Отображение виртуальных XML-документов на таблицы MySQL в ситуационно-ориентированных базах данных: «распределенный» подход[1]

В.В. Миронов, А.С. Гусаренко, Н.И. Юсупова

Аннотация. Ситуационно-ориентированные базы данных (СОБД) – это информационный процессор в составе веб-приложения, обрабатывающий виртуальные XML-документы в соответствии со встроенной ситуационной моделью. В статье рассматривается задача «распределенного» отображения виртуальных XML-документов на реляционную базу данных, при котором данные одного документа физически хранятся во множестве строк таблиц. Обосновывается подход, при котором результат SQL-запроса к базе данных сопоставляется с «плоским» виртуальным XML-документом, который в дальнейшем может быть преобразован в «иерархический» XML-документ средствами СОБД. Обсуждаются примеры извлечения «плоского» XML-документа из базы данных, а также модификации базы данных. Рассматриваются и иллюстрируются два подхода к получению «иерархических» XML-документов – на основе вложенных источников данных и на основе XSL-трансформации. Практическая реализация подхода выполнена на платформе PHP с использованием расширения mysqli для связи с базами данных сервера MySQL.

Ключевые слова: ситуационно-ориентированная база данных; веб-приложение; управление на основе встроенной модели; иерархическая ситуационная модель; виртуальный документ; HSM; NoSQL; XML; DOM; MySQL; PHP; mysqli; SELECT FOR XML.

Введение

При создании современных веб-приложений активно используются новые подходы к управлению на основе встроенных моделей (MDA – Model-Driven Approach), когда для управления веб-приложениями применяются высокоабстрактные модели и нотации [1–6]. Другое активно исследуемое направление развития веб-приложений – движение NoSQL (Not only SQL) – использование нереляционных баз данных наряду с реляционными [7–10].

Ситуационно-ориентированные базы данных (СОБД) [11] – развиваемый авторами подход к управлению веб-приложениями на основе иерархической ситуационной модели – соответствует как MDA (поскольку основан на встроенной конечномерной модели состояний бизнес-процесса), так и NoSQL (в части документоцентричных хранилищ данных). СОБД рассматривается как виртуальное хранилище документов веб-приложения, которое отображается на реальные документы в разнородных хранилищах.

Представленная на Рис. 1 архитектура СОБД включает:

- набор встроенных ситуационных моделей HSM (Hierarchical Situation Model), задающих

[1]Выполнено в рамках проекта, поддержанного РФФИ, грант 16-07-00239.

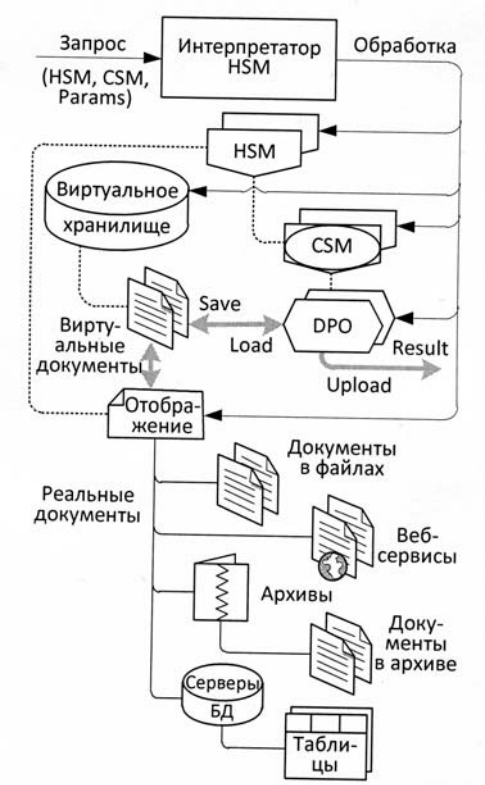

Рис. 1. Архитектура СОБД с виртуальным хранилищем документов [11]

иерархию возможных состояний бизнес-процесса и переходов между ними;

• набор моделей текущего состояния CSM (Current State Model), задающих текущие состояния соответствующих HSM;

• виртуальное хранилище данных бизнес-процессов, включающее набор виртуальных документов в формате XML или JSON;

• интерпретатор ситуационной модели, выполняющий по внешнему запросу цикл интерпретации – обработку заданной в запросе ситуационной модели HSM в контексте заданной модели CSM.

В ходе обработки HSM интерпретатором обновляются текущие состояния в CSM; в соответствии с текущими состояниями создаются объекты обработки данных DPO (Data Processing Objects), такие как DOM (Document Object Model); в DPO загружаются (Load) документы из виртуального хранилища, обрабатываются и сохраняются (Save) или передаются вовне как результат (Result).

Документы виртуального хранилища отображаются на реальные электронные документы, хранящиеся в гетерогенных (разнородных) хранилищах данных, в качестве которых могут выступать: файловая система веб-сервера – документы в виде размеченных текстовых файлов [12–14]; удаленные веб-сервисы – документы передаются по сети Интернет [15]; zip-архивы – документы в виде заархивированных файлов [16, 17]; серверы баз данных – документы в виде данных в таблицах базы данных реляционного сервера. В рамках проекта СОБД авторы ориентировались на популярную свободно распространяемую «связку»: веб-сервер Apache, язык серверных сценариев PHP, сервер реляционных баз данных MySQL.

Для обработки XML-данных в HSM предусмотрено 4 типа элементов:

dom – dom-элемент – DPO-элемент, для которого на цикле интерпретации создается объект DOM (Document Object Model), в который загружается и в котором обрабатывается виртуальный документ XML;

src – src-элемент – источник данных, дочерний для dom-элемента, определяющий загрузку виртуального XML-документа в родительский DOM-объект;

rcv – rcv-элемент – приемник данных, дочерний для dom-элемента, определяющий выгрузку XML-документа из родительского DOM-объекта;

doc – doc-элемент – элемент виртуального документа, определяющий виртуальный документ и его отображение на реальные данные в хранилище.

В реальных условиях документ, загруженный в DOM-объект, подвергается нетривиальной обработке, например, XSL-трансформации. Более подробно детали работы с DOM-объектами описаны в [12–14].

Задача отображения XML-документов в реляционную среду

Известны два основных подхода к отображению XML-документов в таблицы реляционной базы данных:

1) «Сосредоточенный», основанный на хранении XML-документа целиком в ячейке таблицы базы данных («один документ – одна

ячейка»). Применительно к СОБД этот подход рассматривался авторами в работах [23–29].

2) «Распределенный», основанный на хранении данных, составляющих документ, в нескольких ячейках в общем случае нескольких таблиц базы данных и сборке/разборке документа «на лету» (on-the-fly) в процессе его извлечения/сохранения («один документ – несколько ячеек»). Применительно к СОБД такой подход ранее не исследовался, здесь восполняется этот пробел.

