proc. of the Intl. Conf. on Very Large Data Bases (VLDB). – 1997.

[63] Nestorov S., Abiteboul S., Motwani R. Extracting schema from semistructured data // Proc. of the ACM SIGMOD Intl. Conf. on Management of Data. – 1998.

[64] Brayton R. K., McMullen C. The Decompozition and Factorization of Boolean Expressions // Proc. Of the Intl. Symp. On Circuits and Systems. – 1982.

[65] Charikar M., Guha S. Improved Combinatorial Algorithms for the Facility Location and K-median Problem // Proc. of the Ann. Symp. on Foundations of Computer Science (FOCS). – 1999.

[66] Hochbaum D. S. Heuristics for the Fixed Cost Median Problem // Mathematical Programming. – 1982. – N 22. – P. 148-162.

[67] Hopcroft J. E., Ullman J. D. Introduction to Automation Theory, Languages and Computation. – Reading, Massachusetts: Addison-Wesley. – 1979.

[68] Gutmann P. Encryption and Security Tutorial: http://www.cs.auckland.ac.nz/~pgut001/tutorial/index.html

[69] Lyman P., Varian H. R. How much information? Internet – summary. http://www.sims.berkeley.edu/research/projects/how-much-info/internet.html

[70] AIIM User Guide. Business Process Management and Workflow. http://www.aiim.org/inform/all_docrep.asp

[71] AIIM User Guide. Document and Content Capture. http://www.aiim.org/inform/all_docrep.asp

[72] CSC Index Research and Advisory Services. Foundation Report 112: Workflow and Business Process Design Tools. http://www.cscresearchservices.com/foundation/library/112/RP01.asp

[73] Delphi White Paper. The Document is the Process. http://www.delphigroup.com/pubs/whitepapers/

[74] The UCLA Internet Report. Surveying the Digital Future. Year Three. – UCLA Center for Communication Policy. – 2003. http://www.ccp.ucla.edu/pages/internet-report.asp

[75] Wang A. R. R. Algorithms for Multi-Level Logic Optimization. PhD Thesis, Univ. of California, Berkeley. – 1989.

**Сайты научных сообществ**

[76] IEEE Computer Society: http://computer.org/

[77] IEEE Communication Society: http://comsoc.org/

[78] IEEE Professional Communication Society: http://ieeepcs.org/

[79] American Association for Artificial Intelligence: http://www.aaai.org/

[80] American Society for Information Science and Technology: http://www.asis.org/

[81] Association for Computing Machinery: http://portal.acm.org/

[82] Assosiation of Knowledgework: http://www.kwork.org/

[83] Международный семинар Диалог: http://www.dialog-21.ru/

**Сайты конференций**

[84] International Conference on Advances in Computer Entertainment Technology (ACE 2004): http://www.ace2004.org/

[85] 4th International Conference on Computational Semiotics for Games and New Media (COSIGN 2004): http://www.cosignconference.org/

[86] ACM Collaborative Virtual Environments (CVE 2002): http://www.cve2002.org/cve2002-storytelling.html

[87] World Conference on Educational Multimedia, Hypermedia and Telecommunications (ED-MEDIA 2004): http://www.aace.org/conf/edmedia/

[88] 10th Eurographics Symposium on Virtual Environments (EGVE 2004): http://www.eg.org/EG/DL/WS/EGVE/EGVE04

[89] 3nd International Conference on Entertainment Computing (ICEC 2004): http://www.icec.id.tue.nl/

[90] 2nd International Conference on Virtual Storytelling (ICVS 2003): http://www.virtualstorytelling.com/ICVS2003/

[91] 3rd International Conference for Narrative and Interactive Learning Environment (NILE 2004): http://computing.unn.ac.uk/staff/cgpb4/nile/

[92] 2nd International Conference on Technologies for Interactive Digital Storytelling and Entertainment (TIDSE 2004): http://www.zgdv.de/TIDSE04/

[93] 3rd International Conference VEonPC (VEonPC 2003): http://viswiz.imk.fraunhofer.de/VEonPC/2003/

[94] IEEE International Virtual Reality Conference (VR 2002): http://www.hoise.com/primeur/02/articles/monthly/AE-PR-02-02-36.html

[95] 6th Virtual Reality International Conference (VRIC 2004): http://www.laval-virtual.org/en/appel-colloque.php

[96] 9th International Conference on Virtual Systems and MultiMedia (VSMM 2003): http://www.vsmm.org/2003/

