УДК 004.91

В. В. МИРОНОВ, Г. Р. ШАКИРОВА

ИНТЕРПРЕТАЦИЯ XML-ДОКУМЕНТОВ СО ВСТРОЕННОЙ ДИНАМИЧЕСКОЙ МОДЕЛЬЮ

Обсуждается понятие динамического XML-документа, этапы работы с ним, его структура. Затрагивается вопрос поддержания целостности реквизитов динамического XML-документа. Предлагается использование объектно-ориентированного подхода к организации интерфейса пользователя для работы с динамическим XML-документом. *XML-документ*; *динамическая модель*; *интерпретация встроенной динамической модели*; *целостность*; *XML-технологии*; *объектно-ориентированный подход*

Вот уже несколько лет в мире продолжается бум XML. XML (Extensible Markup Language) [6] — это язык разметки, описывающий целый класс объектов данных, называемых XML-документами. Этот язык используется в качестве средства для описания грамматики других языков и контроля за правильностью составления документов. Язык XML привлек к себе уже достаточно много внимания со стороны разработчиков. Большинство новых версий известных приложений к числу своих достоинств причисляет именно поддержку XML.

Сегодня количество приверженцев этой технологии стремительно возрастает. Интерес к этой технологии обусловлен следующим: независимость (XML позволяет обмениваться данными системам, базирующимся на разных платформах); поддержка производителями; расширяемость (в процессе работы с XML-форматом в него можно добавлять любые новые теги); иерархичность (XML позволяет легко описывать сложные структуры данных с неограниченной вложенностью объектов); открытость (XML-документы существуют в форме обычного текста, который можно подготовить, передать, прочитать, исправить обычными текстовыми редакторами). Анализ этих преимуществ показывает эффективность и долгосрочную перспективность применения XML.

В [2] была разработана концепция динамических документов. Динамический документ — это документ, содержащий динамическую модель своего создания и применения. В [2] комплексы динамических документов реализуются с помощью средств Microsoft Office, в частности, используются макросы, описывающие правила организации этих документов. Кроме того, в работе [2] высказывается мысль о возможности реализации динамических документов на основе технологии XML.

В работе [1] авторами были рассмотрены концептуальные основы этого подхода и введено понятие динамического XML-документа (XML-DD) В данной работе проводится более детальная проработка вопросов, касающихся реализации динамических документов в среде XML.

1. КОНЦЕПЦИЯ ДИНАМИЧЕСКИХ XML-ДОКУМЕНТОВ

XML-DD содержит в себе три компонента:

- размеченный в XML-формате текст исходного документа;
- динамическая модель, задающая совокупность ситуаций (состояний), условий выполнения переходов между ними, инструкции по форматированию и ссылки на размеченный исходный документ;
- структура модели текущего состояния, отражающая текущее состояние модели документа.

По сути, применение XML-DD включает в себя два этапа: проектирование и непосредственное использование. В связи с этим в [1] были сформулированы принципы работы с XML-DD, среди которых можно выделить:

1) XML-DD — это корректный XML-документ.

2) В XML-DD должны быть представлены два типа разметки:

• прикладная, представленная тегами, которые должны структурировать документ независимо от текущего состояния;

• ситуационная, включающая теги, определяющие информацию, относящуюся к различным состояниям.

3) В структуре XML-DD должны быть выделены три составляющие (приведенные выше), каждая из них является корректным XML-документом.

4) Ситуационная разметка должна устанавливать соответствие между статическим компонентом XML-DD и возможными состояниями динамического документа.

5) Динамическая часть XML-DD и встроенная динамическая модель должны содержать только ситуационную разметку. С ее помощью в динамической части XML-DD организуются ссылки на фрагменты разметки статической части.

6) Динамическая модель должна допускать обработку внешним интерпретатором. XML-DD должен предоставлять интерфейс для доступа интерпретатора к навигации в динамической составляющей.

7) Должен быть разработан интерфейс для формирования представления динамического документа в контексте текущего состояния.

Основными объектами динамической модели являются: модель (*Model*), подмодель (*SubModel*), состояние (*State*), переход (*Arc*).

Корнем дерева должен являться элемент *Model*, дочерними для которого являются элементы типа *SubModel*. Элемент типа *SubModel* может содержать один или несколько дочерних элементов типа *State*. Элемент типа *State* включает в себя один или несколько элементов типа *Arc*, характеризующих либо переход из данного состояния в другое (переход типа *Jump*), либо переход между подмоделями (переход типа *Dive*). Элемент *Arc* содержит помимо некоторой служебной информации атрибуты *target*, задающие направление перехода.

Каждому элементу динамической модели ставится в соответствие определенный фрагмент размеченного текста исходного документа. Для этого в каждом служебном элементе используется один или более элементов типа *Ref*. Поскольку на один и тот же фрагмент документа может ссылаться несколько документов, то в модели также предусмотрены инструкции по форматированию. Они позволяют представить один и тот же текст по-разному в зависимости от того, с каким элементом он ассоциирован. Например, инструкция по форматированию может быть представлена элементами типа *key-words*, включающими ключевые слова, которые во фрагменте исходного текста будут особым образом форматироваться.

