На правах рукописи

ГУСАРЕНКО Артем Сергеевич

ОБРАБОТКА XML-ДОКУМЕНТОВ В СИТУАЦИОННО-ОРИЕНТИРОВАННЫХ БАЗАХ ДАННЫХ НА ОСНОВЕ ДИНАМИЧЕСКИХ DOM-ОБЪЕКТОВ

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

АВТОРЕФЕРАТ
диссертации на соискание ученой степени кандидата технических наук

Уфа – 2013

Работа выполнена
на кафедре автоматизированных систем управления
ФГБОУ ВПО «Уфимский государственный авиационный технический университет»

Научный руководитель	д-р техн. наук, проф. **МИРОНОВ Валерий Викторович**
Официальные оппоненты	д-р техн. наук, проф. **МАРТЫНОВ Виталий Владимирович** зав. кафедрой экономической информатики, ФГБОУ ВПО «Уфимский государственный авиационный технический университет» канд. техн. наук **АЛИМБЕКОВА Софья Робертовна** начальник отдела управления проектами ООО «НИИ технических систем «Пилот»»
Ведущая организация	ГБОУ ВПО «Башкирская академия государственной службы и управления при Президенте Республики Башкортостан»

Защита диссертации состоится «25» сентября 2013 г. в 10:00 часов
на заседании диссертационного совета Д-212.288.07
при Уфимском государственном авиационном техническом университете
по адресу: 450000, г. Уфа, ул. К. Маркса, 12

С диссертацией можно ознакомиться в библиотеке университета

Автореферат разослан «5» июня 2013 г.

Ученый секретарь
диссертационного совета
д-р техн. наук, проф.  И. Л. Виноградова

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования. Современный этап развития баз данных характеризуется активными исследованиями нереляционнных баз данных – объектно-ориентированных, документно-ориентированных, NoSQL и др. В УГАТУ на кафедре АСУ предложены и исследуются так называемые ситуационно-ориентированные базы данных (СОБД), в основе которых лежит динамическая модель предметной области, с состояниями которой ассоциированы хранимые данные в формате XML. СОБД удобны, например, при построении веб-приложений, управляемых динамическими моделями (Model-driven architecture).

Обработка XML-документов, вообще, и в СОБД, в частности, основана на технологии DOM (Document Object Model – объектная модель документа), которая предусматривает создание DOM-объектов, загрузку в них XML документов из внешней памяти, обработку узлов в дереве загруженного документа, формирование результирующих данных, сохранение документа во внешней памяти. В СОБД в настоящее время для этого приходится создавать программные модули, ассоциированные с состояниями динамической модели, что требует достаточно трудоемкого ручного программирования и приводит к объемному программному коду, дополняющему динамическую модель.

Общий подход к снижению трудоемкости программирования, показавший свою эффективность во многих областях, состоит в использовании высокоабстрактных декларативных спецификаций обработки данных, интерпретируемых на этапе компиляции или исполнения программы. Применительно к СОБД в рамках этого подхода высказана идея декларативного задания в динамической модели спецификаций по обработке XML-документов в зависимости от текущего состояния, названных *динамическими DOM-объектами*.

Идея динамических DOM-объектов является новой, в научной и технической литературе для нее отсутствуют концепции, модели, методы, лингвистическое, алгоритмическое, программное обеспечение. Поэтому создание научно обоснованного подхода к обработке XML-документов в СОБД на основе динамических DOM-объектов является актуальной научно-технической задачей.

Степень разработанности темы. XML (Extensible Markup Language – «расширяемый язык разметки») – «язык с простым формальным синтаксисом, удобный для создания и обработки документов программами и одновременно удобный для чтения и создания документов человеком, нацеленный на использование в Интернете» – создан усилиями D. Connolly, J. Bosak, M. Spilberg-McQueen, J. Clark и др, стандартизован консорциумом W3C и в настоящее время широко используется в разнообразных программных средствах как сам по себе, так и в виде основы для множества специализированных языков. XML-технологии широко применяются в документо-ориентированных базах данных, рассматриваемых обычно в рамках движения NoSQL (Not only SQL – «не только SQL»). В России полномасштабные исследования в области XML-ориентированных баз данных ведутся в Институте системного программирования РАН (проект Sedna).

DOM – «не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML-, XHTML- и XML-документов, а также изменять содержимое, структуру и оформление таких документов» – был разработан в середине 90-х гг. компаниями Netscape Communications и Microsoft, а затем стандартизован консорциумом W3C. В настоящее время он поддерживается в большинстве веб-браузеров и веб-серверов.

В УГАТУ отдельные вопросы использования баз данных XML исследовались В. В. Мироновым и Н. И. Юсуповой. Понятие СОБД, сочетающей иерархические ситуационные модели и XML-документы, впервые предложено в работе В. В. Миронова, Н. И. Юсуповой и Г. Р. Шакировой (2011). Ранее различные аспекты построения иерархических ситуационных моделей рассматривались в кандидатских диссертациях Ю. Б. Головкина, Р. А. Ярцева, Л. Е. Гончар, А. Н. Ситчихина, Р. Ф. Ахметшина. Применение иерархических ситуационных моделей на базе XML как основы веб-приложений рассматривалось в кандидатской диссертации К. Э. Маликовой.

Объект исследования – процесс обработки XML-документов в задачах управления контентом в веб-приложениях на базе СОБД.

Предмет исследования – возможность декларативного задания в динамической модели СОБД спецификаций по обработке XML-документов для динамического создания и загрузки DOM-объектов.

Цель исследования – снижение трудоемкости программирования процедур обработки XML-документов за счет декларативного задания в динамической модели СОБД спецификаций по их обработке.

