ISSN 1992-6502 (Print) Вестник УГАТУ ISSN 2225-2789 (Online)
2014. Т. 18, № 4 (65). С. 191–197 http://journal.ugatu.ac.ru

УДК 004.65:004.7

СОЗДАНИЕ ПЕРСОНАЛИЗИРОВАННЫХ ДОКУМЕНТОВ НА ОСНОВЕ СИТУАЦИОННО-ОРИЕНТИРОВАННОЙ БАЗЫ ДАННЫХ

В. В. МИРОНОВ[1], А. С. ГУСАРЕНКО[2], Р. Р. ДИМЕТРИЕВ[3], М. Р. САРВАРОВ[4]

[1] mironov@list.ru, [2] artyomgusarenko@gmail.com, [3] dimetriev.roman@gmail.com, [4] shtirlizc@gmail.com

ФГБОУ ВПО «Уфимский государственный авиационный технический университет» (УГАТУ)

Поступила в редакцию 14 июня 2014 г.

Аннотация. Обсуждается технология обработки XML-документов в ситуационно-ориентированных базах данных (СОБД) для создания персонализированных документов на веб-сервере по запросам клиентов. Документы генерируются на основе заранее подготовленных шаблонов-заготовок, находящихся в хранилище документов СОБД; в ходе интерпретации динамической модели (HSM) шаблоны загружаются в DOM-объекты, наполняются персональным контентом в соответствии с запросом и пересылаются клиенту, выдавшему запрос. Рассматриваются спецификации и новые элементы HSM для задания операций персонализации. Описываются средства обработки zip-архивов для работы с шаблонами в формате Open XML. Иллюстрируется применение технологии для создания документов в форматах vdx и docx. Описывается практическое применение результатов в учебном процессе для создания документации в ходе курсового проектирования.

Ключевые слова: веб-приложение; база данных; динамическая модель; СОБД; XML; DOM.

ВВЕДЕНИЕ

Современным веб-приложениям часто приходится решать задачу создания на веб-сервере и отправки клиентам электронных документов, соответствующих запросу. Для решения задач подобного рода имеются многочисленные серверные средства генерации электронных документов в различных форматах[1].

Задача упрощается, когда документ нужно создать не «с нуля», а путем «персонализации» заранее подготовленного шаблона-заготовки, в определенные места которого требуется поместить соответствующие клиенту данные [1–4].

В данной статье эта задача рассматривается применительно к веб-приложениям, построенным на основе ситуационно-ориентированных баз данных (СОБД) [5]. СОБД представляет собой хранилище документов в XML-формате, управление которым производится на основе встроенной иерархической динамической модели (HSM – Hierarchical State Model).

В исследованиях по СОБД упор делался на экранный пользовательский интерфейс того или иного вида [6–13]. Вместе с тем вопросы генерации электронных документов, которые можно распечатать на бумаге в виде отчетов высокого полиграфического качества, применительно к СОБД не рассматривались.

Обработка данных в тех или иных состояниях СОБД задается с помощью dom-элементов [14–15], специфицирующих создание DOM-объектов, загрузку в них XML-документов из хранилища, обработку XML-документов в DOM-объектах, выгрузку содержимого DOM-объектов в хранилище или отправку их клиенту.

Электронные документы – текстовые (например, создаваемые для распространенного текстового редактора Word) или графические (например, для популярного графического редактора Visio) – могут быть представлены в форматах на основе XML, что позволяет обрабатывать их в СОБД как обычные XML-документы. При этом «старые» форматы, такие как xml, vdx, позволяют использовать документы в СОБД непосредственно. В «новых» форматах, таких как docx и xsdx, соответствующих стандарту Office Open XML, документ представляет собой zip-архив, содержащий файлы XML. В этих случаях СОБД должна иметь воз-

Работа поддержана грантом РФФИ 10-07-00167-а.

[1] http://msdn.microsoft.com/en-us/library/ee895050(v=office.12).aspx
http://phpword.codeplex.com/
http://docx.codeplex.com/

можность разархивировать нужные файлы при загрузке их в DOM-объекты и сохранять их в архиве при выгрузке.