Для задачи извлечения и сборки «на лету» XML-документа в стандарте SQL:2003 был предусмотрен специальный SQL-запрос `SELECT FOR XML`, предусматривающий ряд возможностей от простейшего, «плоского» варианта `RAW`, при котором результирующая таблица запроса `SELECT` построчно преобразуется в XML-документ, до наиболее гибкого варианта `EXPLICIT`, позволяющего получать из результирующей таблицы запроса `SELECT` «иерархический» XML-документ со сложной структурой. Очевидно, что в идеале реализация этой задачи в СОБД должна базироваться на этих возможностях. Этот подход реализован в крупномасштабных серверах баз данных, таких как Oracle Database, DB2, Microsoft SQL Server. Из отечественных разработок следует отметить СУБД НИКА (xNika), в которой применяется подход, позволяющий динамически преобразовывать XML-документы в данные таблиц и обратно, а также создавать и перестраивать таблицы реляционной базы данных в соответствии с XSD-схемой XML-документа [18–20]. Следует отметить также отечественный проект Sedna – систему управления базами данных, изначально спроектированную для хранения и обработки XML-данных. Sedna поддерживает древовидную модель данных, которые загружаются и извлекаются в виде XML-документов [21, 22]. Проблема состоит в том, что в настоящее время не все реляционные СУБД (в том числе и MySQL) поддерживают `SELECT FOR XML`. Учитывая, что в будущем данная опция будет, несомненно, введена в таком популярном и развивающемся сервере, как MySQL, представляется нецелесообразным реализовывать ее сейчас в полной мере на уровне СОБД.

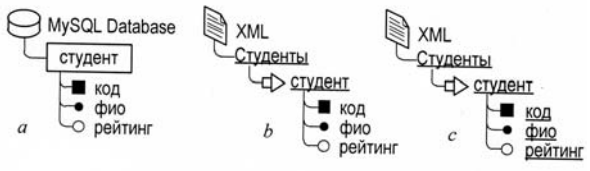

Рис. 2. Примеры моделей «плоских» XML-документов на основе данных из таблицы реляционной базы данных

a – модель таблицы
b – модель атрибутоцентричного XML-документа
c – модель элементоцентричного XML-документа

Вместе с тем, на уровне СОБД не трудно реализовать аналог простейшего, «плоского» варианта `SELECT FOR XML` – варианта `RAW` – и на этой основе обеспечить возможность формирования «на лету» XML-документов сложной иерархической структуры, используя имеющиеся возможности как собственно СОБД, так и XML-технологий, поддерживаемых в СОБД.

«Плоские» XML-документы, отображаемые в базу данных MySQL

Рассмотрим новые возможности СОБД по отображению «плоских» XML-документов на базу данных MySQL, проиллюстрировав их на простом примере ведения и обработки сведений о студентах (Рис. 2). Здесь и далее для графических моделей данных используется нотация, соответствующая работе [25].

Сведения о студентах физически хранятся в реляционной базе данных MySQL, а обрабатываются в СОБД как виртуальные XML-документы. В базе данных каждому студенту соответствует строка таблицы «`студент`», ячейки которой содержат сведения об этом студенте, например, `("s01", "Иванов", "20")`. В соответствующих «плоских» XML-документах каждому студенту соответствует XML-элемент «`студент`». В случае атрибутоцентричного документа сведения студента задаются в виде XML-атрибутов, имена которых совпадают с именами полей исходной таблицы:

```
<студент код = "s01" фио = "Иванов"
        рейтинг = "20"/>,
```

а в случае элементоцентричного – в виде XML-элементов, одноименных с полями таблицы:

```
<студент><код>s01</код><фио>Иванов</фио>
<рейтинг>20</рейтинг></студент>.
```

Извлечение виртуального документа их базы данных. На Рис. 3 представлен фрагмент HSM-модели, обеспечивающий загрузку в DOM-объект XML-документа, сформированного «на лету» из таблицы базы данных.

В состоянии `sta:MySQLi-Load` заданы два дочерних элемента.

1) Элемент `doc:Студенты` – определение виртуального XML-документа, отображаемого на базу данных MySQL. С помощью атрибута `action` задано соединение с базой данных (по умолчанию: хост – `localhost`; пользователь – `hsm`; пароль – `hsm`; база данных – `hsm`; порт и сокет – `NULL`). Атрибут `query` задает SQL-запрос к базе данных для извлечения строк таблицы `студент`. Атрибут `method` предписывает сформировать из полученных строк XML-документ в атрибутоцентричном стиле (`rowAttr`), представив значения полей в виде одноименных XML-атрибутов, при этом каждую строку обернуть XML-тегами `студент`, а документ в целом – тегами `Студенты`. Отметим, что для элементоцентричного документа в атрибуте `method` следовало указать значение «`rowElem`». По умолчанию для строк используется имя `row`, а для документа в целом – имя виртуального документа.

2) Элемент `dom:Студенты` – задание DOM-объекта для обработки XML-документа. Атрибут `srcDoc` (неявный источник данных) ссылается на виртуальный документ, содержимое которого следует загрузить в DOM-объект. При обработке этого элемента интерпретатор ситуационной модели обращается к спецификациям виртуального документа, в соответствии с которыми посылает SQL-запрос к базе данных, результат запроса оформляет в виде XML-документа и загружает его в DOM-объект для дальнейшей обработки. Дочерний приемник данных `rcv:Save` имитирует в данном примере обработку содержимого DOM-объекта – контент сохраняется в виде XML-файла `students.xml` в директории `TMP`.

Модификация виртуального документа в базе данных. Если виртуальный XML-документ отображается в XML-файл или целиком («сосредоточенным» способом) в ячейку таблицы базы данных, то при его модификации

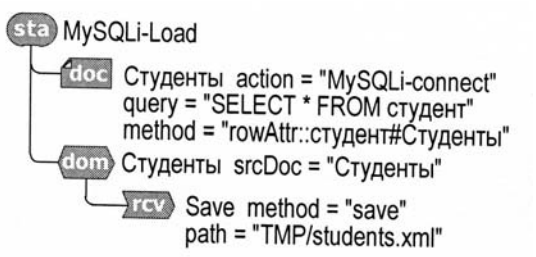

Рис. 3. Загрузка XML-документа в DOM-объект из таблицы базы данных

файл или ячейка просто переписываются. В случае «распределенного» способа модификация виртуального XML-документа может потребовать множество изменений в базе данных: вставку новых, обновление или удаление имеющихся строк таблицы.

Задачу такого рода часто решают на основе вспомогательных «корректировочных» документов, содержащих изменения одного вида, которые требуется внести в основной документ. Например, могут быть сформированы три XML-документа, содержащих соответственно сведения о студентах: новых (корректировки вставки), измененных (корректировки обновления) и отчисленных (корректировки удаления), на основании которых нужно скорректировать исходный XML-документ со сведениями о студентах.