Задачи исследования:

1) разработать *концепцию* динамических DOM-объектов в составе СОБД, позволяющую сократить трудоемкость программирования за счет декларативного описания функциональности в динамической модели;

2) предложить *модель* DOM-элемента в составе модели состояния динамической модели, позволяющую компактно декларировать функциональность обработки XML-документов;

3) предложить *метод* интерпретации DOM-элемента при обработке динамической модели, позволяющий формировать контент результирующего DOM-объекта на основе слияния нескольких XML-документов;

4) разработать *программное обеспечение* для интерпретации динамической модели СОБД в составе веб-приложений, позволяющее практически реализовать предложенный метод и добиться сокращения объема программного кода в части, необходимой для обработки XML-документов;

5) получить *экспериментальные зависимости* времени слияния XML-документов от количества обработанных узлов в процессе интерпретации DOM-элемента предложенным способом, подтверждающие принципиальную работоспособность и эффективность предложенного подхода к обработке XML-документов в СОБД.

Научная новизна

В целом новизна результатов заключается в идее создания динамических DOM-объектов управляемых текущими состояниями динамической модели и применении динамических DOM-объектов в новой области моделирования ситуационно-ориентированных баз данных.

По существу новизна результатов определяется следующим отличительными признаками:

1) *концепция динамических DOM-объектов*: а) в динамической модели предусматриваются DOM-элементы, ассоциированные с ее состояниями, описывающие, какие XML-документы необходимы в этих состояниях и как их следует обработать; б) в процессе интерпретации состояний динамической модели выполняется обработка ассоциированных DOM-элементов, в ходе которой автоматически порождаются соответствующие им DOM-объекты, в которые загружаются XML-документы и выполняется заданная для них трансформация;

2) *модель DOM-элемента*: предусмотрены атрибуты, задающие путь к родительскому XML-документу, а также внутренние элементы, задающие:а) иерархию элементов-источников, задающих обрабатываемые XML-документы, порядок их загрузки и условия слияния с родительским документом и друг с другом; б) элементы-приемники, специфицирующие XSL-трансформацию XML-документов и выгрузку результатов;

3) *метод интерпретации DOM-элемента*: в ходе обработки создается DOM-объект, в который загружается родительский XML-документ и в нем определяется множество целевых узлов; после чего выполняется рекурсивная обработка вложенных элементов-источников с созданием промежуточных DOM-объектов, загрузкой в них XML-документов, извлечением, фильтрацией, очищением множества результирующих узлов источников и их слиянием с множеством целевых узлов родительского XML-документа. При этом слияние с множеством целевых узлов может выполняться: а) «сверху-вниз», когда для каждого целевого узла отыскиваются и присоединяются в качестве дочерних соответствующие результирующие узлы источника; б) «снизу-вверх», когда для каждого результирующего узла источника отыскиваются соответствующие целевые узлы, к которым результирующие узлы присоединяются в качестве дочерних узлов;

4) *программное обеспечение* (техническая новизна): в рамках разработанной концепции и предложенной модели практически реализуется предложенный метод;

5) *экспериментальные зависимости*: для нового способа установлены а) близкий к линейному характер зависимостей в исследованном диапазоне; б) превосходство метода слияния «сверху-вниз» над методом «снизу-вверх».

Теоретическая и практическая значимость работы

Значение результатов для теории (методологии) СОБД выражается в том, что предложенная концепция динамических DOM-объектов расширяет возможности декларативного программирования функций СОБД за счет спецификации обработки XML-документов в динамической модели.

Значение результатов для практики построения документо-ориентированных приложений состоит в том, что они позволяют: сократить трудоемкость программирования за счет декларативного описания функциональности в динамической модели; компактно задавать функциональность обработки XML-данных; сократить объем программного кода в части, связанной с обработкой XML-документов, до 20 раз, что дает 9-кратное сокращение общего объема кода динамической модели (для рассмотренных примеров).

Методология и методы исследования. В работе использовались методы теории графов, технологии объектно-ориентированного программирования, системного анализа, ситуационного управления, иерархических моделей, объектной модели документа DOM, инструментарии кэшированной и потоковой обработки XML платформы PHP.

Положения, выносимые на защиту:

1. Концепция динамических DOM-объектов в составе ситуационно-ориентированной базы данных (СОБД), содержащей динамическую модель в виде иерархии возможных состояний и переходов между ними, а также набор XML-документов, *отличающаяся* тем, что

а) в динамической модели предусмотрены DOM-элементы, ассоциированные с ее состояниями, описывающие, какие XML-документы необходимы в этих состояниях и как их следует обработать;

б) в процессе интерпретации состояний динамической модели выполняется обработка ассоциированных DOM-элементов, в ходе которой автоматически порождаются соответствующие им DOM-объекты, в которые загружаются XML-документы и выполняется заданная для них трансформация.

2. Модель DOM-элемента в составе модели состояния динамической модели (см. п. 1), *отличающаяся* тем, что в ней предусмотрены атрибуты, задающие путь к родительскому XML-документу, а также внутренние элементы, задающие:

а) иерархию элементов-источников, задающих обрабатываемые XML-документы, порядок их загрузки и условия слияния с родительским документом и друг с другом;

б) элементы-приемники, специфицирующие XSL-трансформацию XML-документов и выгрузку результатов.

3. Метод интерпретации DOM-элемента при обработке динамической модели, *отличающийся* тем, что в ходе обработки создается DOM-объект, в который загружается родительский XML-документ и в нем определяется множество целевых узлов; после чего выполняется рекурсивная обработка вложенных элементов-источников с созданием промежуточных DOM-объектов, загрузкой в них XML-документов, извлечением, фильтрацией, очищением множества результирующих узлов источников и их слиянием с множеством целевых узлов родительского XML-документа.

При этом слияние с множеством целевых узлов может выполняться: а) "сверху-вниз", когда для каждого целевого узла отыскиваются и присоединяются в качестве дочерних соответствующие результирующие узлы источника; б) "снизу-вверх", когда для каждого результирующего узла источника отыскива-

ются соответствующие целевые узлы, к которым результирующие узлы присоединяются в качестве дочерних узлов.

4. Программное обеспечение на языке PHP для интерпретации динамической модели СОБД в составе веб-приложений, *отличающееся* тем, что оно обеспечивает в рамках разработанной концепции (см. п. 1) и предложенной модели (см. п. 2) практическую реализацию предложенного метода (см. п. 3).