ТРЕБУЕМАЯ ФУНКЦИОНАЛЬНОСТЬ

Таким образом, для создания персонализированных документов СОБД должна обеспечивать следующую функциональность:

- загрузку шаблона-заготовки документа в DOM-объект в формате XML – целиком или какой-то частью;
- отыскание в дереве загруженного шаблона нужных узлов, которые необходимо наполнить контентом;
- обновление найденных узлов – изменение их содержимого;
- выгрузку наполненного контентом шаблона в хранилище документов;
- отправку персонализированного документа клиенту.

Обсудим эти вопросы для документов в формате XML и в формате Office Open XML.

ДОКУМЕНТЫ В ФОРМАТЕ XML

Загрузка. Шаблоны-заготовки персонализируемых документов в формате XML могут непосредственно загружаться в DOM-объекты, специфицированные в HSM. Например, следующий HSM-фрагмент представляет состояние sta:Content-schema, в котором создается DOM-объект dom:Schema и в него загружается Visio-файл Blanks/schema.vdx:

```
<sta:Content-schema>
    <doc:Schema path = "Blanks/schema.vdx"/>
    <dom: Schema srcDoc = "Schema"/>
</sta:Content-schema>
```

Здесь doc-элемент задает путь к загружаемому файлу в хранилище документов СОБД, а dom-элемент, ссылающийся на doc-элемент – создание и загрузку DOM-объекта.

Выгрузка и отправка. Эти задачи также решаются на основе уже имеющейся функциональности HSM. В следующем фрагменте содержимое DOM-объекта dom:Schema выгружается во временное хранилище документов, после чего документ отправляется пользователю и удаляется из временного хранилища:

```
<doc:TMP-Schema path = "TMP/schema.vdx"/>
<dom:Schema>
    <rcv:SaveSchema method = "save"
        doc = "TMP-Schema"/>
</dom:Schema>
<doc:TMP-Schema action = "send"/>
<doc:TMP-Schema action = "drop"/>
```

Элемент-приемник rcv:SaveSchema предписывает сохранить содержимое DOM-объекта по адресу, заданному doc-элементом doc:TMP-Schema.

Содержимое DOM-объекта может быть отправлено пользователю сразу, минуя временное хранилище:

```
<dom:Schema>
    <rcv:SaveSchema method = "send"
        file = "schema.vdx"/>
</dom:Schema>
```

Поиск и обновление узлов. Для поиска определенных узлов в дереве загруженного в DOM-объект документа и вставки в найденные узлы нужного контента разработаны специальные варианты src-элементов – элементов-источников в составе dom-элементов. Данная функциональность задается атрибутами:

- method = "updateNode" – обновить узлы;
- method = "removeNode" – удалить узлы;
- targNode – задает целевой узел – содержит XPath-выражение для поиска в DOM-объекте узла, подлежащего модификации;
- updateValue – задает новое значение изменяемого узла – содержит ссылку на глобальную переменную или XPath-выражение для поиска в том же или другом DOM-объекте узла, содержащего требуемое значение.

Таким образом, новое значение для обновляемого узла должно быть предварительно подготовлено в глобальной переменной или DOM-объекте.

В качестве примера ниже представлен фрагмент HSM-кода, обеспечивающего в документе, загруженном в DOM-объект dom:Buff, поиск XML-элемента с текстовым значением «Шифр» и замену этого значения на значение, содержащееся в глобальной переменной shifr:

```
<dom:Buff>
    <src:inject-shifr method = "updateNode"
        targNode = "//*[text() = 'Шифр']"
        updateValue = "gbl::shifr"/>
</dom:Buff>
```

Аналогичным образом в следующем фрагменте отыскивается XML-элемент с текстовым значением «Препод» и его значение заменяется на значение атрибута teacher XML-элемента gruppa из DOM-объекта dom:Stud:

```
<dom:Buff>
    <src:inject-teacher method = "updateNode"
        targNode = "//*[text()='Препод']"
        updateValue =
            "dom::Stud#//gruppa/@teacher"/>
</dom:Buff>
```

ДОКУМЕНТЫ В ФОРМАТЕ OPEN XML

Для обработки документов в этом формате в HSM потребовалось предусмотреть дополнительную функциональность работы с zip-архивами. На лингвистическом уровне эта функциональность обеспечивается новыми атрибутами в составе doc-элементов:

- type = "zip" – специфицируемый файл является zip-архивом;
- action – действие с архивом: "create" – создать; "close" – закрыть;
- flags – задает опции действий с архивом;
- from – специфицирует файл, на основе которого создается архив;
- zipEntry – указывает файл в архиве, подлежащий обработке.