Динамическая составляющая XML-DD характеризует модель текущего состояния. Она формируется на основе структуры динамической модели и содержит данные того состояния, которое на момент наблюдения является активным.

Таким образом, XML-DD состоит из трех основных элементов: исходный документ, динамическая модель и модели текущего состояния (которая является частью динамической модели).

2. ЦЕЛОСТНОСТЬ РЕКВИЗИТОВ ДИНАМИЧЕСКОГО XML-ДОКУМЕНТА

Наряду со служебной информацией, задающей структуру динамического документа, вводятся пользовательские реквизиты, содержащие информацию, актуальную для пользователя в тот или иной момент жизненного цикла. Эти реквизиты имеют сложную структуру, неоднократно встречаются в документе и могут быть связаны между собой. В связи с этим необходим механизм для обеспечения целостности этих реквизитов. В [2] были выделены четыре группы ограничений целостности:

• структурности — в виде ограничений на форму структуры;

• обязательности — в виде условия наличия значений реквизитов;

• ссылочности — в виде наличия для одних реквизитов — других реквизитов с соответствующими значениями;

• баланса — в виде ограничений на совместные значения нескольких реквизитов.

Возникает вопрос, как решить эти задачи в среде XML, максимально используя предоставляемые этой технологией возможности. В связи с этим в работе [3] были выделены два класса ограничений целостности:

• стандартные, реализуемые с помощью схем XML-документа (декларативный подход);

• нестандартные, имеющие место в тех случаях, когда инструментария схем недостаточно и необходимы какие-то дополнительные механизмы, реализующие эти правила (недекларативный подход).

К стандартным ограничениям можно отнести правила обязательности, структурности и ссылочности. Эти ограничения реализуются с помощью инструментария схем[1] XML-документа. Причем ограничения структурности могут быть реализованы как на уровне совокупности реквизитов, так и для каждого отдельно взятого реквизита. Задание схемой стандартных ограничений в XML-DD возможно как на уровне элементов, так и на уровне атрибутов.

XML-схема строго фиксирует структуру документа. Нужно заранее знать, какие элементы, в каком порядке, в каком месте XML-DD будут присутствовать и каким образом они должны быть связаны между собой. При добавлении новых или изменения структуры или порядка следования уже заданных элементов необходимо перестраивать всю XML-схему. Для сложной многоуровневой структуры XML-DD изменение схемы — достаточно трудоемкий, а зачастую, невыполнимый процесс. Конечно, XML-схемы предоставляют набор средств для гибкого формирования XML-документа (например, элемент *all*, который задает группу элементов с произвольным порядком появления в валидном документе [3]). Однако использование встроенных возможностей схем имеет свои «подводные камни». Например, упомянутый выше элемент *all* хотя и допускает произвольный порядок следования элементов в документе, однако, ограничивает их появление только одним разом или полным отсутствием. А в XML-DD одни и те же реквизиты встречаются неоднократно и могут не иметь фиксированной позиции, т. е. требуют от механизма проверки большей гибкости.

Поэтому предлагается следующее решение. В схеме мы будем просто декларировать структуру возможных элементов-реквизитов, создадим их *прототипы*. Проверка валидности самих реквизитов на соответствие определенным в схеме прототипам будет осуществляться обработчиком ограничений целостности. Таким образом, осуществляется не полное, а выборочное сопоставление XML-схемы XML-документу. Для выборки элементов-реквизитов из XML-DD будем использовать инструментарий XML-технологий — язык запросов X*Path*[2].

Вторая группа ограничений целостности реквизитов — нестандартные — реализуется посредством дополнительных механизмов. К нестандартным относятся ограничения баланса, которые нельзя задать с помощью инструментария схем XML-документа. Здесь возможны два варианта реализации ограничений. Первый предполагает использование XML-инструментария, а именно, языка X*Query*[3]. X*Query* позволяет написать программный код, который и будет реализовывать нестандартные ограничения целостности реквизитов XML-DD. Нет необходимости применять запросы X*Query* ко всему XML-DD, поэтому с помощью XPath-выражений осуществляется фильтрация всех элементов-реквизитов динамической модели и проверка применяется только к ним.

Вторым вариантом приложения нестандартных ограничений целостности реквизитов является использование встроенных механизмов языка программирования, с помощью которого реализован интерпретатор. В частности, авторами используется язык C# платформы .NET.

Платформа .NET Framework использует XML Document Object Model (DOM), чтобы обеспечить доступ к данным в XML-документах и дополнительные классы для чтения, записи и навигации (XmlReader, XmlWriter и др.) в пределах XML-документа. Эти классы поддерживаются пространством имен System.XML. Используя эти механизмы, пишем программный код, реализующий правила ограничения целостности, которые нельзя задать с помощью XML-схем.

Таким образом, в структуру XML-DD, предложенную авторами в [1], добавляется новый элемент — база ограничений целостности (рис. 1). Он также должен быть корректным XML-документом, содержащим, во-первых, правило целостности, а во-вторых — XPath-выражение для указания фрагмента, к которому это правило должно быть применено.

Прототипы реквизитов XML-DD, реализующие декларативные ограничения целост-

[1] Схема XML-документа — это модель, отделенная от самого документа, в которой заданы его структурные и параметрические ограничения. Она определяет, что должно и чего не должно быть в документе.