5. Экспериментальные зависимости времени слияния XML-документов от количества обработанных узлов в процессе интерпретации DOM-элемента предложенным способом (см. п. 3), демонстрирующие: а) близкий к линейному характер зависимостей в исследованном диапазоне (от 10 до 100 000 узлов); б) превосходство (в 1,5–150 раз в зависимости от количества обработанных узлов) метода слияния «сверху-вниз» над методом «снизу-вверх».

Степень достоверности и апробация результатов. Достоверность результатов подтверждена путем разработки программного обеспечения, базирующегося на предложенных модели и методе, и практическом применении их в исследовательском прототипе веб-приложения на основе СОБД.

Разработанное программное обеспечение внедрено в научно-производственной фирме «РД-технология» и в ФГБОУ ВПО «УГАТУ».

Основные результаты диссертационного исследования были представлены на 10 конференциях российского и международного формата обсуждения.

Результаты получены в рамках плановых исследований в области СОБД, проводимых на кафедре автоматизированных систем управления УГАТУ при поддержке РФФИ (гранты №№ 10-07-00167 и 13-07-00011).

Публикации. Список публикаций по теме исследования включает 12 работ общим объемом 6 печатных листов, среди них 2 статьи в рецензируемом научном журнале из перечня ВАК, 6 публикаций без соавторов. Получено свидетельство о государственной регистрации программы для ЭВМ.

Структура диссертации. Диссертация включает введение, четыре главы, заключение, список литературы, и приложений. Объем диссертации 165 стр. машинописного текста, из которых основной текст составляет 145 стр.

СОДЕРЖАНИЕ ДИССЕРТАЦИИ

Во введении обсуждается тема, ставятся задачи, решаемые в ходе диссертационного исследования. Рассматриваются модели и методы, предлагаемые ситуационным подходом для решения задач в области проектирования ситуационно-ориентированных XML баз данных.

В первой главе рассматривается соотношение темы диссертационного исследования с фоном научно-технических тенденций и подходов в области проектирования баз данных, обсуждается ее актуальность, формулируется цель исследования, а также ставятся задачи, решаемые для достижения поставленной цели. Обосновывается выбор методов и средств, необходимых для воплощения концепции моделирования СОБД на основе динамических DOM-объектов. Приводятся основные результаты и положения, выносимые на защиту. Характеризуется науч-

ная новизна и практическая значимость, полученные в результате апробации работы.

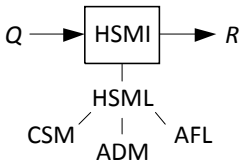


Рисунок 1 — Архитектура ситуационно-ориентированной базы данных

Во второй главе в русле подхода к разработке информационных систем на основе использования моделей высокого уровня абстракции (Model-based, model-driven engineering) рассматривается идея динамических DOM-объектов в составе СОБД, в которых данные в формате XML ассоциированы с состояниями встроенной динамической модели и могут обрабатываться в контексте ее текущих состояний. СОБД могут служить основой интернет-приложений, обеспечивающих динамическое формирование контента в соответствии со сложившейся ситуацией.

В СОБД входят следующие компоненты (рис. 1):

- HSML (HSM Library) — библиотека динамических моделей, содержащая набор иерархических ситуационных моделей HSM (Hierarchical Situational Models) в виде иерархии графов переходов с конечным числом состояний (Finite State Model);
- CSM (Current State Memory) — память текущего состояния, хранящая сведения о текущих состояниях динамических моделей;
- ADM (Associated Data Memory) — память ассоциированных данных, хранящая XML-данные, соотнесенные с различными состояниями динамической модели;
- AFL (Associated Functions Memory) — библиотека ассоциированных функций, хранящая функции обработки данных, соотнесенные с состояниями динамической модели;
- HSMI (HSM Interpreter) — интерпретатор динамической модели, который в ответ на внешний запрос Q формирует ответ R путем обработки динамической модели HSM из HSML на основе отслеживания ее текущих состояний, сохраняемых в CSM, обработки ассоциированных данных из ADM и выполнения ассоциированных функций из AFL.

Решаемая задача. Таким образом, следует предоставить возможность специфицировать на уровне HSM преобразование XML-данных, содержащихся в DOM-объектах. Объекты с подобным поведением будем называть динамическими DOM-объектами.

DOM-элементы. В ходе интерпретации динамической модели могут автоматически создаваться DOM-объекты, для этого предусмотрим в модели специальные DOM-элементы с заданными для них источниками XML-данных. Соответственно в процессе интерпретации модели предусмотрим обработку указанных DOM-элементов, создание соответствующих им DOM-объектов и загрузку в них XML-данных из различных источников, когда родительские состояния становятся текущими, удаление DOM-объектов, когда родительские состояния перестают быть текущими.

Источники XML-данных. Может быть предложено несколько вариантов наполнения данными DOM-объектов. DOM-элемент представляется символом

dom, справа от него, указывается имя элемента (в данном случае, DOM-элементы dom:D1, dom:D2, dom:D3). Символ источника данных src, прикрепленный к DOM-элементу в качестве дочернего элемента, специфицирует особенности загрузки XML-данных в DOM-объект, порождаемый DOM-элементом. Атрибуты элемента-источника, в свою очередь, специфицируют, откуда берутся XML-данные.

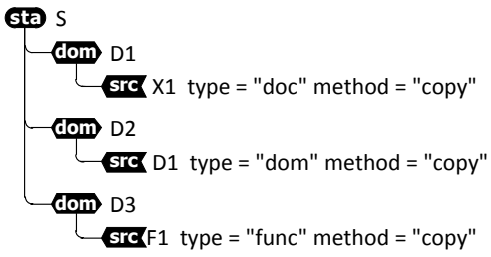


Рисунок 2 — Задание источника XML-данных: как XML-документа; как другого DOM-элемента; как результата функции

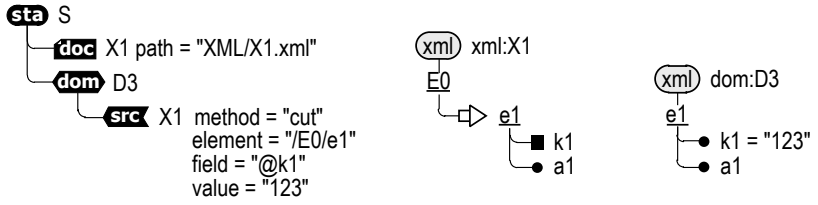
На рис. 2 для DOM-элемента специфицирован источник данных в виде XML-документа из ADM, специфицирован источник данных в виде ссылки на другой DOM-элемент, специфицирован источник данных в виде функции, возвращающей в качестве результата строку XML-данных.