Возможны два варианта решения этой задачи – на основе одного шаблона запроса к базе данных, общего для всех видов корректировок, и на основе нескольких шаблонов, специальных для разных видов корректировок. Шаблоны задаются в атрибутах `query` и `mquery` определения виртуального документа и используются интерпретатором для динамического формирования SQL-запросов к базе данных. Для записи шаблонов используется следующий синтаксис, являющийся модификацией случая «сосредоточенного» отображения [23, 24]:

вид`::`*таблица*`.`*поля*`[`*ключи*`]#`*метод*

вид – код вида запроса: `SEL` – выборка select, `INS` – вставка insert, `UPD` – обновление update, `DEL` – удаление delete (необязательный элемент, по умолчанию вид запроса назначается интерпретатором в контексте выполняемой операции);

таблица – имя таблицы базы данных, к которой обращен запрос;

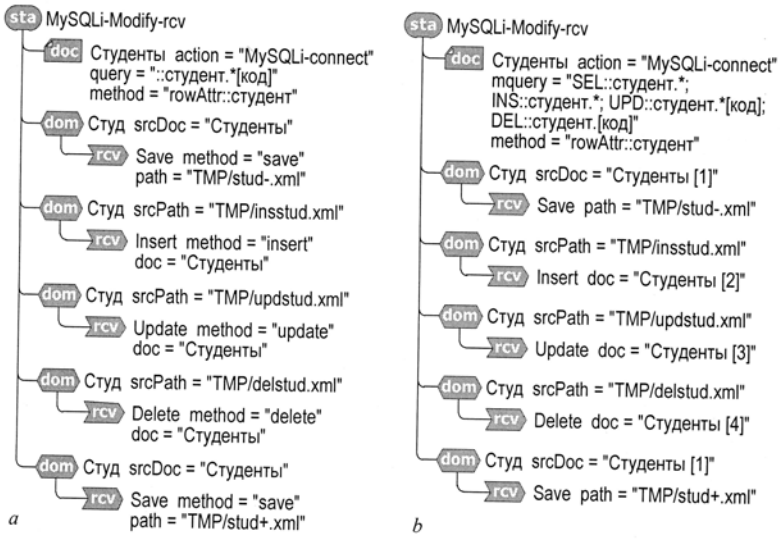

Рис. 4. Модификация таблиц базы данных из XML-документов

a – универсальный шаблон запроса и специализированные приемники данных
b – специализированные запросы и универсальные приемники данных

поля – список имен полей таблицы, участвующих в запросе (необязательный элемент, символ * означает все поля);

ключи – список имен полей, идентифицирующих строки таблицы (необязательный элемент);

метод – задает соответствие запроса к базе данных и виртуального XML-документа, например, стиль (атрибуто- или элементоцентричный), имена XML-тегов для каждой строки и для документа в целом (необязательный элемент, содержащиеся в нем сведения могут быть заданы с помощью отдельного атрибута `method`).

На Рис. 4 представлены два варианта HSM-модели, обеспечивающей модификацию виртуального документа со сведениями о студентах на основе «плоских» корректировочных XML-документов.

В обоих вариантах корректировки для вставки, обновления и удаления хранятся в директории `TMP` в файлах `insstud.xml`, `updstud.xml` и `delstud.xml` соответственно. Предполагается, что структура XML-документов в файлах корректировок соответствует структуре основного XML-документа (Рис. 2, *b*). Файлы корректировок последовательно загружаются в DOM-объект `dom:Студ`, откуда с помощью соответствующих приемников направляются в основной виртуальный документ `doc:Студенты` для модификации последнего.

Содержимое основного документа до и после модификации выводится для контроля в директорию `TMP` в виде файлов `stud-.xml` и `stud+.xml` соответственно.

В первом варианте (Рис. 4, *a*) в определении виртуального документа `doc:Студенты` специфицирован один общий шаблон запроса для отображения в базу данных MySQL:

```
::студент.*[код],
```

который отображает виртуальный документ на таблицу `студент`, при этом указывая, что в запросах следует принимать во внимание все поля таблицы, а для идентификации строк следует использовать поле `код`. Поскольку вид запроса в шаблоне не указан, он определяется в зависимости от инициировавшего запрос источника или приемника данных.

В случае источников данных подразумевается запрос выборки, и в рассматриваемом примере при загрузке DOM-объекта (атрибут `srcDoc` в первом и в последнем элементах `dom:Студ`) формируется SQL-запрос выборки

```
SELECT * FROM студент.
```

Атрибут `method` предписывает сформировать из результирующих строк XML-документ в атрибутоцентричном стиле, при этом каждую строку обернуть XML-тегами `студент`, а до-

кумент в целом – тегами Студенты – по умолчанию по имени виртуального документа.

В случае приемников данных вид формируемого запроса определяется значением атрибута method: insert – запрос вставки; update – запрос обновления; delete – запрос удаления.

В случае вставки формируется SQL-запрос вида

```
INSERT INTO студент (код, фио, рейтинг)
                    VALUES
('s02', 'Петров', '20'), ('s03', 'Си-
          доров', '10'), …
```

При этом в соответствии с атрибутом method список полей новых строк формируется как список имен XML-атрибутов в XML-элементе студент, а кортежи значений новых строк – как списки значений этих XML-атрибутов для каждого XML-элемента студент, обнаруженного в загруженном документе insstud.xml.

В случае обновления формируется серия SQL-запросов вида

```
UPDATE студент SET фио =
'Петрова' WHERE код = 's04'.
```

Запрос формируется согласно атрибуту method для каждого XML-элемента студент, обнаруженного в загруженном документе updstud.xml, при этом список обновляемых полей и идентификатор обновляемой строки строится на основе имеющихся XML-атрибутов элемента студент.

В случае удаления формируется SQL-запрос удаления строк вида

```
DELETE FROM студент WHERE код IN
       ('s08', 's09', …).
```

При этом в соответствии с атрибутом method список идентификаторов удаляемых строк формируется на основании XML-атрибутов в XML-элементах студент, обнаруженных в загруженном документе delstud.xml.

Во втором варианте (Рис. 4, *b*) doc:Студенты определен как виртуальный мультидокумент, в котором специфицированы четыре шаблона запросов для отображения в базу данных MySQL: для выборки, вставки, обновления и удаления соответственно. В этих условиях источники и приемники данных ссылаются на экземпляр мультизапроса, что освобождает от необходимости задавать атрибут method в приемниках данных.

Формирование «иерархических» XML-документов

Реляционные базы данных обычно содержат нормализованные таблицы, содержащие сведения об однотипных сущностях (entities) предметной области (например, о студентах, об изучаемых предметах, о сдачах студентами предметов и т.п.). Поэтому и «плоские» виртуальные документы, построенные на основе отдельных таблиц, тоже соответствуют однотипным сущностям. Приложениям же часто требуются сложные, «иерархические» XML-документы, содержащие иерархически организованные данные, относящиеся к сущностям различного типа, для которых «плоские» документы недостаточны. Например, могут требоваться не просто сведения о студентах, но и сведения о том, какие предметы сданы каждым студентом, а также сведения о самих предметах (Рис. 5).