[2] Язык X*Path* предназначен для указания определенных фрагментов дерева XML-документа. Концепция X*Path* — это концепция языка высокого уровня абстракции, предназначенного для адресации фрагментов XML-документа, подлежащих той или иной обработке в зависимости от среды применения [3]. В самых различных областях, где используются XML-документы, обнаруживается в той или иной степени присутствие X*Path* как языка навигации.

[3] X*Query* — функциональный язык, состоящий из нескольких видов выражений, которые могут использоваться в разных сочетаниях. X*Query* — это язык SQL для XML-данных.

ности, должны быть записаны в базе ограничений с использованием синтаксиса XML-схем. Вводится XML-элемент *decl_integr*, содержащий два элемента: *rule* (для записи выражения-правила) и *xpath* (для хранения значения XPath-выражения). Для недекларативных ограничений (выражений X*Query*-кода) зададим в базе ограничений элемент *nondecl_integ* с вложенными элементами *rule* и *xpath*, выполняющими те же функции, что и одноименные элементы XML-элемента *decl_integr*.

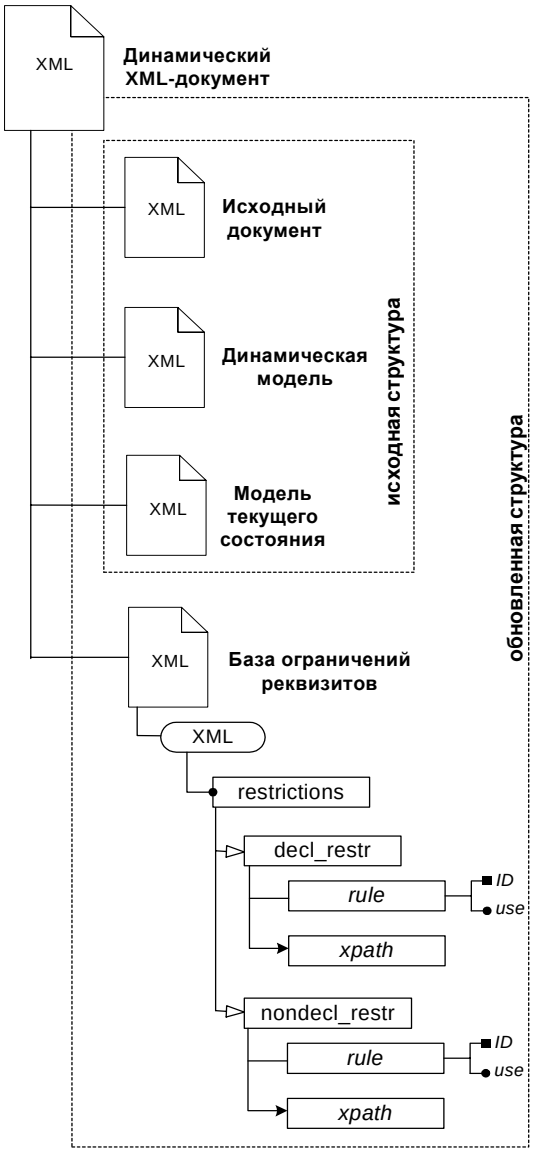

Рис. 1. Обновленная структура динамического XML-документа

Для элемента *rule* предусмотрены два атрибута: *ID* (уникальный в рамках базы ограничений идентификатор правила) и *use* (тип правила — обязательное или необязательное). Одно и то же правило может быть задействовано для разных реквизитов, поэтому допускается наличие нескольких элементов *xpath*.

3. КОНТРОЛЬ ОГРАНИЧЕНИЙ ЦЕЛОСТНОСТИ РЕКВИЗИТОВ

Проверка выполнения ограничений целостности реквизитов XML-DD должна осуществляться с помощью специального обработчика, отличного от интерпретатора XML-DD, на любом этапе работы с XML-DD. В качестве входной информации здесь будут использоваться динамическая модель, к которой привязаны реквизиты, и база ограничений целостности этих реквизитов. Эти XML-файлы должны быть переданы обработчику для последующей обработки.

Обходя последовательно все элементы XML-файла с базой ограничений, обработчик извлекает фрагменты динамической модели с помощью соответствующих XPath-выражений. К полученным XML-фрагментам применяются соответствующие описанные в базе ограничений правила.

В случае с декларативными ограничениями на каждой итерации логической проверки XML-DD обработчик ограничений должен извлекать определенный XML-фрагмент с некоторым реквизитом. Затем к выбранному фрагменту применяется фрагмент схемы, содержащий прототип соответствующего реквизита. Для нестандартных ограничений обработчик тоже должен просто сделать запрос на выборку отдельных фрагментов динамической модели и уже к ним применить X*Query*-код. Как и в случае со схемами, для выборки XML-фрагментов будем использовать XPath-выражения. После проверки реквизитов динамической модели управление передается интерпретатору XML-DD для дальнейшей работы с динамическим документом.