Загрузка, предусматривающая фильтрацию XML-данных. В реальных условиях может потребоваться отфильтровать данные. На рис. 3 иллюстрируется фильтрация для источника данных в виде XML-файла из ADM.

В динамической модели на рис. 3, *а* в корневом состоянии sta:S заданы определения XML-документа doc:X1, хранящегося в ADM, и DOM-элемента dom:D3, в котором предусмотрена загрузка документа doc:X1 с фильтрацией. Источник данных src:X1 имеет атрибут method = "cut", указывающий на то, что из исходного XML-документа будет «вырезано» поддерево. Дополнительные атрибуты «element», «field» и «value», задают условие фильтрации. Атрибут «element» содержит XPath-выражение, которое определяет множество узлов-элементов в XML-дереве, один из которых будет корнем загружаемого поддерева. Атрибут «field» содержит XPath-выражение, которое задает в поддереве проверяемый XML-элемент или атрибут. Атрибут «value» содержит требуемое значение. В данном примере требуется загрузить в DOM-объект поддерево, начинающееся в элементе «e1», атрибута «k1» которого имеет значение «123».

На рис. 3, *б* представлена модель XML-данных doc:X1 с корневым XML-элементом «E0», он может содержать несколько дочерних XML-элементов «e1», каждый из которых содержит атрибуты «k1» (идентификатор) и «a1». Таким образом, XPath-выражение «/E0/e1», заданное в атрибуте «element» источника src:X1 на рис. 3, *а*, адресует множество всех XML-элементов «e1»; атрибут «field» адресует в «e1» XML-атрибут «k1», а атрибут «value» задает для него искомое значение «123».

В ходе обработки элемента dom:D3 интерпретатор обращается к источнику src:X1, перебирает XML-элементы «e1», отыскивая тот, у которого XML-атрибут «k1» имеет искомое значение «123», и загружает соответствующее поддерево в DOM-объект. В результате в DOM-объект «D3» будут загружены XML-данные, соответствующие модели, представленной на рис. 3, *в*.

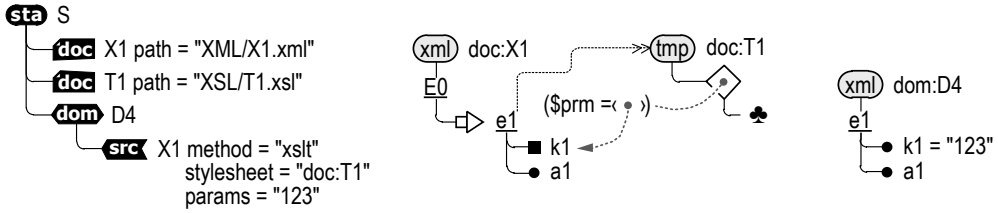


а *б* *в*

Рисунок 3 — Пример загрузки DOM-объекта с фильтрацией XML-данных: *а* — фрагмент динамической модели; *б* — модель XML-данных источника; *в* — модель XML-данных, загружаемых в DOM-объект

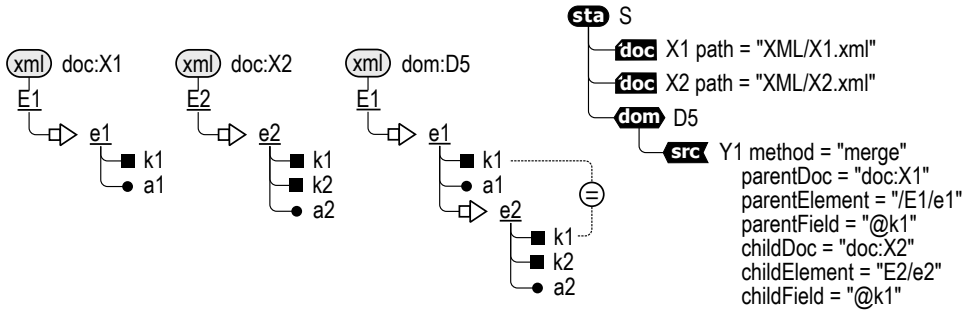
Загрузка, предусматривающая слияние XML-данных. На рис. 5 иллюстрируется источник, предусматривающий слияние XML-документов. Модель первого из сливаемых документов, содержащего общие сведения и выступающего в качестве родительского, приведена на рис. 5, *а* (см. рис. 4). Модель второго из сливаемых документов, содержащего детальные сведения и выступающего в качестве дочернего, приведена на рис. 5, *б*. Модель результирующего документа, загружаемого в DOM-объект, приведена на рис. 5, *в*.

Соответствующий фрагмент динамической модели, предусматривающий слияние двух документов при загрузке DOM-объекта «D3», приведен на рис. 5, *г*. Элемент-источник имеет атрибут method = "merge", сообщающий интерпретатору, что загружаемые в DOM-объект XML-данные получаются путем слияния двух XML-документов. Эти документы задаются атрибутами parentDoc (родительский документ) и childDoc (дочерний документ).



а *б* *в*

Рисунок 4 — Пример загрузки DOM-объекта с XSL-трансформацией данных: *а* — фрагмент динамической модели данных; *б* — модель XSL-трансформации; *в* — модель XML-данных, загружаемых в DOM-объект



а *б* *в* *г*

Рисунок 5 — Пример источника со слиянием XML-данных: *а* — модель общих сведений; *б* — модель детальных сведений; *в* — модель результирующих данных, загружаемых в DOM-объект; *г* — фрагмент динамической модели, предусматривающей слияние документов

В третьей главе в отличие от известного подхода, где функциональность манипулирования DOM-объектами достигается за счет программирования соответствующих функций в подпрограммах-акциях, ассоциированных с состояниями динамической модели, здесь:

1) у элементов-состояний динамической модели в качестве дочерних предусматриваются DOM-элементы, у которых, в свою очередь, дочерние элементы-источники задают загружаемые XML-данные, а дочерние элементы-приемники — сохраняемое XML-содержимое;

2) в ходе интерпретации динамической модели интерпретатором выполняется автоматическое создание DOM-объектов для текущих состояний модели и загрузка XML-данных с возможным преобразованием, а также автоматическое удаление DOM-объектов при смене текущих состояний.