**Корпоративные сайты**

[97] 1С: http://www.1c.ru/

[98] Арсеналъ: http://www.ars.ru/

[99] Галактика: http://www.galaktika.ru/

[100] ЛАНИТ: http://www.lanit.ru/

[101] МедиаЛингва: http://www.medialingua.ru/
[102] НИЦ CALS-технологий "Прикладная Логистика": http://www.cals.ru
[103] Оптима: http://www.optima.ru/
[104] Парус: http://www.parus.ru/
[105] ПРОМТ: http://www.promt.ru/
[106] Телеком Транспорт: http://www.tt.ru
[107] ЭОС: Электронные офисные системы: http://eos.ru/eos/
[108] ABBYY: http://www.abbyy.ru/
[109] Adobe: http://www.adobe.com/
[110] Baan: http://www.baan.com/
[111] Borland Software Corporation: http://www.borland.com/
[112] Cognitive Technologies: http://www.cognitive.ru/
[113] Convera: http://www.convera.com/
[114] Delphi Group: http://www.delphigroup.com/
[115] Digital Design: http://www.digdes.ru/
[116] Divine: http://www.divine.com/
[117] Document Scanners: http://www.highspeedscanner.com/
[118] Documentum: http://www.documentum.ru/
[119] Gartner: http://www.gartner.com/
[120] Hummingbird: http://www.hummingbird.ru/
[121] Hyperwave: http://www.hyperwave.com/
[122] i2 Technologies: http://www.i2.com/
[123] IBM: http://www.ibm.com/
[124] IDC: http://www.idc.com/
[125] I.R.I.S.: http://www.irislink.com/
[126] J. D. Edwards: http://www.jdedwards.com/
[127] Macromedia: http://www.macromedia.com/
[128] META Group: http://www.metagroup.com/
[129] Microsoft: http://www.microsoft.com/
[130] Microsystems: http://www.analyst.ru/
[131] MOVES Institute: http://www.movesinstitute.org/
[132] Novell: http://www.novell.com/
[133] Open Text Corporation: http://www.opentext.com/
[134] Oracle: http://www.oracle.com/
[135] Ovum: http://www.ovum.com/
[136] PeopleSoft: http://www.peoplesoft.com/
[137] Sage Group: http://www.sage.com/

[138] SAP: http://www.sap.com/
[139] Scala: http://www.scala.net/
[140] ScanSoft: http://www.scansoft.com/
[141] Siebel Systems: http://www.siebel.com/
[142] Sun Microsystems: http://www.sun.com/
[143] Sybase: http://www.sybase.org/
[144] Verity: http://www.verity.com/
[145] Vignette: http://www.vignette.com/

**Сайты стандартов, технологий и программных продуктов**

[146] Официальная страница Allora: http://www.hitsw.com/products_services/xml_platform/allora_dsheet.html
[147] Официальная страница AlphaWorks: Data Descriptors by Example: http://www.alphaworks.ibm.com/tech/DDbE
[148] Официальный сайт разработчиков Apache: http://www.apache.org.
[149] Официальный сайт ASP: http://www.asp.net/
[150] Официальная страница Autonomy: http://www.autonomy.com/Content/Technology/
[151] Официальная страница Avango: http://www.avango.org/
[152] Официальный сайт HTML: http://www.w3.org/MarkUp/
[153] Официальная страница IBM DB2 XML Extender: http://www-306.ibm.com/software/data/db2/extenders/xmlext/index.html.
[154] Официальная страница Java 2 Standard Edition: http://java.sun.com/j2se/
[155] Knowledge Markup Language (KML) home page: http://kml.mipt.ru/
[156] Официальная страница FOR XML EXPLICIT: http://msdn.microsoft.com/library/periodic/period01/xmlExplicit.htm.
[157] Официальный сайт разработчиков MySQL: http://www.mysql.org/
[158] Официальная страница MySQL: Open Source Relational Database Management System: http://www.mysql.com.
[159] Официальная страница Oracle XML SQL Utilities: http://otn.oracle.com/tech/xml/index.html.
[160] Официальный сайт разработчиков Perl: http://www.perl.com.
[161] Perl Template Toolkit Home Page: http://template-toolkit.org/
[162] Библиотека модулей для Perl: http://perl.cpan.org/.
[163] Официальный сайт разработчиков PHP: http://www.php.net.
[164] PHP Smarty Template Engine: http://smarty.php.net/
[165] Официальный сайт разработчиков PostgreSQL: http://www.postgresql.org.
[166] Resource Description Framework (RDF):

http://www.w3.org/RDF/Overview.html
[167] Официальная страница SAXON: http://sourceforge.net/projects/saxon
[168] Overview of SGML Resources: http://www.w3.org/MarkUp/SGML/
[169] Официальная страница Sybase Inc. – XML & Web-Services: http://www.sybase.com/products/databaseservers/ase/javaxml
[170] TeX Users Group (TUG) home page: http://www.tug.org/
[171] Официальный сайт XML: http://www.xml.org/
[172] Спецификация XML 1.0: http://www.w3.org/TR/REC-xml
[173] Официальная страница XML Spy: http://www.xmlspy.com/

**Другие ресурсы сети Интернет**

[174] The Internet Operating System Counter: http://www.leb.net/hzo/ioscount/index.html.
[175] Netcraft Web Server Survey: http://www.netcraft.com.
[176] Netstat: http://www.netstat.ru.
[177] Augmented Reality Homepage: http://www.augmented-reality.org
[178] Augmented Reality Page: http://www.se.rit.edu/~jrv/research/ar/
[179] Jaron Lanier's Homepage: http://www.advanced.org/jaron/
[180] Глоссарий.Ру: http://www.glossary.ru
[181] Федеральный закон РФ "Об информации, информатизации и защите информации" от 20.02.1995 № 24-ФЗ