4. ИНТЕРПРЕТАЦИЯ ДИНАМИЧЕСКОГО XML-ДОКУМЕНТА

Обработка динамического XML-документа осуществляется с помощью XML-процессора (парсера) и интерпретатора XML-DD. Парсер считывает XML-документ, проверяет его синтаксис, сообщает об ошибках и обеспечивает программный доступ к содержимому документа. Интерпретатор осуществляет логическую обработку XML-DD и отвечает за формирование пользовательского интерфейса, в частности, навигации по документу. На рис. 2 приведена обобщенная схема обработки динамического XML-документа.

Как говорилось выше, каждый компонент XML-DD является XML-документом.

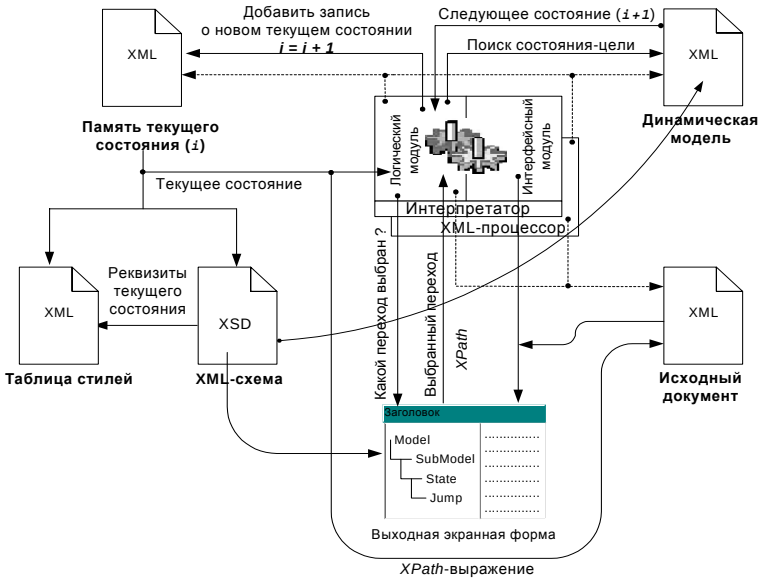

Рис. 2. Обобщенная схема обработки динамического XML-документа

На каждом этапе обработки XML-DD XML-процессор осуществляет проверку корректности всех этих компонентов.

По сути, интерпретатор должен быть представлен двумя модулями: модулем логической обработки и интерфейсным модулем, разделенными, по крайней мере, логически в рамках единого программного модуля.

На начальном этапе интерпретатор посредством интерфейсного модуля формирует некоторую экранную форму для того, чтобы позволить пользователю выбрать файлы-компоненты XML-DD. Затем управление передается логическому модулю интерпретатора, который осуществляет такие проверки XML-файлов, которые неосуществимы механизмами XML-процессора. В частности, в XML-DD ссылки элементов на исходный документ, а также значения атрибутов *target* элементов-переходов задаются с помощью описанных выше выражений XPath.

Логический модуль интерпретатора осуществляет проверку *достижимости* всех XPath-запросов в XML-DD. Также, например, может быть проведена проверка наличия по крайней мере двух основных компонентов XML-DD (динамической модели и исходного документа в XML-формате). Значение текущего состояния формируется логическим модулем интерпретатора уже в процессе работы с динамическим XML-документом.

После логической проверки управление вновь передается интерфейсному модулю интерпретатора. Он формирует выходную экранную форму для осуществления работы пользователя с XML-DD с учетом значения текущего состояния.

На основании XML-документа, содержащего данные о текущем состоянии, формируется экранная форма. Интерпретатор обращается к XML-документу с динамической моделью и в соответствии с тем или иным выбором пользователя и использованием механизма запросов XPath определяет, какое состояние будет текущим на следующей итерации. Результат передается в XML-файл памяти текущего состояния и начинается новая итерация. При этом значение предыдущего текущего состояния не удаляется, а «сдвигается вниз», таким образом обеспечивается «темпоральность»[1] динамического XML-документа.

При обработке текущего состояния логическим модулем интерпретатора в работу вступают также XML-схемы и XSL-таблицы стилей. С их помощью проводится соответствие между текущим состоянием и теми реквизитами, которые должны ему соответствовать. В зависимости от класса ограничений целостности реквизитов задействуют либо XML-схемы (для стандартных ограничений), либо механизмы логического модуля интерпретатора (для нестандартных ограничений).

5. ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС XML-DD

Можно сформулировать следующие требования к интерфейсу пользователя динамического XML-документа.

[1] По аналогии с *темпоральными* базами данных

1) Должны быть представлены три основные составляющие:

• механизмы навигации в динамическом XML-документе;

• информационная область (для отображения фрагментов исходного документа);

• инструментарий работы с пользовательскими реквизитами.

2) Механизм навигации должен отвечать требованиям историзма и позволять пользователю просматривать предысторию текущих состояний.

3) Средство навигации в динамическом XML-документе должно также позволять пользователю просмотреть возможные переходы из текущего состояния и всех последующих.

4) Должны быть предусмотрены два основных режима отображения навигации: только текущее состояние и вся динамическая модель. В последнем случае допустимо отклониться от соблюдения принципа темпоральности.

5) Экранная форма интерфейса XML-DD должна включать две основные функциональные области:

• навигационную (для отображения иерархии объектов динамической модели);

• информационную (для фрагментов исходного текста).