На Рис. 5, *a* представлена реляционная модель базы данных, содержащей три нормализованных таблицы: студент – информация о студентах; предмет – информация о предметах; сдача – информация о сдачах студентами предметов и полученных при этом оценках. На Рис. 5, *b* представлены модели трех «плоских» XML-документов, соответствующих отдельным таблицам базы данных. Рис. 5, *c* представляет модель целевого «иерархического» XML-документа, содержащего данные из нескольких таблиц. Здесь в каждом XML-элементе студент, XML-атрибуты которого несут информацию о студенте, предусмотрены вложенные XML-элементы сдача, XML-атрибуты которых несут информацию о сдаче конкретным студентом конкретного предмета, а вложенный XML-элемент предмет содержит атрибуты, несущие информацию о сданном предмете.

Возможности XML-технологий, реализованных в СОБД, позволяют предложить два подхода к формированию XML-документов со сложной иерархической структурой на основе «плоских» XML-документов, извлеченных из базы данных.

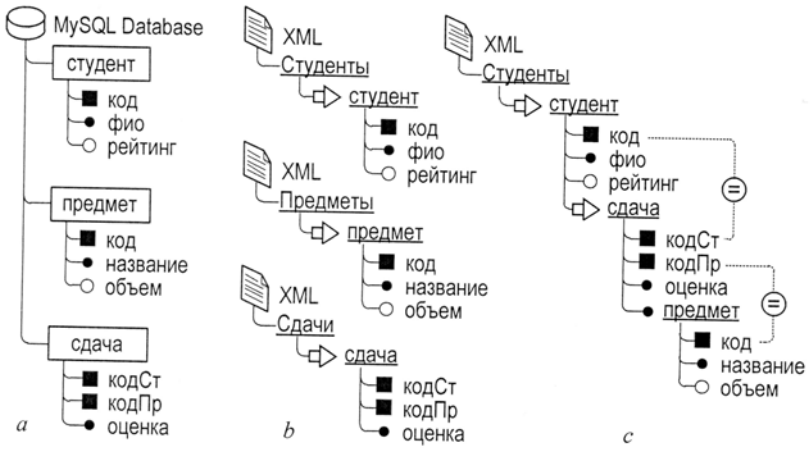

Рис. 5. Пример сложного XML-документа, отображаемого на несколько таблиц базы данных

a – таблицы базы данных
b – «плоские» XML-документы с данными из отдельных таблиц
c – «иерархический» XML-документ с данными из нескольких таблиц

1) Вложенные источники данных. В этом случае из базы данных с помощью нескольких простых (однотабличных) SQL-запросов извлекаются все необходимые сведения в виде набора «плоских» XML-документов. Эти документы загружаются в DOM-объект с помощью иерархически вложенных источников данных, в результате чего «на лету» формируется нужная иерархическая структура целевого XML-документа.

2) XSL-трансформация (XSLT). В этом случае из базы данных с помощью сложного (многотабличного) SQL-запроса извлекаются все необходимые сведения в виде «плоского» XML-документа. Этот документ загружается в DOM-объект и далее подвергается преобразованию по технологии XSLT для формирования нужной иерархической структуры целевого XML-документа.

Проиллюстрируем эти возможности на примере расширенных сведений о студентах (Рис. 5).

Использование вложенных источников данных. Пусть в базе данных имеются таблицы с информацией о студентах, о предметах и о сдачах студентами предметов (Рис. 5, *a*). Необходимо загрузить в DOM-объект XML-документ в соответствии с моделью Рис. 5, *c*. Фрагмент ситуационной модели `sta:MySQLi-Hierarchy-src` на Рис. 6 решает эту задачу.

Виртуальный мультидокумент `doc:Студенты` с помощью трех шаблонов-запросов задает отображение на таблицы базы данных `студент`, `сдача`, `предмет`. В каждом из шаблонов после символа # представлено значение атрибута `method` – указание сформировать «плоский» атрибутоцентричный XML-документ.

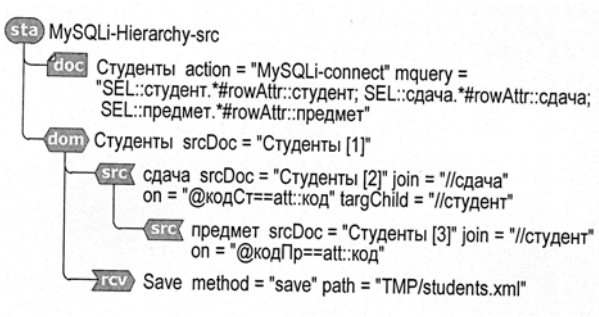

Рис. 6. Формирование «иерархического» XML-документа из таблиц базы данных с использованием вложенных источников данных

Элемент `dom:Студенты` порождает DOM-объект `Студенты` и загружает в него документ `Студенты` с данными о студентах, сформированный из таблицы `студент` (атрибут `srcDoc` – скрытый источник данных).

Далее источник данных `src:сдача` извлекает документ `Сдачи` с данными о сдачах, сформированный из таблицы `сдача`, и каждому узлу студента в ранее загруженном документе `Студенты` добавляет новый уровень иерархии – дочерние узлы сдач этого студента. Для этого в источнике предусмотрены атрибуты:

`join` – задает XPath-выражение, адресующее множество узлов в дочернем документе, в котором ищутся узлы для присоединения к узлам в родительском документе, в данном случае это множество XML-элементов `сдача` в документе `Сдачи`;

`targChild` – задает XPath-выражение, адресующее множество узлов в родительском документе, для каждого из которых ищутся присоединяемые узлы в дочернем документе, в данном случае это множество XML-элементов `студент` в документе `Студенты`;

`on` – задает условие соединения, которому должен удовлетворять дочерний узел для присоединения к родительскому узлу, в данном случае это требование, чтобы атрибут `кодСт` узла `сдача` совпадал с атрибутом `код` узла `студент`.

Далее аналогичным образом действует источник данных `src:предмет` извлекает документ `Предметы` с данными о предметах и для каждого узла `сдача` в уже загруженном документе добавляет новый уровень иерархии – дочерний узел `предмет` о предмете этой сдачи.

Таким образом, в DOM-объекте `dom:Студенты` сформирован требуемый «иерархический» XML-документ, соответствующий Рис. 5, *c*. Предусмотренный в dom-элементе приемник данных `rcv:Save` выполняет тестовое сохранение контента DOM-объекта в файле `TMP/students.xml`.

Использование XSL-трансформации. Хотя SQL-запросы выборки возвращают результат всегда в виде «плоской» таблицы, результирующая таблица может быть сформирована на основе нескольких таблиц базы данных. С помощью многотабличного SQL-запроса можно извлечь все необходимые сведения в виде «плоского» XML-документа, загрузить его в DOM-объект и далее подвергнуть XSLT-преобразованию для формирования нужной иерархической структуры целевого XML-документа.