На рис. 7 иллюстрируется процесс формирования XML-содержимого DOM-объекта на основе информации из источников данных.

XML-технологии предусматривают два основных подхода к обработке документов:

- **потоковая** обработка – XML-документ обрабатывается потоком по мере чтения его узлов из файла. Документ при этом размещается в оперативной памяти не целиком, а частями.
- **кэшированная** обработка – перед обработкой XML-документ целиком загружается в структурированном виде (в виде дерева) в оперативную память компьютера. Это дает возможность гибкой обработки документов ограниченного объема. Инструмент кэшированной обработки – технология DOM.

DOM-элемент. На рис. 6 приведена синтаксическая диаграмма DOM-элемента. DOM-элемент может содержать следующие необязательные атрибуты, указывающие исходный XML-документ для загрузки в порождаемый DOM-объект:

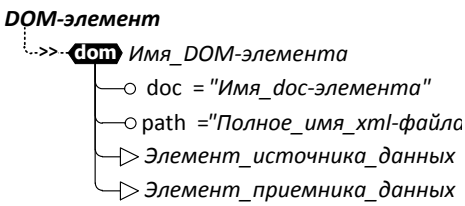


Рисунок 6 — Синтаксическая диаграмма DOM-элемента

- doc – задает XML-документ в виде ссылки на doc-элемент;
- path – задает XML-документ путем явного указания пути к XML-файлу.

DOM-элемент может содержать несколько дочерних элементов – источников и приемников данных.

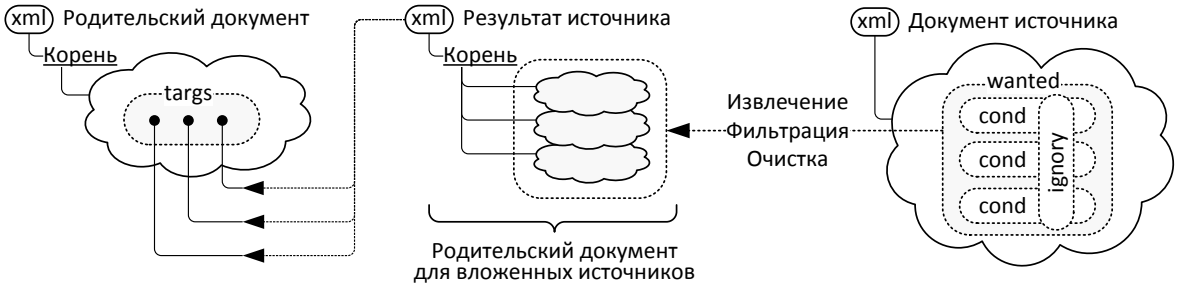


Рисунок 7 — Формирование XML-содержимого DOM-объекта на основе информации из источников данных

Элемент источника данных представляет собой элемент динамической модели, размещаемый в DOM-элементе в качестве потомка. Он включает набор атрибутов, задающих свойства источника, и может, в свою очередь, содержать дочерние элементы-источники. Синтаксическая диаграмма элемента-источника приведена на рис. 8.

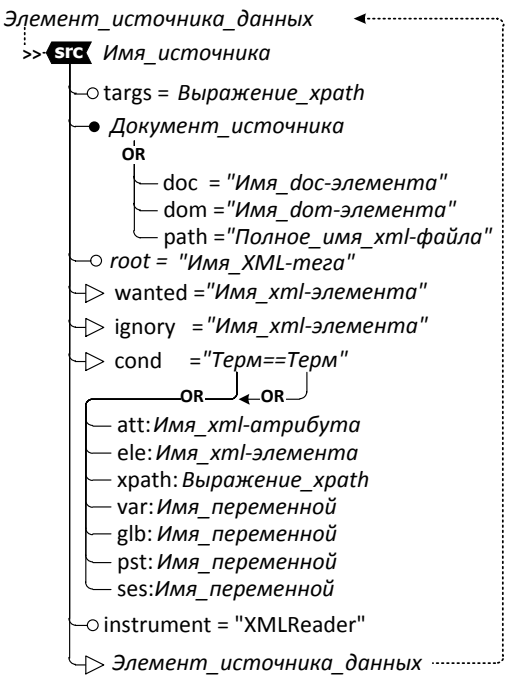


Рисунок 8 — Синтаксическая диаграмма элемента-источника

Атрибуты элемента-источника:

- targs – атрибут, указывающий в родительском XML-документе множество целевых XML-элементов, к каждому из которых будет прикреплен в качестве дочернего элемента XML-результат источника.
- *Документ_источника* – обязательная конструкция, задающая документ, из которого берется XML-содержимое.

Эта конструкция представляет собой один из трех атрибутов:

- path – задает документ источника путем явного указания пути к XML-файлу;
- doc – задает документ источника в виде ссылки на doc-элемент;
- dom – задает в качестве документа XML-содержимое DOM-объекта, который порожден DOM-элементом с указанным именем;
- root – атрибут, предписывающий дополнительно заключить XML-результат источника в теги с заданным именем;
- wanted – многозначный атрибут, задающий список имен XML-элементов, отыскиваемых в документе источника для формирования;
- ignory – многозначный атрибут, задающий список имен XML-элементов в документе источника, которые следует игнорировать при формировании результата;
- cond – многозначный атрибут, задающий список условий, которым должен удовлетворять искомый XML-элемент документа источника (из списка в атрибуте wanted), чтобы попасть в результат.
- instrument – необязательный атрибут, предписывающий использовать для чтения источника XMLReader – метод потокового доступа.