6) Должно быть предусмотрено различные стили форматирования одних и тех же фрагментов текста в информационной области, относящихся к разным элементам динамической модели.

В работе [5] был предложен объектно-ориентированный подход к организации иерархической модели.

Объектно-ориентированный подход основан на трех основных понятиях — объект, класс и экземпляр. Объект — это абстракция множества предметов реального мира, обладающих одинаковыми характеристиками и поведением. Объект представляет собой элемент такого множества. Экземпляр объекта — это конкретный элемент множества. Класс — это множество предметов реального мира, связанных общностью структуры и поведением. Элемент класса — это конкретный элемент данного множества. Таким образом, объект — это типичный представитель класса, а термины экземпляр объекта и элемент класса равнозначны.

Следующую группу важнейших понятий объектного подхода составляют инкапсуляция, наследование и полиморфизм. Объектный подход предполагает, что собственные ресурсы, которыми могут манипулировать только методы самого объекта, скрыты от внешних компонент. Сокрытие данных и методов в качестве собственных ресурсов объекта получило название инкапсуляции. Понятие полиморфизма может быть интерпретировано как способность объекта принадлежать более чем одному типу. Наследование означает построение новых классов на основе существующих с возможностью добавления или переопределения данных и методов.

Для реализации вышеописанных требований к формированию интерфейса предлагается использовать приведенные выше понятия и принципы объектно-ориентированного подхода.

За основу предлагается взять класс *TreeView* (дерево), используемый во многих объектно-ориентированных языках программирования высокого уровня. Этот класс предназначен для отображения иерархических структур. Поскольку динамическая модель является корректным XML-документом, то она представляет собой иерархию XML-элементов, соответствующих объектам модели. А динамическая модель является основой для построения механизма навигации по XML-DD. Следовательно, навигационный инструментарий по динамическому XML-документу также является древовидной структурой, потому вполне обоснован выбор класса *TreeView* в качестве базиса.

Предлагается создать три новых класса деревьев:

• *ModelTree* (для представления иерархии объектов-элементов XML динамической модели XML-DD);

• *CurrentTree* (для представления иерархии объектов-элементов XML динамической модели относительно данного текущего состояния);

• *HistoryTree* (для хранения записей о текущих объектах-элементов XML динамической модели, сохраняемых между итерациями).

Согласно одному из требований к проектированию интерфейса должны быть доступны два режима: текущее состояние и динамическая модель. Классы деревьев *CurrentTree* и *ModelTree* позволяют реализовать оба этих представления. Дерево класса *HistoryTree* будет скрыто от пользователя и должно нести только функциональную нагрузку для формирования истории выбора объектов модели в контексте текущего состояния.

Согласно принципу наследования, разработанные нами классы деревьев, являясь потомками класса *TreeView*, как приобретают свойства и методы этого класса, так и допускают введение некоторых новых.

Элементы структуры *TreeView* являются объектами класса *TreeNode*, а все эти объекты образуют коллекцию *Nodes*. Предлагается создать на основе класса *TreeNode* группу классов, соответствующих иерархии объектов динамической модели XML-DD:

- *ModelNode* (модель),
- *SubModelNode* (подмодель),
- *StateNode* (состояние),
- *JumpNode* (переход),

которые также будут входить в коллекцию *Nodes*. Иерархия этих классов представлена на рис. 3.

Таким образом, каждый элемент динамической модели рассматривается как объект, наделенный соответствующими свойствами и методами. Наряду со свойствами и методами класса *TreeNode*, каждый из его классов-наследников будет наделен и своими специфичными свойствами и методами. Каждый экземпляр класса-объекта динамической модели будет соответствовать XML-элементу файла, содержащего эту модель.

Так, каждый узел любого экземпляра классов *CurrentTree*, *ModelTree* и *HistoryTree* будет поставлен в соответствие некоторому XML-элементу. Поэтому представляется целесообразным введение свойства *XPath* для хранения XPath-выражения XML-элемента, соответствующего заданному узлу дерева, а также метода *GetXPath(XmlNode XNode)* с входными данными XML-узла для получения этого значения.

Для узлов, являющихся экземплярами класса *JumpNode*, вводится также свойство *TargetXPath* для хранения значения атрибута *target* XML-элемента типа *Jump*, содержащего путь к состоянию-цели перехода. Кроме того, вводятся свойства *predicate* и *action* для хранения данных о предикатах срабатывания перехода и выполняемых действиях.

Для узлов типа *SubModel* (представляющих не только саму подмодель, но и погружение в нее из некоторого состояния) вводится также свойство *TargetXPath* для хранения XPath-выражения самого первого состояния данной подмодели, для вычисления которого вводится метод *GetStateXPath(XmlNode StateNode)* по аналогии с методом *GetXPath(XmlNode XNode)* классов, образующих деревья *CurrentTree*, *ModelTree* и *HistoryTree*.

Для дерева класса *HistoryTree* предлагается простое добавление текстовых узлов, содержащих XPath-выражения последовательно выбираемых узлов. В результате на основании экземпляра можно для каждого текущего узла просмотреть предысторию его выбора.

Для экземпляра класса *CurrentTree*, отображающего текущее состояние модели, вводятся два основных дочерних узла — *Предыстория* и *Перспектива*.