Проиллюстрируем эти возможности на том же примере расширенных сведений о студентах. Для формирования целевого документа (Рис. 5, *c*) необходимы данные из трех таблиц базы данных. На Рис. 7, a представлена модель многотабличного запроса выборки этих данных посредством внутреннего эквисоединения (inner equijoin)

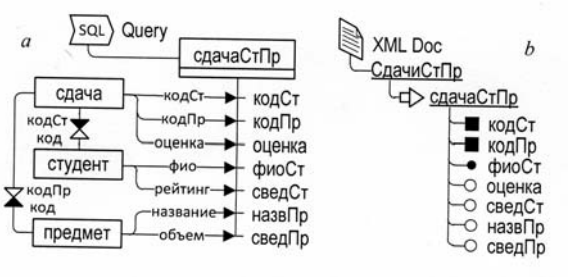

Рис. 7. Формирование «плоского» XML-документа из нескольких таблиц базы данных

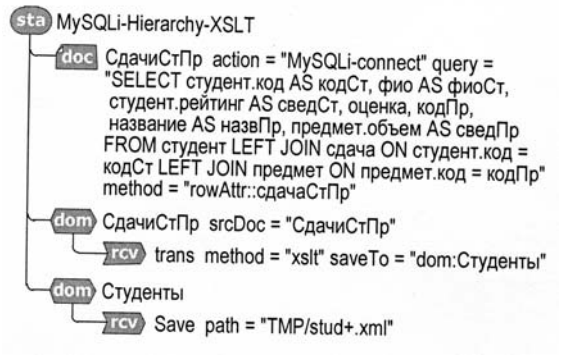

Рис. 8. Формирование «иерархического» XML-документа путем XSL-трансформации

таблицы `сдача`, с одной стороны, с таблицей `студент` (по равенству кодов студентов), а с другой стороны, с таблицей `предмет` (по равенству кодов предметов). Каждая строка результирующей таблицы `сдачаСтПр` («сдача студентом предмета») содержит информацию об одной сдаче, дополненную информацией о сдавшем студенте и о сданном предмете. Соответствующая модель «плоского» XML-документа приведена на Рис. 7, *b*.

Фрагмент ситуационной модели `sta:MySQLi-Hierarchy-XSLT` на Рис. 8 решает задачу извлечения из базы данных «плоского» XML-документа и преобразования его в «иерархический».

Виртуальный документ `doc:СдачиСтПр` в атрибуте `query` определяет многотабличный SQL-запрос формирования результирующей таблицы в соответствии с моделью на рис. 7, *a*. Атрибут `method` предписывает оформить результат в виде «плоского» атрибутоцентричного XML-документа. Этот документ извлекается из базы данных и загружается в DOM-объект

dom:СдачиСтПр. Приемник данных rcv:trans подвергает содержимое DOM-объекта XSL-трансформации и записывает результат в другой DOM-объект – dom:Студенты.

XSL-трансформация выполняется в соответствии со стилевой таблицей (Style sheet), которая по умолчанию хранится в директории XSL в файле trans.xsl (в соответствии с именем приемника данных). Модель стилевой таблицы приведена на Рис. 9. Преобразование запрограммировано в стиле «извлекающей» трансформации, для группирования сдач предметов, соответствующих одному студенту, применен метод Мюнча [23, гл. 4].

Таким образом, в DOM-объекте dom:Студенты сформирован требуемый «иерархический» XML-документ, соответствующий Рис. 5, *с*. Предусмотренный в dom-элементе приемник данных rcv:Save, как и в предыдущем примере, выполняет тестовое сохранение контента DOM-объекта в файле TMP/students.xml.

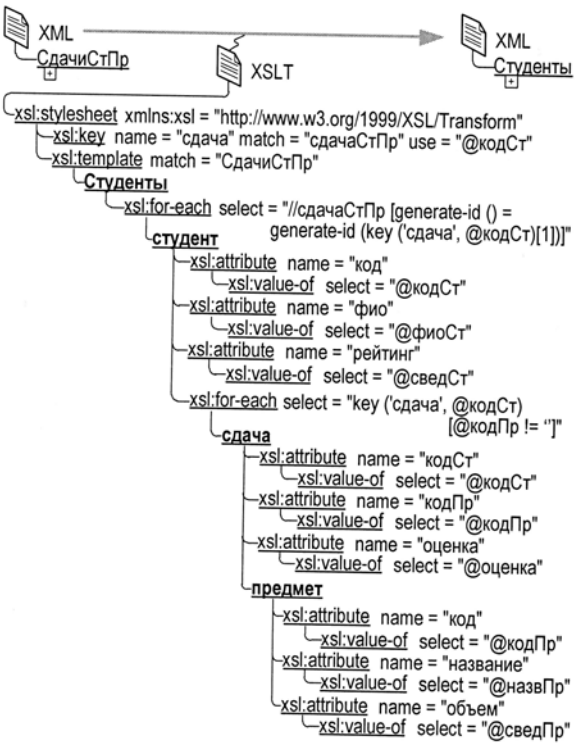

Рис. 9. Стилевая таблица XSL-трансформации для преобразования «плоского» XML-документа в «иерархический»

Реализация на платформе PHP

Исследовательский прототип СОБД реализован на платформе PHP в составе «движка» для управления веб-приложениями на веб-сервере Apache. Для связи с базами данных сервера MySQL используется модуль расширения mysqli [26, 30, 31, 33, 34]. Для реализации рассмотренной выше функциональности применяется взаимодействие через глобальные массивы. При обработке ситуационной модели для каждого виртуального документа, заданного в текущем состоянии, в оперативной памяти интерпретатором создается одноименный глобальный массив, в который помещаются параметры и объекты, необходимые для доступа к реальным документам. При дальнейшей обработке источников и приемников данных, ссылающихся на виртуальный документ, интерпретатор обращается к соответствующему глобальному массиву, получая тем самым сведения, необходимые для доступа к реальным документам. Ошибки, обнаруженные при обработке виртуальных документов, также отражаются в глобальном массиве – массиве ошибок, наличие которого позволяет программировать в ситуационной модели реакцию на ошибки – изменять текущее состояние модели, оповещать пользователя об ошибке и т.д.

Рассмотренная в статье функциональность СОБД была реализована, отлажена и протестирована на приведенных выше примерах в исследовательском прототипе интерпретатора ситуационных моделей. В настоящее время ведется работа по использованию этих возможностей в реальном проекте веб-приложения.

Заключение

1. Ситуационно-ориентированные базы данных (СОБД), основываясь на понятии виртуальных документов, позволяют обрабатывать XML-данные, реально хранящиеся в таблицах реляционной базы данных в виде XML-документов («сосредоточенный» подход) или в виде данных, размещенных во множестве строк различных таблиц («распределенный» подход).

2. Для реализации «распределенного» подхода целесообразно результат SQL-запроса к базе данных сопоставлять с «плоским» вирту-

альным XML-документом, который в дальнейшем преобразуется в «иерархический» XML-документ средствами СОБД. Извлечения «плоского» XML-документа из базы данных, а также модификация базы данных на этой основе выполняется с помощью SQL-запросов, формируемых интерпретатором на основе шаблонов, задаваемых в определении виртуального XML-документа.

3. Для получения «иерархических» XML-документов на основе «плоских», извлеченных из базы данных, используются два подхода – на основе вложенных источников данных, когда «иерархический» документ собирается «на лету» из «плоских» в процессе загрузки DOM-объекта, и на основе XSL-трансформации, когда «плоский» документ загружается в DOM-объект и там подвергается преобразованию по технологии XSLT.