Головной алгоритм интерпретации DOM-элемента представлен в виде блок-схемы (интерпретации) DOM-элемента. При отсутствии DOM-объекта с именем обрабатываемого DOM-элемента создается DOM-объект с корневым элементом. Затем проверяется наличие атрибута doc или path и выполняется загрузка в DOM-объект соответствующего документа. Далее в цикле обрабатываются все дочерние элементы DOM-элемента. Для каждого дочернего элемента проверяется, является ли он источником, и в этом случае для него вызывается алгоритм обработки источника данных, является ли он приемником, и в этом случае для него

вызывается алгоритм обработки приемника данных. По окончании цикла алгоритм завершается и выполняет возврат в точку вызова.

Алгоритм кэшированной обработки элементов документа источника изображен на блок-схеме на рис. 9. Алгоритм начинает работу с подготовки кэша в виде DOM-объекта с загруженным в него XML-документом источника. Далее формируется массив ссылок на искомые элементы документа источника, имена которых заданы в атрибуте wanted элемента-источника. Путем цикла по массиву ссылок проверяется каждый искомый элемент. Проверяется выполнение условий фильтрации, которые заданы в атрибуте cond. Если обрабатываемый узел успешно прошел проверки, он присоединяется в качестве дочернего элемента к целевому элементу родительского документа. Далее выполняется очистка присоединенного дочернего элемента от внутренних элементов, имена которых заданы в атрибуте ignory. После того как обрабатываемый узел документа источника проверен, присоединен к целевому элементу и очищен, для него рекурсивно обрабатываются внутренние источники данных.

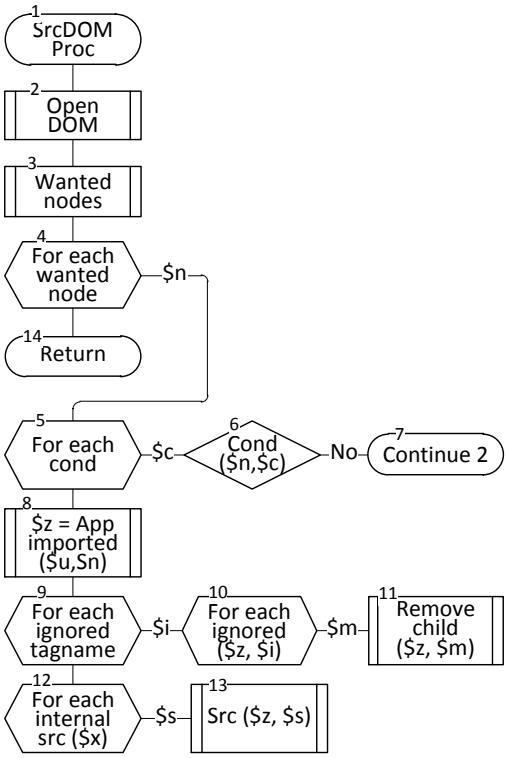


Рисунок 9 — Алгоритм кэшированной обработки элементов документа источника

Алгоритм потоковой обработки элементов документа источника представлен в виде блок-схемы. Алгоритм начинает работу с инициализации инструмента потокового ввода XMLReader. Далее в цикле выполняется потоковое чтение узла за узлом из документа источника. Проверяется, присутствует ли в элементе-источнике атрибут wanted, и в этом случае проверяется, является ли обрабатываемый узел искомым, в цикле перебираются имена из списка, заданного в атрибуте wanted, сравниваются с именем обрабатываемого узла и в случае совпадения устанавливается флаг. По завершении цикла проверяется флаг совпадения и, если флаг не установлен, завершается обработка данного узла и выполняется переход к обработке следующего. Если обрабатываемый узел признан искомым, он проверяется на выполнение условий фильтрации, которые заданы в атрибуте cond. Такие условия требуют обращения к другим узлам документа из потока ввода. Для этого обрабатываемый элемент (вместе с содержимым) извлекается в кэш. Далее в цикле из атрибута cond извлекаются условия этого рода и проверяются для обрабатываемого узла, находящегося в кэше. Если обрабатываемый узел прошел все проверки, он присоединяется как дочерний элемент к обрабатываемому целевому элементу родительского документа. Далее выполняется очистка присоединенного дочернего элемента от внутренних элементов, имена которых заданы в атрибуте ignory. После того как обрабатываемый узел документа источника

проверен, присоединен к целевому элементу и очищен, для него рекурсивно обрабатываются внутренние источники данных.

Алгоритм интерпретации элемента-источника представлен в виде блок-схемы. При наличии атрибута targs у элемента-источника массив целевых элементов формируется в результате поиска в родительском документе множества элементов с именами, заданными в атрибуте targs. В противном случае множество целевых элементов включает единственный корневой элемент родительского документа. После этого выполняется циклическая обработка по каждому целевому элементу. В зависимости от атрибута "instrument" вызывается алгоритм потоковой или кэшированной обработки.

В четвертой главе рассматривается работоспособность и эффективность разработанного исследовательского прототипа веб-приложения.

Разработанное алгоритмическое обеспечение реализовано в виде модулей на языке серверных сценариев PHP в составе интерпретатора динамической модели СОБД. С его помощью проведено экспериментальное исследование времени обработки DOM-элементов при загрузке XML-документов разного объема. Результаты измерений показывают, что метод "сверху-вниз" при объеме 10 узлов в 1,5 раза быстрее метода "снизу-вверх", выигрыш в производительности при объеме 100 узлов увеличивается в 3 раза, при 1 000 в 10 раз, при 10 000 и 100 000 в 150 раз. Полученные экспериментально зависимости производительности интерпретации DOM-элементов носят близкий к линейному характер на исследуемом интервале от 10 до 100 000 узлов.

Для демонстрации работоспособности полученных результатов на реальном примере и оценки практического эффекта использования результатов в плане сокращения объема программного кода был разработан исследовательский прототип веб-приложения с использованием динамических DOM-объектов. В качестве основы для разработки использовано приложение, относящееся к классу систем управления содержимым (англ. Content management system, CMS). Исходное приложение предоставляет возможность диссертантам загружать данные, относящиеся к процессу защиты диссертаций и генерировать на их основе заготовки документов, сопровождающих диссертационный процесс, а другим участникам диспроцесса – руководству и членам диссертационных советов, руководству организации и др. – получать информацию, контролировать и управлять ходом диссертационного процесса. Оно основано на динамической модели, в которой фрагменты, относящиеся к обработке XML-документов, были запрограммированы в виде традиционных вызываемых процедур.