Узел *Предыстория* использует информацию из дерева класса *HistoryTree* для формирования предыстории выбора данного текущего состояния (через какой переход / погружение, из какого состояния, подмодели, модели мы пришли в это состояние).

Узел *Перспектива* предназначен для того, чтобы предоставить пользователю механизм просмотра возможных путей из данного текущего состояния. При этом непосредственными потомками этого узла являются те переходы и погружения, которые относятся к данному состоянию. На следующем уровне появляются состояния и подмодели, к которым приводят переходы и погружения из предыдущего уровня иерархии. Пользователю для навигации (т. е. для выбора того или иного перехода для смены текущего состояния) доступны только элементы первого уровня узла *Перспектива*.

В дереве, относящемся к классу *ModelTree*, не предусмотрены узлы *Предыстория* и *Перспектива*, поскольку эти функции неявно заданы в самом дереве объектов динамической модели.

Выбор того или иного перехода или погружения модели приводит к перестройке всего дерева текущего состояния, добавлению нового узла в дерево класса *HistoryTree* и отображению в информационной области экранной формы фрагментов исходного текста, соответствующих новому текущему состоянию. При этом фрагменты текста, касающиеся ранее выбранных объектов динамической модели, не удаляются и доступны для просмотра пользователем.

6. ПРИМЕР РЕАЛИЗАЦИИ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

В [1] предлагается в качестве примера динамического XML-документа использовать «Федеральный закон о несостоятельности (банкротстве)». В нем устанавливаются основания для признания должника несостоятельным (банкротом), регулируются порядок и условия осуществления мер по предупре-

ждению несостоятельности (банкротства), а также порядок и условия проведения процедур банкротства [1].

«Закон о банкротстве» — яркий пример документа, ориентированного на ситуации. Его применение требует сопоставления реальной ситуации, сложившейся в организации, с соответствующими ситуациями, определенными в законе [1].

В [1] рассматривается ситуация, связанная с процедурой финансового оздоровления. С каждым объектом динамической модели ассоциируются соответствующие фрагменты закона. Динамическая модель процедуры банкротства на основе содержания закона может быть представлена в виде иерархической ситуационной модели, разработанной в [4].

На основании сформулированных в предыдущей главе принципов построения интерфейса работы пользователя с XML-DD, был разработан интерфейс навигации для данного примера. Для реализации интерфейса были использованы возможности языка C# платформы .NET Framework.

На рис. 4 показан фрагмент работы системы для текущего состояния «Утверждение» подмодели «План_оздоровления» состояния «Финансовое_оздоровление».

Как видно из рисунка, предусмотрены два пункта меню: *Файл* и *Режим*. Пункт меню *Файл* позволяет выбрать XML-файлы, содержащие динамическую модель и исходный текст. На основании этих файлов формируется дерево класса *CurrentTree*, где в качестве текущего состояния рассматривается первое состояние модели.

В правой, информационной, области окна отображаются те фрагменты исходного текста (в нашем случае — фрагменты федерального закона), которые соответствуют текущему состоянию.

Изменение текущего состояния перестраивает все дерево. При этом в узлах *Предыстория* и *Перспектива* отображается информация обо всех ранее выбранных объектах модели и тех объектах, которые доступны из данного состояния.

Пункт меню *Режим* предназначен для выбора одного из двух режимов отображения навигационного дерева: *Текущее состояние* и *Динамическая модель*. По умолчанию используется первый режим.

На рис. 5 представлено то же состояние, что и на рис. 4, но уже в режиме *Динамическая модель*.

Как видно из рисунка, содержимое информационной области меняется, а в навигационном дереве текущее состояние является выделенным. Предыстория и перспектива в полной модели задаются неявно и не полностью.