4. Практическая реализация предложенных решений выполнена для популярной свободно распространяемой «связки»: веб-сервер Apache, язык серверных сценариев PHP, сервер реляционных баз данных MySQL, при этом для связи с базой данных применялся модуль PHP-расширения mysqli.

Литература

1. Agustin J.L.H., Del Barco P.C. A model-driven approach to develop high performance web applications. In: J. of Systems and Software. Vol. 86, no. 12, pp. 3013–3023 (2013).
2. Pinheiro P.V.P., Endo A.T., Simao A. Model-Based Testing of RESTful Web Services Using UML Protocol State Machines. In: Brazilian Workshop on Systematic and Automated Software Testing (2013).
3. Daniel F., Matera M. Model-Driven Software Development. In: Mashups, pp. 71–93. Springer, Berlin, Heidelberg (2014)
4. Aguilar J. A. et al. An Analysis of Techniques and Tools for Requirements Elicitation in Model-Driven Web Engineering Methods In: Computational Science and Its Applications ICCSA 2015, pp. 518–527. Springer International Publishing (2015).
5. Delgado A., Marotta A., González L. Towards the construction of quality-aware W eb Warehouses with BPMN 2.0 Business Processes. In: IEEE Eighth International Conference on Research Challenges in Information Science (RCIS'2014), pp. 1–6. IEEE (2014).
6. Delgado A., Marotta A. Automating the process of building flexible Web Warehouses with BPM Systems. In: Computing Conference (CLEI), 2015 Latin American, pp. 1–11. IEEE Press (2015).
7. Pokorny J. NoSQL databases: a step to database scalability in web environment. In: Int. J. of Web Information Systems. Vol. 9, no. 1, pp. 69–82 (2013).
8. Bugiotti, F., Cabibbo, L., Atzeni, P., Torlone, R. Database design for NoSQL systems. In: Conceptual Modeling, pp. 223–231. Springer International Publishing (2014).
9. Zhang S. Application of document-oriented NoSQL database technology in web-based software project documents management system. In: IEEE International Conference on Information Science and Technology (ICIST), pp. 504–507 (2013).
10. Фаулер М., Садаладж П. Дж. NoSQL: новая методология разработки нереляционных баз данных. Пер. с англ. М.: Вильямс, 2013. 192 с.
11. Миронов В.В., Юсупова Н.И., Гусаренко А.С. Ситуационно-ориентированные базы данных: современное состояние и перспективы исследования // Вестник УГАТУ. 2015. Т. 19, № 2 (68). С. 188–199.
12. Миронов В.В., Гусаренко А..С. Ситуационно-ориентированные базы данных: концепция управления XML-данными на основе динамических DOM-объектов // Вестник УГАТУ. 2012. Т. 16, № 3 (48). С. 159–172.
13. Миронов В.В., Гусаренко А.С. Динамические DOM-объекты в ситуационно-ориентированных базах данных: лингвистическое и алгоритмическое обеспечение источников данных // Вестник УГАТУ. 2012. Т. 16, № 6 (51). С. 167–176
14. Гусаренко А. С., Миронов В. В. Smarty-объекты: вариант использования гетерогенных источников в ситуационно-ориентированных базах данных // Вестник УГАТУ. 2014. Т. 18, № 3 (63). С. 242–252.
15. Гусаренко А.С., Миронов В.В. Использование RESTful-сервисов в ситуационно-ориентированных базах данных // Вестник УГАТУ. 2015. Т. 19, № 1 (67). С. 204–211.
16. Миронов В.В., Гусаренко А.С., Диметриев Р.Р., Сарваров М.Р. Создание персонализированных документов на основе ситуационно-ориентированной базы данных // Вестник УГАТУ. 2014. Т. 18, № 4 (65). С. 191–197.
17. Гусаренко А. С. Модели создания документов в формате Office Open XML на основе ситуационно-ориентированной базы данных // Прикладная информатика. 2015. Т. 10, № 3. С. 62–75.
18. Арлазаров В. Л., Емельянов Н. Е. Документооборот как информационная база накопления знаний // Труды Института системного анализа Российской академии наук. 2007. Т. 29. С. 6–48.
19. Долгоруков А. Ю., Ерохин В. И. Язык запросов хранилища документов, построенного на НИКА-технологии // Труды Института системного анализа Российской академии наук. 2010. Т. 58. С. 48–60.
20. Емельянов Н.Е., Тищенко В.А. Методология построения многоуровневого индекса ключевого массива по лексикографическому признаку на основе метода регрессионного анализа на примере СУБД НИКА // Труды Института системного анализа Российской академии наук. 2010. Т. 58. С. 6–17.

21. Фомичев А.В. Управление памятью в XML-ориентированной СУБД Sedna. http://synthesis.ipi.ac.ru/sigmod/seminar/s20050127/
22. Ilya Taranov, Ivan Shcheklein, Alexander Kalinin, et al. Sedna: native XML database management system (internals overview) // SIGMOD '10: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. New York, NY, USA: ACM, 2010. Pages 1037–1046.
23. Миронов В. В., Юсупова Н. И., Шакирова Г. Р. Иерархические модели данных: концепции и реализация на основе XML / под ред. проф. Н. И. Юсуповой. М.: Машиностроение, 2011. 453 с.
24. Гусаренко А.С. Усовершенствование модели ситуационно-ориентированной базы данных для взаимодействия с MySQL // Известия высших учебных заведений. Приборостроение. 2016. Т. 59. № 5. С. 355–363.
25. Гусаренко А.С., Миронов В.В. Гетерогенные источники документов в ситуационно-ориентированных базах данных // Вестник УГАТУ. 2015. Т. 19. № 4. С. 124–131.
26. Канашин В.В., Миронов В.В. Иерархические виджеты: организация интерфейса пользователя в веб-приложениях на основе ситуационно-ориентированных баз данных // Вестник УГАТУ. 2013. Т. 17, № 2 (55). С. 138–149.
27. Канашин В.В., Миронов В.В. Иерархические виджеты: ввод и контроль данных пользователя в веб-приложениях на основе ситуационно-ориентированных баз данных // Вестник УГАТУ. 2013. Т. 17, № 5 (58). С. 166–176.
28. Канашин В.В., Миронов В.В. Иерархические виджеты: алгоритмы контроля данных пользователя в веб-приложениях на основе ситуационно-ориентированных баз данных // Вестник УГАТУ. 2014. Т. 18, № 1 (62). С. 204–213.
29. Канашин В.В., Миронов В.В. Иерархические виджеты: опыт применения в веб-приложении на основе ситуационно-ориентированной базы данных // Вестник УГАТУ. 2014. Т. 18, № 2 (63). С. 185–196.
30. Макарова Е.С., Миронов В.В. Проектирование концептуальной модели данных для задач Web-OLAP на основе ситуационно-ориентированной базы данных // Вестник УГАТУ. 2012. Т. 16, № 6 (51). С. 177–188.
31. Макарова Е.С., Миронов В.В. Функции аналитики в веб-приложениях на основе ситуационно-ориентированных баз данных // Вестник УГАТУ. 2013. Т. 17, № 5 (58). С. 150–165.
32. Dejanovic I., Milosavljevic G., Perisic B., Tumbas M. A. Domain-specific language for defining static structure of database applications. In Computer Science and Information Systems, Vol. 7, No. 3, P. 409–440, 2010.
33. Djukic V. и др. Model Execution: An Approach based on extending Domain-Specific Modeling with Action Reports // Computer Science and Information Systems. 2013. T.10. № 4. С. 1585–1620.
34. Руководство по PHP. Улучшенный модуль MySQL [Электронный ресурс]. URL: http://php.net/manual/ru/book.mysqli.php (дата обращения 17 июля 2016 г.).