Исходная динамическая модель была модифицирована так, что обработка XML-данных осуществляется с помощью DOM-элементов. Модель содержит 28 субмоделей, общее количество состояний составляет 108. Из них в 76 состояниях (70 %) требуется обработка XML-данных, для чего использованы DOM-элементы.

В полученном веб-приложении достаточно большой процент состояний динамической модели – 70 % – содержит DOM-элементы. Это свидетельствует о том, что для рассматриваемой задачи динамические DOM-объекты интенсивно используются, т. е. оказались вполне востребованы.

Из 77 DOM-элементов 4 элемента предусматривают только загрузку данных в DOM-объект для последующего использования, 50 элементов – только вывод данных из предварительно загруженного DOM-объекта, 8 элементов – сохранение содержимого DOM-объекта в XML-документе, 15 элементов – то или другое вместе. Элементы первого типа составляют 5 %, второго – 65 %, третьего – 10 %, четвертого – 20 % от общего количества.

DOM-элементы используются многократно: DOM-объекты, созданные и загруженные в одном состоянии, затем используются для вывода тех или иных данных в той или иной форме в нескольких других состояниях (как правило – в подсостояниях исходного состояния). Так, загрузка DOM-элемента сведениями о выбранном диссертанте предусмотрена в 3 состояниях, а использование этих загруженных сведений – в 53 состояниях.

Из 19 DOM-элементов, предусматривающих загрузку DOM-объектов, большинство, а именно 13 (69 %), делают это на основе одного XML-документа. Однако встречаются случаи, требующие формирование контента DOM-объекта на основе нескольких XML-документов, а именно: в 5 элементах (26 %) используется два XML-документа, а в 1 элементе (5 %) – три XML-документа.

Использование иерархических DOM-объектов ведет к тому, что часть функций, связанных с созданием DOM-объектов и загрузкой в них XML-документов, которые раньше разработчик должен был запрограммировать вручную в виде программных модулей, вызываемых при интерпретации динамической модели, теперь выполняется автоматически интерпретатором на основании спецификаций, содержащихся в DOM-элементах. Это ведет к сокращению объема кода, который необходимо запрограммировать при разработке веб-приложения. Оценим величину такого сокращения. Для этого сравним объем кода, необходимого для представления DOM-элементов, с объемом кода, выполняемого интерпретатором в ходе обработки этих DOM-элементов. При сокращении программного кода, разумеется, программист или проектировщик избавляется от труда по созданию функциональности обработки данных в источниках и DOM-объектах. Кроме того, при написании модели временные затраты будут также незначительными. Использование динамических DOM-объектов показывает, что спецификация 10 DOM-объектов занимает примерно 2 Кб программного кода модели, тогда как спецификация одного DOM-объекта вручную занимает 4 Кб. Сокращение кода динамической модели за счет динамических DOM-объектов 9-кратный, в данном случае сравнивался код динамической модели, из которой были вычленены DOM-объекты и программный код, требуемый для описания аналогичной функциональности.

В заключении изложены основные результаты работы.

В приложениях приведены: листинги модулей интерпретации динамических DOM-объектов СОБД, а также листинг тестовой динамической модели с DOM-элементами и элементами-источниками.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

В работе представлены новые научно обоснованные информационно-технологические решения и разработки в области управления XML-данными в ситуационно-ориентированных базах данных, имеющие значение для развития веб-технологий. При этом получены результаты:

1. Концепция динамических DOM-объектов в составе СОБД, содержащей динамическую модель в виде иерархии возможных состояний и переходов между ними, а также набор XML-документов, *отличающаяся* тем, что

а) в динамической модели предусмотрены DOM-элементы, ассоциированные с ее состояниями, описывающие, какие XML-документы необходимы в этих состояниях и как их следует обработать;

б) в процессе интерпретации состояний динамической модели выполняется обработка ассоциированных DOM-элементов, в ходе которой автоматически порождаются соответствующие им DOM-объекты, в которые загружаются XML-документы и выполняется заданная для них трансформация.

Это позволяет сократить трудоемкость программирования за счет декларативного описания функциональности в динамической модели.

2. Модель DOM-элемента в составе модели состояния динамической модели (см. п. 1), *отличающаяся* тем, что в ней предусмотрены атрибуты, задающие путь к родительскому XML-документу, а также внутренние элементы, задающие:

а) иерархию элементов-источников, задающих обрабатываемые XML-документы, порядок их загрузки и условия слияния с родительским документом и друг с другом;

б) элементы-приемники, специфицирующие XSL-трансформацию XML-документов и выгрузку результатов.

Это позволяет компактно задавать функциональность обработки XML-данных.

3. Метод интерпретации DOM-элемента при обработке динамической модели, *отличающийся* тем, что в ходе обработки создается DOM-объект, в который загружается родительский XML-документ и в нем определяется множество целевых узлов; после чего выполняется рекурсивная обработка вложенных элементов-источников с созданием промежуточных DOM-объектов, загрузкой в них XML-документов, извлечением, фильтрацией, очищением множества результирующих узлов источников и их слиянием с множеством целевых узлов родительского XML-документа.

При этом слияние с множеством целевых узлов может выполняться:

а) "сверху-вниз", когда для каждого целевого узла отыскиваются и присоединяются в качестве дочерних соответствующие результирующие узлы источника;

б) "снизу-вверх", когда для каждого результирующего узла источника отыскиваются соответствующие целевые узлы, к которым результирующие узлы присоединяются в качестве дочерних узлов.

Это позволяет формировать контент результирующего DOM-объекта на основе слияния нескольких XML-документов

4. Программное обеспечение на языке PHP для интерпретации динамической модели СОБД в составе веб-приложений, обеспечивающее в рамках разработанной концепции (см. п. 1) и предложенной модели (см. п. 2) практическую реализацию предложенного метода (см. п. 3).

Это позволяет сократить объем программного кода в части, необходимой для обработки XML-документов, до 20 раз, что дает 9-кратное сокращение общего объема кода динамической модели (для рассмотренных примеров).

