



*Перчёнок Олег Владимирович,
Поздняков Сергей Николаевич,
Посов Илья Александрович*

УДК 004.434

АВТОМАТИЗАЦИЯ ПРОВЕРКИ РЕШЕНИЯ ГЕОМЕТРИЧЕСКИХ ЗАДАЧ ПО ОПИСАНИЮ ИХ УСЛОВИЙ НА ПРЕДМЕТНО-ОРИЕНТИРОВАННОМ ЯЗЫКЕ

Аннотация

В статье предложен подход к разработке системы поддержки геометрических задач, в которой описание задачи строится на специально разработанном предметно-ориентированном языке, так что алгоритм решения задачи, подготовленный в какой-либо системе динамической геометрии, может быть верифицирован на формальном описании задачи. Приведено сравнение данного подхода с другими, рассмотрен формат хранения задач на геометрическом предметно-ориентированном языке. Описание проиллюстрировано примерами задач.

Ключевые слова: предметно-ориентированный язык, конструктивные геометрические задачи, система «1С: Образование 4.0 Школа», Java.

1. ВВЕДЕНИЕ

Существующие системы динамической геометрии, такие как «Живая математика» (The Geometer's Sketchpad), Cabri, Cinderella, GeoGebra [1], 1С Математический Конструктор (1С МК) и др. основаны на выполнении ряда геометрических построений, преобразований и измерений. Все они, кроме последней (1С МК), не имеют инструментов для проверки решений (в Cabri есть возможность проверять выполнение простейших отношений, например параллельность линейных объектов). В 1С МК автор задачи может построить предварительно её решение (задать

линейный алгоритм построением динамического чертежа). После этого ученик может проверить свое решение, сравнив его динамически с эталонным (то есть, решение будет признано правильным даже при ином алгоритме решения).

В то же время за рамками рассмотрения остается проблема адекватного описания задачи. Получается, что описанием задачи служит текст, никак не привязанный к задаче инструментально. Поэтому достаточно сделать ошибку в формулировке, как правильное решение учеником измененной задачи будет оценено как ошибочное, так как фактически сравнивалось с решением другой задачи.

В статье предложен иной подход к разработке системы поддержки геометрических

© Перчёнок О.В., Поздняков С.Н.,
Посов И.А., 2012

задач, когда описание задачи строится на специально разработанном предметно-ориентированном языке, так что алгоритм решения задачи, подготовленный в какой-либо системе динамической геометрии, может быть верифицирован на формальном описании задачи.

Предметно-ориентированный подход к составлению задач по комбинаторике был описан в [2]. Отличием представляемой системы является иной язык описания задач (в работе [2] использовалось специально разработанное xml-описание задач) и иной способ верификации решений.

2. СТРУКТУРА СИСТЕМЫ

Система включает модули учителя и ученика, реализованные принципиально на разных концепциях. В модуле учителя автор задачи описывает условие в виде текста и в виде предикатов и логических связей между ними в привычной для себя форме, строит исходный чертеж и задает набор инструментов, которым может пользоваться ученик при решении задачи. В модуле ученика школьник может прочитать текстовое условие, выполнить необходимые построения и

отослать решение на контроль, который состоит в проверке заданных учителем предикатов.

Поддержка динамической геометрии необходима для построения чертежей к задачам. Реализация собственной системы динамической геометрии трудоемка, но существует немалое количество уже реализованных систем. Необходимо было исследовать существующие системы и выбрать из них ту, которая реализована на платформе Java, имеет открытый исходный код, имеет все необходимые возможности и хороший пользовательский интерфейс. Из свободных систем динамической геометрии, реализованных на Java, наиболее подходящими для решения поставленных задач оказались GeoGebra и CaRMetal. Предпочтение было отдано первому варианту, поскольку интерфейс CaRMetal имеет слишком много элементов управления и разнообразных панелей с настройками, которые, хоть и предоставляют большие возможности, но при этом пугают пользователей. Во-вторых, при попытке программистами выделить редактор из CaRMetal, чтобы воспользоваться им в разрабатываемой системе, оказалось, что это выделение связано с определенными слож-