Миронов Валерий Викторович. Профессор кафедры автоматизированных систем управления Федерального государственного бюджетного образовательного учреждения высшего образования «Уфимский государственный авиационный технический университет» (ФГБОУ ВО УГАТУ). Окончил Воронежский государственный университет в 1975 году. Доктор технических наук. Количество печатных работ: 50. Область научных интересов: исследования в области иерархических моделей и ситуационного управления. E-mail: mironov@ugatu.su

Гусаренко Артем Сергеевич. Старший преподаватель кафедры автоматизированных систем управления ФГБОУ ВО УГАТУ. Окончил Уфимский государственный авиационный технический университет в 2010 году. Кандидат технических наук. Количество печатных работ: 25. Область научных интересов: исследования в области ситуационно-ориентированных баз данных. E-mail: gusarenko@ugatu.su

Юсупова Нафиса Исламовна. Декан факультета информатики и робототехники ФГБОУ ВО УГАТУ. Окончила Воронежский государственный университет в 1975 году. Доктор технических наук, профессор. Количество печатных работ: 135. Область научных интересов: исследования в области иерархических моделей, ситуационного управления в технических и социальных системах. E-mail: yusupova@ugatu.ac.ru

Displaying virtual XML-documents on MySQL tables in the situation-oriented databases, "distributed" approach

V.V. Mironov, A.S. Gusarenko, N.I. Yusupova

Abstract: Situation-oriented database (SODB) represents the information processor for processing the XML-documents as part of a Web application that is running by means of built-in situational models. Considered SOBD architecture with a virtual document repository, which contains the XML-data and virtual documents processed by the data processing objects DPO. The document repository can store heterogeneous data as files, web services, the zip-archives and DBMS. Considered SODB integration with relational environment MySQL as a source of XML-Data - XML-documents are physically stored in a MySQL database, how to handle the process of loading data in DOM-objects created by the interpreter situational model. Proposed linguistic tools for specifying in the situational model for displaying XML-data in a MySQL table. In SODB there are several types of virtual documents dom-element, src-element, rcv-element, doc-element. These types of elements are used to display documents from the file system in a DBMS table. Discussed the practical implementation of the approach using PHP platform and the mysqli extension, and its application in web applications.

Keywords: situation-oriented database; web-application; model-driven approach; hierarchical situation model; virtual document; relational data; data integration; HSM; NoSQL; XML; DOM; MySQL; PHP; mysqli.

References

1. Agustin J.L.H., Del Barco P.C. A model-driven approach to develop high performance web applications. In: J. of Systems and Software. Vol. 86, no. 12, pp. 3013–3023 (2013).
2. Pinheiro P.V. P., Endo A.T., Simao A. Model-Based Testing of RESTful Web Services Using UML Protocol State Machines. In: Brazilian Workshop on Systematic and Automated Software Testing (2013).
3. Daniel, F., & Matera, M. Model-Driven Software Development. In: Mashups, pp. 71–93. Springer, Berlin, Heidelberg (2014)
4. Aguilar J.A. et al. An Analysis of Techniques and Tools for Requirements Elicitation in Model-Driven Web Engineering Methods In: Computational Science and Its Applications ICCSA 2015, pp. 518–527. Springer International Publishing (2015).
5. Delgado A., Marotta A., González L. Towards the construction of quality-aware Web Warehouses with BPMN 2.0 Business Processes. In: IEEE Eighth International Conference on Research Challenges in Information Science (RCIS'2014), pp. 1–6. IEEE (2014).
6. Delgado A., Marotta A. Automating the process of building flexible Web Warehouses with BPM Systems. In: Computing Conference (CLEI), 2015 Latin American, pp. 1–11. IEEE Press (2015).
7. Pokorny J. NoSQL databases: a step to database scalability in web environment. In: Int. J. of Web Information Systems. Vol. 9, no. 1, pp. 69–82 (2013).
8. Bugiotti, F., Cabibbo, L., Atzeni, P., Torlone, R. Database design for NoSQL systems. In: Conceptual Modeling, pp. 223–231. Springer International Publishing (2014).
9. Zhang S. Application of document-oriented NoSQL database technology in web-based software project documents management system. In: IEEE International Conference on Information Science and Technology (ICIST), pp. 504–507 (2013).
10. Fowler M., Sadalage P.J. NoSQL: novaja metodologija razrabotki nereljacionnyh baz dannyh. Moscow: Vil'jams, 2013. 192 pp. (NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley, 2013.)
11. Mironov V.V., Yusupova N.I., Gusarenko A.S. Situacionno-orientirovannye bazy dannyh: sovremennoe sostojanie i perspektivy issledovanija // Vestnik UGATU. 2015. V. 19, no. 2 (68). pp. 188–199. (Situation-oriented databases: current status and prospects for research. In: Vestnik UGATU. 2015. V. 19, no. 2 (68). pp. 188-199.)
12. Mironov V.V., Gusarenko A.S. Situacionno-orientirovannye bazy dannyh: koncepcija upravlenija XML-dannymi na osnove dinamicheskih DOM-objektov // Vestnik UGATU. 2012. V. 16, no. 3 (48). pp. 159–172. (Situation-oriented databases: The concept of XML-data management based on the dynamic DOM-objects. In: Vestnik UGATU. 2012. V. 16, no. 3 (48). pp. 159-172.)
13. Mironov V. V., Gusarenko A. S. Dinamicheskie DOM-objekty v situ-acionno-orientirovannyh bazah dannyh: lingvistich-eskoe i algoritmicheskoe obespechenie istochnikov dannyh // Vestnik UGATU. 2012. V. 16, no. 6 (51). pp. 167–176. (Dynamic the DOM-objects in the situation-oriented databases: linguistic and algorithmic support of data sources. In: Vestnik UGATU. 2012. V. 16, no. 6 (51). pp. 167-176.)
14. Gusarenko A.S., Mironov V. V. Smarty-objekty: variant ispol'zovanija geterogennyh istochnikov v situacionno-orientirovannyh bazah dannyh // Vestnik UGATU. 2014. V. 18, no. 3 (63). pp. 242–252. (Smarty-objects: using of heterogeneous sources in a situation-oriented databases. In: Vestnik UGATU. 2014. V. 18, no. 3 (63). pp. 242-252).
15. Gusarenko A.S., Mironov V.V. Ispol'zovanie RESTful-servisov v situacionno-orientirovannyh bazah dannyh // Vestnik UGATU. 2015. V. 19, no. 1 (67). pp. 204–211. (Using of a RESTful-services in situation-oriented databases. In: Vestnik UGATU. 2015. V. 19, no. 1 (67). pp. 204-211.)