Представленный ниже фрагмент обеспечивает следующую последовательность действий: создание во временной директории TMP копии файла Blanks/pz.docx (шаблона-заготовки Word-документа), извлечение из него заархивированного файла word/header2.xml (спецификаций колонтитулов документа), загрузку его в DOM-объект dom:PZ-header, обработку (обозначено многоточием), возвращение обработанного файла на прежнее место в Word-документе, закрытие файла документа, отправка документа клиенту, удаление документа из временной директории:

```
<doc:PZ path = "TMP/pz.docx" action="create"
    from = "Blanks/pz.docx" type="zip"
    flags = "CREATE" />
<dom:PZ-header pass = "1" srcDoc = "PZ"
    zipEntry = "word/header2.xml">
    …
    <rcv:SaveHeader pass = "1" method = "save"
        doc = "PZ" zipEntry = "word/header2.xml"/>
</dom:PZ-header>
<doc:PZ action = "close"/>
<doc:PZ action = "send"/>
<doc:PZ action = "drop"/>
```

Таким образом, при переходе к документам в формате Office Open XML структура HSM-модели в целом остается прежней, расширяется набор атрибутов, которые нужно задать для спецификации обрабатываемого документа.

ЗАДАНИЕ ТОЧЕК ИНЪЕКЦИИ

Следующий важный вопрос связан с тем, что на этапе подготовки шаблона-заготовки необходимо как-то указать в нем те точки, в которые должен быть вставлен определенный контент на этапе создания персонализированного документа. Эта задача решается по-разному для различных типов документов с учетом особенностей организации их внутренней XML-разметки. На практике нами применялись два подхода: уникальные метки и закладки.

Уникальные метки. Этот подход заключается в том, что средствами «родного» редактора в точки инъекции в шаблоне-заготовке помещаются уникальные ключевые слова, которые заведомо не совпадают с другими словами в документе. Тогда в процессе обработки шаблона-заготовки может быть выполнен поиск в дереве документа соответствующих текстовых значений и замена их нужным контентом.

Для работоспособности данного подхода помимо требования уникальности меток необходимо, чтобы внутреннее представление текстовых значений меток в XML-разметке было стабильно и предсказуемо. Например, если мы при создании шаблона-заготовки в каком-то месте помещаем текст «99.99.9999» с тем, чтобы впоследствии заменить его текущей датой, то нужно быть уверенным в том, что в разметке шаблона будет присутствовать XML-элемент с этим текстовым значением. Тогда, например, XPath-выражение

```
//*[text () = '99.99.9999']
```

обеспечит нахождение нужного текстового узла в DOM-объекте.

Данное требование выполняется, например, в документах Visio, и метод уникальных меток хорошо работает в этом случае. Однако при попытке применить его для документов Word возникают затруднения. В XML-разметке Word исходный текст, вводимый пользователем при создании или редактировании документа, разбивается на фрагменты, размещаемые в разных XML-элементах. Таким образом, нет гарантий, что введенному тексту «99.99.9999» будет соответствовать в DOM-объекте текстовый узел в точности с этим значением. Это существенно затрудняет организацию поиска узлов, соответствующих уникальным меткам.

Работоспособность подхода, конечно, можно сохранить, если открывать XML-разметку шаблона в каком-нибудь XML-редакторе и вручную «склеивать» расчлененные метки (или написать программу, выполняющую эту процедуру), однако это заметно увеличит трудоемкость процесса подготовки и модификации шаблона.

Закладки. В редакторе Word предусмотрен механизм вставки именованных закладок для выделения позиции в тексте или участка текста, который можно использовать для указания то-

чек инъекции. В XML-разметке началу закладки соответствует XML-элемент w:bookmarkStart, имя закладки задает атрибут w:name этого элемента. Элемент начала закладки размещается на том же уровне иерархии, что и элементы w:r, внутри которых содержатся элементы w:t с текстовыми значениями. Поэтому для указания на текстовый узел, расположенный сразу после закладки с именем, скажем, «Дата», можно использовать следующее выражение XPath:

```
//w:r [preceding-sibling::w:bookmarkStart/
    @w:name = 'Дата'][1]/w:t
```

После заполнения шаблона узлы закладок следует удалить из персонализированного документа.