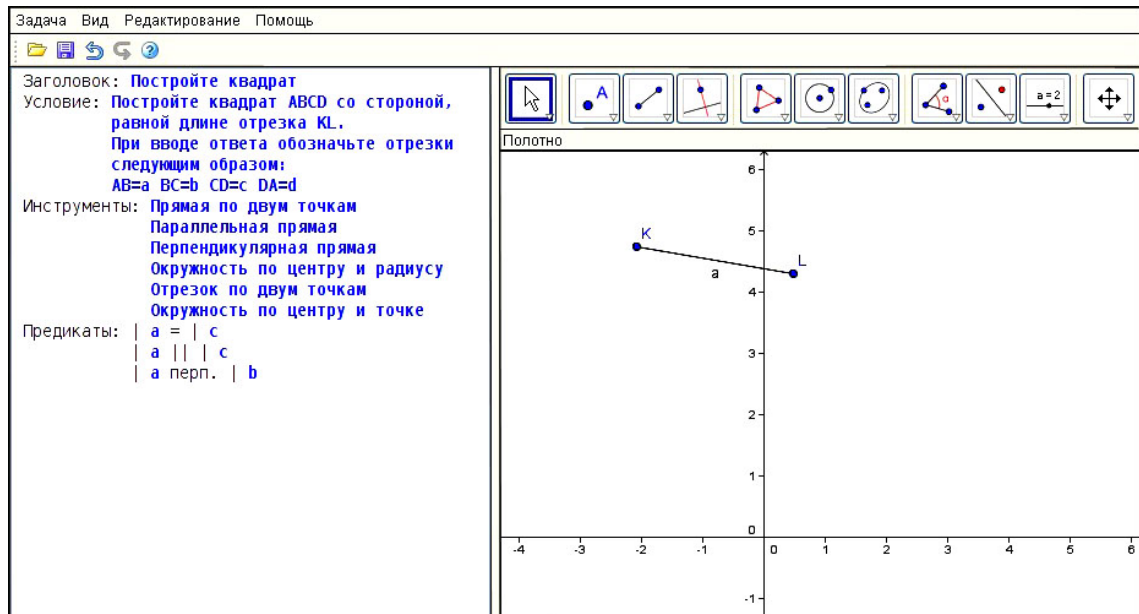


Рис. 1. Задача в модуле учителя. Заголовок и условие вводятся с клавиатуры, набор инструментов и предикаты формируются посредством выпадающих меню

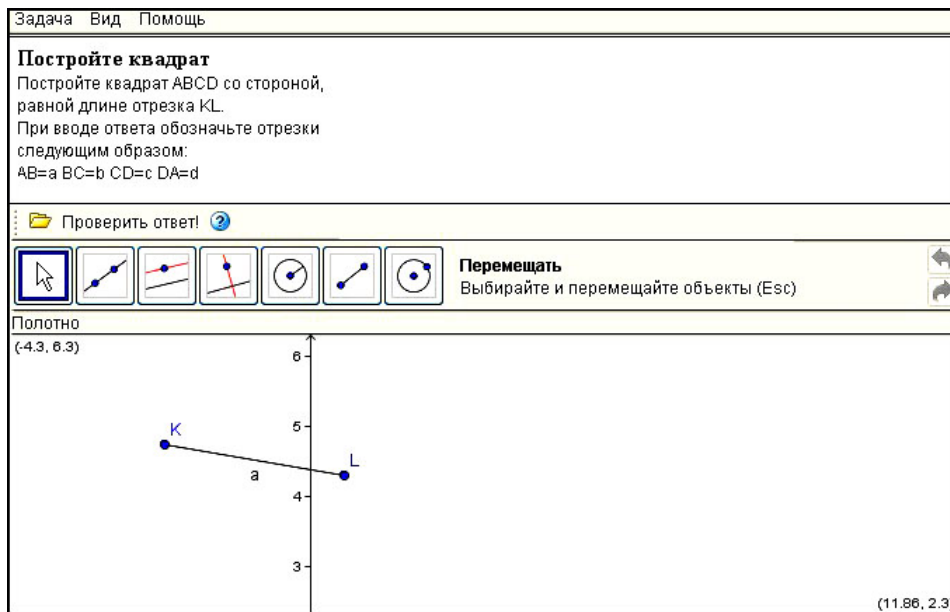


Рис. 2. Задача в среде ученика.

Описание инструментов превращается в набор допустимых инструментов

ностями, которых не обнаружилось при выделении редактора из GeoGebra. Программа GeoGebra написана Маркусом Хохенвартером на языке Java (работает на большом числе операционных систем). Переведена на 39 языков. [1]

3. ПРИМЕР ЗАДАЧИ

Для иллюстрации структуры системы приведем пример задачи: «Постройте квадрат ABCD со стороной, равной отрезку KL, заданному на чертеже». Данная задача в среде учителя и ученика приведена соответственно на рис. 1 и 2.

- ➔ Прямая по двум точкам
- ➔ Параллельная прямая
- ➔ Точка
- ➔ Перпендикулярная прямая
- ➔ Окружность по центру и радиусу
- ➔ Середина отрезка
- ➔ Отрезок по двум точкам
- ➔ Угол заданной величины
- ➔ Окружность по центру и точке
- ➔ Угол по трем точкам

Рис. 3. Список доступных инструментов

Модуль учителя состоит из предметно-ориентированного редактора (см. рис. 1, слева), в котором автор вводит условие, и области построения исходного чертежа (см. рис. 1, справа).

Модуль ученика состоит из текста задачи (см. рис. 2, сверху) и области построения решения задачи (см. рис. 2, снизу).

4. ПОДДЕРЖИВАЕМЫЕ ПРИМИТИВЫ

В системе разработаны инструменты для включения в интерфейс ученика: построения точек, отрезков и углов, параллельных и перпендикулярных прямых, окружности по центру и точке, центру и радиусу, построения середины отрезка. Раскрытый список из редактора приведен на рис. 3.

В формальном условии задачи можно описывать отношения между базовыми геометрическими объектами, такими как точка, окружность, отрезок, луч, прямая. Возможны ссылки как на объекты исходного чертежа, так и на те объекты, которые необходимо построить ученику (рис. 4).