16. Mironov V.V., Gusarenko A.S., Dimetriev R.R., Sarvarov M. R. Soz-danie personalizirovannyh dokumentov na osnove situacionno-orientirovannoj bazy dannyh // Vestnik UGATU. 2014. V. 18, no. 4 (65). pp. 191–197. (Creating personalized documents based on situation-oriented database. In: Vestnik UGATU. 2014. V. 18, no. 4 (65). pp. 191-197.)
17. Gusarenko A.S. Modeli sozdanija dokumentov v formate Office Open XML na osnove situacionno-orientirovannoj bazy dannyh // Prikladnaja informatika. 2015. V. 10, no. 3. pp. 62–75. (Models for creation documents in Office Open XML format based on situation-oriented database. In: Applied Informatics. 2015. V. 10, no. 3. pp. 62-75.)
18. Arlazarov V.L., Emel'janov N. E. Dokumentooborot kak informacionnaja baza nakoplenija znanij // Trudy Instituta sistemnogo analiza Rossijskoj akademii nauk. 2007. T. 29. S. 6–48.
19. Dolgorukov A.Ju., Erohin V. I. Jazyk zaprosov hranilishha dokumentov, postroennogo na NIKA-tehnologii // Trudy Instituta sistemnogo analiza Rossijskoj akademii nauk. 2010. T. 58. S. 48–60.
20. Emel'janov N.E., Tishhenko V. A. Metodologija postroenija mnogourovnevogo indeksa kljuchevogo massiva po leksikograficheskomu priznaku na osnove metoda regressionnogo analiza na primere SUBD NIKA // Trudy Instituta sistemnogo analiza Rossijskoj akademii nauk. 2010. T. 58. S. 6–17.
21. Fomichev A.V. Upravlenie pamjat'ju v XML-orientirovannoj SUBD Sedna. http://synthesis.ipi.ac.ru/sigmod/seminar/s20050127/
22. Ilya Taranov, Ivan Shcheklein, Alexander Kalinin, et al. Sedna: native XML database management system (internals overview) // SIGMOD '10: Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. New York, NY, USA: ACM, 2010. Pages 1037–1046.
23. Mironov V.V., Yusupova N.I., Shakirova G.R. Ierarhicheskie modeli dannyh: koncepcii i realizacija na osnove XML / pod red. prof. N. I. Yusupovoj. Moscow: Mashinostroenie, 2011. 453 p. (Hierarchical data models: concepts and implementation based on XML. Ed. by Prof. N. I. Yusupova. Moscow: Mashinostroenie, 2011.)
24. Gusarenko A. S. Usovershenstvovanie modeli situacionno-orientirovannoj bazy dannyh dlja vzaimodejstvija s MySQL // Izvestija vysshih uchebnyh zavedenij. Priborostroenie. 2016. T. 59. № 5. C. 355–363.
25. Gusarenko A.S., Mironov V.V. Geterogennye istochniki dokumentov v situacionno-orientirovannyh bazah dannyh // Vestnik UGATU. 2015. V. 19. № 4. P. 124–131.
26. Kanashin V.V., Mironov V.V. Ierarhicheskie vidzhety: organizacija interfejsa pol'zovatelja v veb-prilozhenijah na osnove situacionno-orientirovannyh baz dannyh // Vestnik UGATU. 2013. V. 17, № 2 (55). P. 138–149.
27. Kanashin V.V., Mironov V.V. Ierarhicheskie vidzhety: vvod i kontrol' dannyh pol'zovatelja v veb-prilozhenijah na osnove situacionno-orientirovannyh baz dannyh // Vestnik UGATU. 2013. T. 17, № 5 (58). C. 166–176.
28. Kanashin V.V., Mironov V.V. Ierarhicheskie vidzhety: algoritmy kontrolja dannyh pol'zovatelja v veb-prilozhenijah na osnove situacionno-orientirovannyh baz dannyh // Vestnik UGATU. 2014. T. 18, № 1 (62). C. 204–213.
29. Kanashin V.V., Mironov V.V. Ierarhicheskie vidzhety: opyt primenenija v veb-prilozhenii na osnove si-tuacionno-orientirovannoj bazy dannyh // Vestnik UGATU. 2014. T. 18, № 2 (63). C. 185–196.
30. Makarova E.S., Mironov V.V. Proektirovanie konceptual'noj modeli dannyh dlja zadach Web-OLAP na osnove situacionno-orientirovannoj bazy dannyh // Vestnik UGATU. 2012. T. 16, № 6 (51). C. 177–188.
31. Makarova E.S., Mironov V.V. Funkcii analitiki v veb-prilozhenijah na osnove situacionno-orientirovannyh baz dannyh // Vestnik UGATU. 2013. V. 17, № 5 (58). P. 150–165.
32. Dejanovic I., Milosavljevic G., Perisic B., Tumbas M. A. Domain-specific language for defining static structure of database applications. In Computer Science and Information Systems, Vol. 7, No. 3, P. 409–440, 2010.
33. Djukic V. et. al. Model Execution: An Approach based on extending Domain-Specific Modeling with Action Reports // Computer Science and Infor-mation Systems. 2013. T. 10. № 4. C. 1585–1620.
34. Rukovodstvo po PHP. Uluchshennyj modul' MySQL [Jelektronnyj resurs]. URL: http://php.net/manual/ru/book.mysqli.php (data obrashhenija 17 maja 2016 y.) (PHP Manual. Improved MySQL module [electronic resource]. URL: http://php.net/manual/ru/book.mysqli.php (reference date of May 17, 2016))

Mironov Valeriy Viktorovich. Dr. Tech. Sciences, Professor, Department of automated control systems, Qualified radio physics (Voronezh State University, 1975). Dr. Tech. Sciences in technical systems management (USATU, 1995). Ufa State Aviation Technical University. Research interests: Research in hierarchical models and situational management. The number of publications: 50. E-mail: mironov@ugatu.su

Gusarenko Artem Sergeevich. Ph.D., Senior Lecturer, Department of automated control systems, Qualified informatics economist (USATU, 2010), Ph.D. Tech. Sciences in mathematical and software of computers, complexes and computer networks (USATU 2013). Ufa State Aviation Technical University. Research interests: Research in situation-oriented databases. The number of publications: 25. E-mail: gusarenko@ugatu.su

Yusupova Nafisa Islamovna. Dr. Tech. Sciences, professor, faculty dean of computer science and robotics. Qualified radio physics (Voronezh State University, 1975). Dr. Tech. Sciences in technical systems management (USATU, 1997). Research interests: Research in hierarchical models of situational management in technical and social systems. Ufa State Aviation Technical University. The number of publications: 135. E-mail: yusupova@ugatu.ac.ru