ЦИКЛИЧЕСКОЕ ЗАПОЛНЕНИЕ ШАБЛОНА

Рассмотренные выше простые примеры связаны с занесением одиночной порции контента на определенное место в шаблоне, подготовленное заранее. Помимо этого на практике достаточно часто возникает задача внесения множества единообразно оформленных однотипных порций контента. Речь идет о формировании множества элементов списка, строк таблицы и т. п. структур с повторяющимися элементами, причем количество элементов заранее (на этапе подготовки шаблона-заготовки) не известно. Необходимо на этапе создания шаблона подготовить заготовку для элемента повторяющейся структуры, а в процессе заполнения шаблона организовать циклическое заполнение заготовки контентом и занесение ее в шаблон.

Для решения этой задачи в HSM должны быть средства для организации многократного повторения деклараций поиска и обновления узлов в DOM-объектах в зависимости от содержимого других DOM-объектов. Такие средства были построены путем модификации for-элементов, применявшихся для циклической загрузки и обработки множества однотипных XML-документов из хранилища документов СОБД [10].

Элемент цикла. Модифицированный for-элемент, размещенный внутри HSM-состояния, обеспечивает многократное повторение (и, соответственно, обработку) своих вложенных элементов – тела цикла. Циклы обработки управляются следующими атрибутами:

- scan – задает совокупность узлов указанного DOM-объекта, для каждого из которых выполняется цикл обработки;
- scanNum – задает переменную, в которую записывается порядковый номер текущего цикла обработки;
- scanNode – задает переменную, в которую записывается обрабатываемый узел множества узлов, заданного атрибутом scan.

Следующий пример иллюстрирует многократную обработку элемента dom:Buff

```
<for:Each-Stud scan = "dom::Studs#//stud"
    scanNum = "glb::StudNum"
    scanNode ="StudNode">
    <dom:Buff pass = "1">
            …
    </dom:Buff>
</for:Each-Stud>
```

Здесь атрибут scan задает множество XML-элементов stud из DOM-объекта dom:Studs. Тело цикла – элемент dom:Buff – обрабатывается для каждого экземпляра stud, при этом текущий номер цикла хранится в глобальной переменной StudNum, а узел текущего экземпляра stud – в глобальной переменной StudNode.

Формирование порций контента. При построении разного рода списков, таблиц и т. п. необходимо сначала сформировать порцию контента (элемента списка, строки таблицы и т. д.) с соответствующей XML-разметкой, а затем внести ее в соответствующее место шаблона-заготовки. Для практического использования предложен и реализован следующий подход:

1) В шаблоне-заготовке создается структура с одним экземпляром повторяющегося элемента (например, таблица с одной строкой); экземпляр размечается, например, с помощью закладок.

2) На этапе заполнения узлы размеченного экземпляра извлекаются из загруженного в DOM-объект шаблона и переносятся в буферный DOM-объект.

3) Циклически обрабатывается буферный DOM-объект, при этом на каждом цикле:

а) экземпляр наполняется очередной порцией контента;

б) экземпляр копируется из буфера в DOM-объект шаблона на прежнее место.

Ниже приведен фрагмент HSM-кода, иллюстрирующего формирование в Word-документе нумерованного списка студентов. Предполагается, что файл document.xml, в котором требуется сформировать список, уже извлечен из zip-архива шаблона-заготовки и загружен в DOM-объект dom:document. В шаблоне подготовлена заготовка для нумерованного списка с одним элементом, помеченным закладкой «нумСписок». В заготовке элемента также имеются за-

кладки, указывающие точки занесения контента, в частности закладка «ФИО». Кроме того, предполагается, что в DOM-объект dom:Studs уже загружен XML-документ, содержащий сведения о студентах, на основании которых требуется сформировать список.

```
<dom:Buff pass = "1" nsPrefix = "w"
    nsUri = "http://schemas.openxmlformats.org/
    wordprocessingml/2006/main">
    <src:insert-ListElement join =
        "//w:p [w:bookmarkStart/@w:name =
        'нумСписок']" dom = "Target-document"/>
</dom:Buff>
<dom:Target-document pass = "1">
    <src:clear-ListElement method="removeNode"
        targNode = "//w:p [w:bookmarkStart/
          @w:name = 'нумСписок']"/>
</dom:Target-document>
<for:Each-Stud  pass="1" scan="dom::Studs#//stud"
    scanNum = "glb::StudNum"
    scanNode ="StudNode">
    <dom:Buff>
        <src:Insert-FIO method = "updateNode"
            targNode = "//w:r [preceding-sibling::
            w:bookmarkStart/@w:name = 'ФИО']
            [1]/w:t" updateValue = "dom::Studs#
            //stud [glb:: StudNum]/FIO"/>
        …
        <rcv:Return-Element method = "insertBefore"
            srcNode = "//w:p [w:bookmarkStart/
                @w:name = 'нумСписок']"
            targNode = "dom::Target-document#
                //w:p[w:bookmarkStart/
                @w:name = 'конецСписка']"/>
    </dom:Buff>
</for:Each-Stud>
```

ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ

Представленные результаты получили практическую апробацию при разработке веб-приложения, обслуживающего курсовое проектирование студентов по учебной дисциплине «Базы данных» на кафедре АСУ Уфимского государственного авиационного технического университета. Курсовое проектирование на различных этапах предусматривает разработку 8 видов графических (формат vdx) и 5 видов текстовых (формат docx) документов. Чтобы уменьшить нагрузку студентов в части рутинных операций при подготовке проектной документации, были разработаны шаблоны-заготовки документов различных типов и динамические модели HSM для их персонализации.

На различных этапах проектирования студенты через веб-интерфейс запрашивают те или иные заготовки документов и получают их с сервера в персонализированном виде с учетом индивидуальных сведений студента, хранящихся в базе в формате XML. Персонализация предусматривает, во-первых, заполнение стандартных реквизитов документов – в основных надписях, на титульных листах, колонтитулах и т. п., во-вторых, занесение сведений из индивидуального варианта задания (например, формируется техническое задание на проектирование).

Создаваемые на сервере персонализированные документы освобождают студентов от значительного объема рутинных действий по оформлению проектной документации и позволяют сосредоточиться на содержательной стороне проектирования. За семестр приложение обслуживает около 200 студентов по 3 направлениям обучения.

Применение технологии (заполнения шаблона как альтернативы генерации документа «с нуля») продемонстрировало относительную простоту создания и последующей модификации заготовки средствами «родного» редактора документов.

Недостатком следует признать относительную сложность составления громоздких XPath-запросов для поиска точек инъекции в размеченном шаблоне-заготовке: это требует знания «анатомии» внутреннего XML-представления документа соответствующего формата. В дальнейшем предполагается облегчить эту задачу путем введения в src-элементах специальных атрибутов, ориентированных на поиск точек инъекции для конкретных форматов документов (vdx, docx, vsdx и др.).

Другим направлением развития данной технологии представляется использование JSON-источников данных [16] для загрузки контента в шаблоны-заготовки наряду с XML.

ЗАКЛЮЧЕНИЕ

В статье рассмотрена технология создания персонализированных документов в веб-приложениях, в соответствии с которой предварительно создается размеченный шаблон-заготовка документа, который далее наполняется персональным контентом по запросам клиентов.

Для реализации технологии в ситуационно-ориентированных базах данных потребовалось дополнить функциональность элементов HSM-модели возможностью поиска и замены узлов в DOM-объектах документа, а также возможностями обработки содержимого zip-архивов (для персонализации документов в формате Open XML).

Для формирования структур с повторяющимися элементами предложен подход, при котором размеченный элемент переносится из шаблона в буферный DOM-объект, где циклически заполняется и копируется в результирующий документ.

Работоспособность технологии подтверждена путем практического применения при создании веб-приложения, обслуживающего студенческое учебное проектирование.

СПИСОК ЛИТЕРАТУРЫ

1. **Миронов В. В., Шакирова Г. Р., Яфаев В. Э.** Информационная технология персонализации электронных документов Microsoft Office в web-среде на основе XML // Вестник УГАТУ. 2008. Т. 10, № 2. С. 112–122. **[** V. V. Mironov, G. R. Shakirova, and V. E. Yafaev, “Information technology of personalization of Microsoft Office electronic documents in the web-based XML environment,” (in Russian), *Vestnik UGATU*, vol. 10, no. 2, pp. 112-122, 2008. **]**

2. **Миронов В. В., Шакирова Г. Р., Яфаев В. Э.** Иерархическая модель персонализованных документов и ее XML-реализация // Вестник УГАТУ. 2008. Т. 11, № 1. С. 164–174. **[** V. V. Mironov, G. R. Shakirova, and V. E. Yafaev, “Hierarchical model of personalized documents and XML-realization,” (in Russian), *Vestnik UGATU*, vol. 11, no. 1, pp. 164-174, 2008. **]**