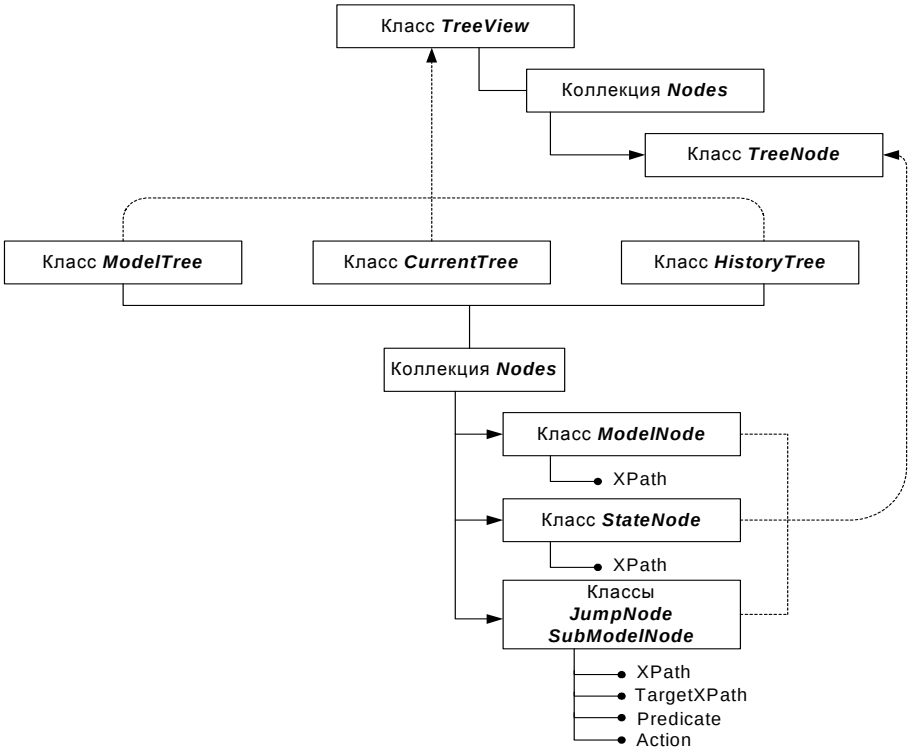

Рис. 3. Обобщенная схема иерархии классов интерфейса навигации пользователя по XML-DD

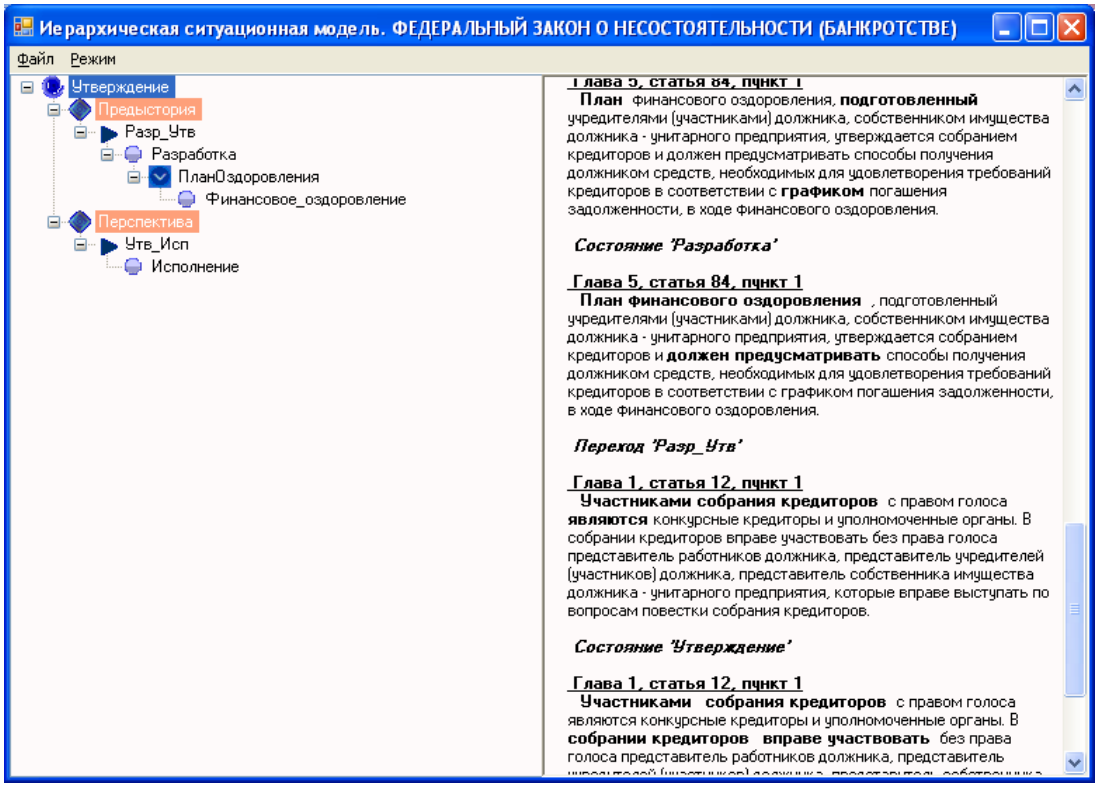

Рис. 4. Иллюстрация пользовательского интерфейса в режиме «Текущее состояние»