5. Экспериментальные зависимости времени слияния XML-документов от количества обработанных узлов в процессе интерпретации DOM-элемента предложенным способом (см. п. 3), демонстрирующие:

а) близкий к линейному характер зависимостей в исследованном диапазоне (от 10 до 100 000 узлов);

б) превосходство (в 1,5–150 раз в зависимости от количества обработанных узлов) метода слияния «сверху-вниз» над методом «снизу-вверх».

Это подтверждает принципиальную работоспособность и эффективность предложенного подхода к обработке XML-документов в СОБД.

Перспективы дальнейшей разработки темы. В ходе дальнейших исследований планируется разработка концепции, моделей, методов и программного обеспечения, позволяющих масштабировать СОБД на основе динамических DOM-объектов. Предполагается построить модель высокого уровня абстракции, обеспечивающую снижение трудоемкости масштабирования баз данных с помощью описания данной функциональности на языке динамической модели.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

В рецензируемых журналах из списка ВАК

1. Ситуационно-ориентированные базы данных: концепция управления XML-данными на основе динамических DOM-объектов / В. В. Миронов, А. С. Гусаренко // Вестник УГАТУ: науч. журн. Уфимск. гос. авиац. техн. ун-та. 2012. Т. 16, № 3 (48). С. 159–172.

2. Динамические DOM-объекты в ситуационно-ориентированных базах данных: лингвистическое и алгоритмическое обеспечение источников данных / В. В. Миронов, А. С. Гусаренко // Вестник УГАТУ: науч. журн. Уфимск. гос. авиац. техн. ун-та. 2012. Т. 16, № 6 (51). С. 167–176.

В других изданиях

3. Управление XML-данными на основе динамических DOM-объектов / А. С. Гусаренко // Перспективы развития информационных технологий: сб. мат. 6-й междунар. науч.-практ. конф. Новосибирск: Сибпринт, 2011. С. 103–108.

4. Управление динамическими DOM-объектами в ситуационно-ориентированных базах данных / А. С. Гусаренко // Актуальные научные проблемы: мат. 4-й всерос. заоч. науч.-практ. конф. Екатеринбург: Изд. дом "Ажур", 2011. С. 48–51.

5. Управление XML-данными с помощью динамических DOM-объектов в ситуационно-ориентированных базах данных / В. В. Миронов, А. С. Гусаренко // Модернизация современного общества: пути развития и перспективы: матер. 2-й

междунар. науч.-практ. конф. Ставрополь: ЦНЗ "Логос", 2011. С. 38–43. (опубл. на англ. яз.)

6. Ситуационно-ориентированные базы данных: концепция управления XML-данными основанная на динамических DOM-объектах / В. В. Миронов, А. С. Гусаренко // Труды 14-й междунар. конф. по выч. наукам и инфор. техн. (CSIT'2012). Уфа-Гамбург-Норвежские Фьорды, 2012. Т. 1. С. 61–67. (опубл. на англ. яз.)

7. CRUD-операции в ситуационно-ориентированных базах данных на основе динамических DOM-объектов / А. С. Гусаренко // Повышение эффективности использования информационных технологий в государственном и муниципальном управлении: матер. всерос. науч.-практ. конф. с участием междунар. представителей. Уфа: БАГСУ, 2012. С. 34–37.

8. Ситуационно-ориентированные динамические DOM-объекты в XML-базах данных / А. С. Гусаренко // Повышение эффективности использования информационных технологий в государственном и муниципальном управлении: матер. всерос. науч.-практ. конф. с участием междунар. представителей, Уфа: БАГСУ, 2012. С. 37–40.

9. Инфраструктура ситуационно-ориентированных баз данных на основе динамических DOM-объектов / А. С. Гусаренко // Информационные и инфокоммуникационные технологии: сб. науч. тр. 7-й всерос. зим. шк.-сем. аспирантов и молодых ученых, 2012, Уфа: УГАТУ, 2012. Т. 1. С. 58–62.

10. Структуры динамических DOM-объектов в ситуационно-ориентированных базах данных / А. С. Гусаренко // Информационные и инфокоммуникационные технологии: сб. науч. тр. 7-й всерос. зимн. шк.-сем. аспирантов и молодых ученых. Уфа: УГАТУ, 2012. Т. 1. С. 62–66.

11. Применение динамических DOM-объектов в ситуационно-ориентированных базах данных / В. В. Миронов, А. С. Гусаренко // Современное состояние естественных и технических наук: матер. 5-й междунар. науч.-практ. конф. М.: Спутник+, 2012. С. 93–97.

12. Профилирование алгоритма ситуационно-ориентированной базы данных на основе динамических DOM-объектов / А. С. Гусаренко // Информационные и инфокоммуникационные технологии: сб. науч. тр. 8-й всерос. зим. шк.-сем. аспирантов и молодых ученых, 2013, Уфа: УГАТУ, 2013. Т. 1. С. 115–118.

13. Свид. о гос. рег. программы для ЭВМ № 2013610948. Модули интерпретации динамических DOM-объектов ситуационно-ориентированной базы данных / В. В. Миронов, А. С. Гусаренко. Зарег. 09.01.2013. М.: Фед. сл. по интел. собств. (Роспатент).

Диссертант  А. С. Гусаренко

ГУСАРЕНКО Артем Сергеевич

ОБРАБОТКА XML-ДОКУМЕНТОВ В СИТУАЦИОННО-ОРИЕНТИРОВАННЫХ БАЗАХ ДАННЫХ НА ОСНОВЕ ДИНАМИЧЕСКИХ DOM-ОБЪЕКТОВ

Специальность 05.13.11

Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

кандидата технических наук

Подписано к печати «28» мая 2013. Формат 60x84 1/16.
Бумага офсетная. Печать плоская. Гарнитура Times New Roman.
Усл. печ. л. 1,0. Усл. кр.-отт. 1,0. Уч.-изд. л. 0,9.
Тираж 100 экз. Заказ № 341.

ФГБОУ ВПО «Уфимский государственный авиационный технический университет»
Центр оперативной полиграфии
450000, Уфа-центр, ул. К. Маркса, 12