3. **Миронов В. В., Шакирова Г. Р.** Технология персонализации документов формата Open Office XML на основе XSL-трансформации // Вестник УГАТУ. 2009. Т. 13, № 2. С. 188–197. . **[** V. V. Mironov and G. R. Shakirova, “Personalization technology for document format Open Office XML based on XSL-transformation,” (in Russian), *Vestnik UGATU*, vol. 13, no. 2, pp. 188-197, 2009. **]**

4. **XML-технологии** в электронных документах. Документы Word: учеб. пособие / В. В. Миронов, Н. И. Юсупова, Г. Р. Шакирова. Уфа: УГАТУ, 2009. **[** V. V. Mironov, N. I. Yusupova, and G. R. Shakirova, *XML-documents in electronic technology. Word documents*, (in Russian). Ufa: UGATU, 2009. **]**

5. **Миронов В. В., Юсупова Н. И., Шакирова Г. Р.** Ситуационно-ориентированные базы данных: концепция, архитектура, XML-реализация // Вестник УГАТУ. 2010. Т. 14, № 2 (37). С. 233–244. **[** V. V. Mironov, N. I. Yusupova, and G. R. Shakirova, “Situation-oriented databases: concept, architecture, XML realization,” (in Russian), *Vestnik UGATU*, vol. 14, no. 4 (39), pp. 200-209, 2010. **]**

6. **Миронов В. В., Маликова К. Э.** Интернет-приложения на основе встроенных динамических моделей: идея, концепция, безопасность // Вестник УГАТУ. 2009. Т. 13, № 2 (35). С. 167–179. **[** V. V. Mironov and K. E. Malikova, “Internet applications based on embedded dynamic models: idea, concept,” (in Russian), *Vestnik UGATU*, vol. 13, no. 2 (35), pp. 167-179, 2009. **]**

7. **Миронов В. В., Маликова К. Э.** Интернет-приложения на основе встроенных динамических моделей: архитектура, структура данных, интерпретация // Вестник УГАТУ. 2010. Т. 14, № 1 (36). С. 154–163. **[** V. V. Mironov and K. E. Malikova, “Internet applications based on embedded dynamic models: architecture, data structure, interpretation,” (in Russian), *Vestnik UGATU*, vol. 14, no. 1 (36), pp. 154-163, 2010. **]**

8. **Миронов В. В., Маликова К. Э.** Интернет-приложения на основе встроенных динамических моделей: элементы управления пользовательского интерфейса // Вестник УГАТУ. 2010. Т. 14, № 5 (40). С. 170–175. **[** V. V. Mironov and K. E. Malikova, “Internet applications based on embedded dynamic models: user interface controls,” (in Russian), *Vestnik UGATU*, vol. 14, no. 5 (40), pp. 170-175, 2010. **]**

9. **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. **[** E. S. Makarova and V. V. Mironov, “Web OLAP conceptual data model design on the basis of situation-oriented database,” (in Russian), *Vestnik UGATU*, vol. 16, no. 6 (51), pp. 177-188, 2012. **]**

10. **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. **[** E. S. Makarova and V. V. Mironov, “Analytical functions in web applications based on situation-oriented databases,” (in Russian), *Vestnik UGATU*, vol. 17, no. 5 (58), pp. 150-165, 2013. **]**

11. **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. **[** V. V. Kanashin and V. V. Mironov, “Hierarchical widgets: user interface organization in web applications based on situation-oriented databases,” (in Russian), *Vestnik UGATU*, vol. 17, no. 2 (55), pp. 138-149, 2013. **]**

12. **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. **[** V. V. Kanashin and V. V. Mironov, “Hierarchical widgets: input and control of user data in web applications on the basis of situation-oriented databases,” (in Russian), *Vestnik UGATU*, vol. 17, no. 5 (58), pp. 166-176, 2013. **]**

13. **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. **[** V. V. Kanashin and V. V. Mironov, “Hierarchical widgets: user data control algorithms in web applications on the basis of situation-oriented databases,” (in Russian), *Vestnik UGATU*, vol. 18, no. 1 (62), pp. 204-213, 2014. **]**

14. **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. **[** V. V. Mironov and A. S. Gusarenko, “Situation-oriented databases: concept of XML data management based of dynamic DOM objects,” (in Russian), *Vestnik UGATU*, vol. 16, no. 3 (48), pp. 159-172, 2012. **]**