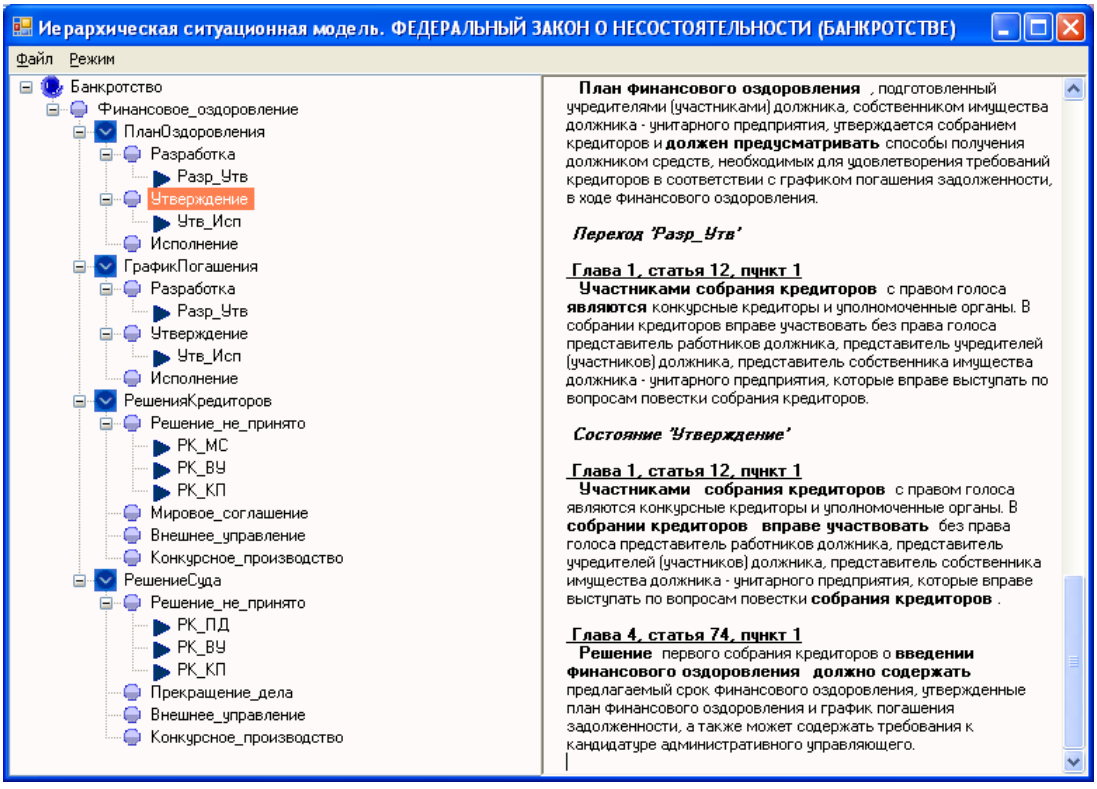

Рис. 5. Иллюстрация пользовательского интерфейса в режиме «Динамическая модель»

В информационной области фрагменты закона, соответствующие различным состояниям, форматируются по-разному. В нашем случае, ключевые слова, предусмотренные в соответствующих XML-элементах модели, выделяются жирным шрифтом

ВЫВОДЫ

1. В динамическом XML-документе представлены основные особенности концепции динамических документов и XML-технологий. В структуре динамического XML-документа выделяются размеченный исходный документ, динамическая модель (в т. ч. модель текущего состояния) и модель ограничений целостности реквизитов.

2. Использование взаимосвязанных реквизитов в рамках одного или комплекса динамических XML-документов делает актуальным вопрос о поддержании их целостности. На базе XML-технологий ограничения целостности могут быть реализованы на уровне совокупности реквизитов и для каждого отдельного реквизита. Ограничения целостности реквизитов в соответствии с механизмом их реализации делятся на две группы: декларативные и недекларативные.

3. Обработка динамического XML-документа осуществляется с помощью XML-процессора и интерпретатора XML-DD. Интерпретатор должен быть представлен двумя модулями: модулем логической обработки и интерфейсным модулем

4. В интерфейсе работы с XML-DD предусмотрены три класса деревьев *ModelTree*, *CurrentTree* и *HistoryTree* для всей модели, только текущего состояния и истории выбора объектов модели. Эти классы являются потомками класса *TreeView*. Три новых класса состоят из классов-наследников класса *TreeNode*, соответствующих XML-объектам динамической модели.

СПИСОК ЛИТЕРАТУРЫ

1. **Миронов, В. В.** Концепция динамических XML-документов / В. В. Миронов, Г. Р. Шакирова // Вестник УГАТУ. 2006. Т. 8, № 2(18). С. 58–63.
2. **Гарифуллин, Т. А.** Обеспечение целостности комплекса электронных документов на основе встраиваемых динамических моделей : автореф. дис. ... канд. техн. наук / Т. А. Гарифуллин. Уфа : УГАТУ, 2006.
3. **Миронов, В. В.** Контроль целостности в динамических XML-документах / В. В. Миронов, Г. Р. Шакирова // Вычислительная техника и новые информационные технологии. Уфа :УГАТУ, 2007, С. 178–184. (В печати).
4. **Миронов, В. В.** Информационная поддержка принятия решений при антикризисном управлении предприятием в условиях возможного банкротства / В. В. Миронов, Я. А. Олейник, Н. И. Юсупова // Вестник УГАТУ. 2005. № 2 (13). С. 112–120.
5. **Ахметшин, Р. Ф.** Инструментальные средства разработки систем поддержки принятия решений на основе асинхронной децентрализованной интерпретации иерархических ситуационных моделей : автореф. дис.... канд. техн. наук / Р. Ф. Ахметшин. Уфа : УГАТУ, 2004. 16 с.
6. **W3C.** Спецификация языка XML [Электронный ресурс]. http://www.w3.org/TR/REC-xml.

ОБ АВТОРАХ

Миронов Валерий Викторович, проф. каф. автоматиз. систем упр-я. Дипл. радиофизик (Воронежск. гос. ун-т, 1975). Д-р техн. наук по упр-ю в техн. сист. (УГАТУ, 1995). Иссл. в обл. моделей крит. ситуаций и ситуац. управления.

Шакирова Гульнара Равилевна, аспирант той же каф., асс. каф. вычислит. математики и кибернетики. Дипл. инженер по АСОиУ (УГАТУ, 2005). Готовит дис. в обл. использования встроенных моделей в XML-документах.