15. **Гусаренко А. С., Миронов В. В.** Динамические dom-объекты в ситуационно-ориентированных базах данных: лингвистическое и алгоритмическое обеспечение источников данных // Вестник УГАТУ. 2012. Т. 16, № 6 (51). С. 167–176. **[** A. S. Gusarenko and V. V. Mironov, “Dynamic DOM objects in situation-oriented databases: lingware and knoware of data sources,” (in Russian), *Vestnik UGATU*, vol. 16, no. 6 (51), pp. 176-167, 2012. **]**

16. **Гусаренко А. С., Миронов В. В.** Smarty-объекты: вариант использования гетерогенных источников в ситуационно-ориентированных базах данных // Вестник УГАТУ. 2014. Т. 18, № 3 (64). С. 242–252. **[** A. S. Gusarenko and

V. V. Mironov, “Smarty-objects: use case of heterogeneous sources in situation-oriented databases,” (in Russian), *Vestnik UGATU*, vol. 18, no. 3 (64), pp. 242-252, 2014. **]**

ОБ АВТОРАХ

МИРОНОВ Валерий Викторович, проф. каф. автоматизированных систем управления. Дипл. радиофизик (Воронежск. гос. ун-т, 1975). Д-р техн. наук по упр. в техн. системах (УГАТУ, 1995). Иссл. в обл. иерархических моделей и ситуационного управления.

ГУСАРЕНКО Артем Сергеевич, асс. каф. автоматизированных систем управления. Канд. техн. наук (УГАТУ, 2013). Дипл. информатик-экономист (УГАТУ, 2010). Иссл. в обл. иерархических моделей, ситуационно-ориентированных баз данных, ситуационного управления и NoSQL.

ДИМЕТРИЕВ Роман Римович, м-рант каф. автоматизированных систем управления. Б-р техн. и технол. (УГАТУ, 2013). Готовит маг. дис. в обл. иерархических моделей ситуационно-ориентированных баз данных.

САРВАРОВ Марат Рашитович, м-рант каф. автоматизированных систем управления. Б-р техн. и технол. (УГАТУ, 2013). Готовит маг. дис. в обл. иерархических моделей ситуационно-ориентированных баз данных.

METADATA

Title: The personalized documents generating using DOM-objects in situation-oriented databases.

Authors:
A. S. Gusarenko[1], R. R. Dimetriev[2], V. V. Mironov[3], M. R. Sarvarov[4]

Affiliation:
Ufa State Aviation Technical University (UGATU), Russia.

Email:
[1] artyomgusarenko@gmail.com,
[2] dimetriev.roman@gmail.com,
[3] mironov@list.ru,
[4] shtirlizc@gmail.com.

Language: Russian.

Source: Vestnik UGATU (scientific journal of Ufa State Aviation Technical University), vol. 18, no. 4 (65), pp. 191-197, 2014. ISSN 2225-2789 (Online), ISSN 1992-6502 (Print).

Abstract: The technology of processing XML-documents in the situation-oriented databases (SOBD) to create personalized documents on the web server at the request of customers is discussed. Documents are generated based on predefined templates-the-blanks, are in a document repository SOBD; in the interpretation of the dynamic model (HSM) templates are loaded into the DOM-objects filled with personal content in accordance with the request and sends the client that issued the request. Specifications and new elements to define HSM operations personalization are discussed. The processing zip-archives to work with templates in Open XML is described. The use of technology to create documents in formats vdx and docx is illustrated. The practical application of the results in the learning process to create documentation in the course design is described.

Keywords: web-application; dynamic model; XML; DOM; PHP.

About authors:

MIRONOV, Valeriy Viktorovich, Prof., Dept. of Automated Systems. Dipl. Radiophysicist (Voronezh State Univ., 1975). Cand. of Tech. Sci. (USATU, 1978), Dr. of Tech. Sci. (USATU, 1995).

GUSARENKO, Artem Sergeevich, Cand. of Tech. Sci. (USATU, 2013), Dept. of Automated Systems., Grad. informatic-economist (USATU, 2010).

DIMETRIEV, Roman Rimovich, Dept. of Automated Systems. Dipl. Bachelor of Engineering and Technology (USATU, 2013).

SARVAROV, Marat Rashitovich, Dept. of Automated Systems. Dipl. Bachelor of Engineering and Technology (USATU, 2013).