В системе реализованы следующие отношения между объектами: принадлежность точки прямой, окружности, параллельность

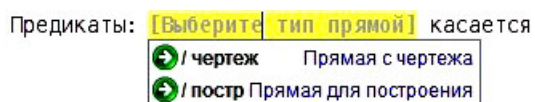


Рис. 4. Типы расположения объекта «прямая»

и перпендикулярность линейных объектов, принадлежность точки середине отрезка, равенство отрезков и углов, касание прямой и окружности (рис. 5).

5. ФОРМАТ ХРАНЕНИЯ ПРЕДМЕТНО-ОРИЕНТИРОВАННОГО ОПИСАНИЯ

Условия задач хранятся в двух файлах. В одном файле хранится формальное условие из предметно-ориентированного редактора в формате XML, а в другом – исходный чертеж в формате GGB (формат системы GeoGebra).

Описание задачи находится внутри тега «task», который имеет атрибут «name» – имя задачи. Оно состоит из списков инструментов, предикатов и текстового условия задачи.

Инструменты перечисляются внутри тега «instruments». Каждый инструмент представляет собой тег «instrument» с атрибутом «name» – название инструмента.

Условия для проверки (предикаты) содержатся внутри тега «predicates». Каждый предикат представляет собой тег «predicate»

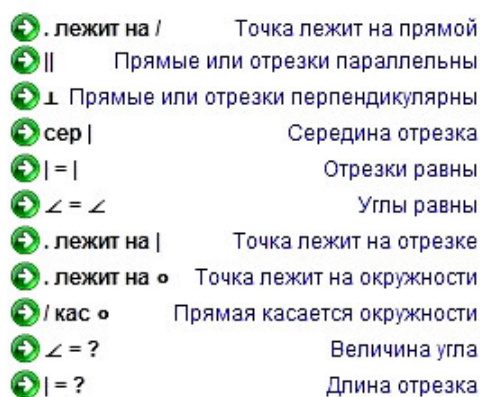


Рис. 5. Доступные отношения между объектами

с атрибутом «name» – название предиката. Внутри тега указываются параметры предиката – геометрические объекты.

Геометрический объект (элемент) описывается тегом «geoElement» с атрибутами «type» – тип объекта (прямая, окружность и т. д.), «locType» – тип расположения объекта (на исходном чертеже или для построения учеником).

Пример XML-файла с задачей приведен в листинге 1.

Благодаря однотипности описания элементов данный язык легко расширяем.

6. РЕДАКТОР ПРЕДМЕТНО-ОРИЕНТИРОВАННЫХ ЯЗЫКОВ

Редактор предметно-ориентированного языка может быть текстовым, графическим или являться набором элементов управления. В первом случае программа вводится в редактор в виде текста, как это происходит во всех языках программирования общего назначения, во втором – в виде изображения, например в виде блок-схемы или геометрического чертежа, что используется в системе. В третьем случае для ввода «программы» пользователь пользуется элементами управления, такими как кнопки, списки, ползунки и т. п.

Последний способ хорош для небольших языков, где количество элементов управления не превышает критического предела, то есть они помещаются на экране, и пользователь способен быстро их воспринять. К тому же в последнем случае любое изменение языка требует изменения редактора, и это не всегда удобно при реализации.

При выборе редактора для предметно-ориентированных языков за основу была взята система MPS (Meta Programming System) от компании JetBrains [3]. Она позволяет создавать предметно-ориентированные языки и редакторы для них. К сожалению, эта система слишком тяжеловесна, и в ней пока нет возможности экспортировать созданный редактор в другой программный продукт. По утверждению разработчиков, это произойдет, но не скоро. Вместо того

Листинг 1. Пример XML-файла с геометрической задачей

```

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<task title="Параллельные прямые">
  <tools>
    <tool name="LINE_PERPEND" />
  </tools>
  <predicates>
    <predicate name="Parall">
      <geoElem locType="given" name="a" type="Line" />
      <geoElem locType="new" name="b" type="Line" />
    </predicate>
    <predicate name="LaysOn">
      <geoElem locType="given" name="D" type="Point" />
      <geoElem locType="new" name="b" type="Line" />
    </predicate>
  </predicates>
  <description>
    <![CDATA[
      Даны прямая a и точка D. Построить прямую b, параллельную a.
      При этом точка D принадлежит прямой a. Использовать только
      инструмент "прямая, перпендикулярная данной".
    ]]>
  </description>
</task>

```

чтобы ждать появления этой функциональности в MPS, было решено повторить ее и самостоятельно реализовать MPS-образные редакторы для предметно-ориентированных языков, используемых в системе. Такой редактор похож на текстовый редактор, но при этом он постоянно помогает и подсказывает пользователю, что именно и в каком месте текста он ожидает увидеть. Возможности ввести синтаксически неверный текст не существует, каждая конструкция может появиться только в том месте, где редактор попросил ее ввести. В частности, это избавляет программистов от необходимости синтаксического анализа текстов, то есть вводимый пользователем текст сразу появляется в виде дерева синтаксического разбора.

Задача в структурированном редакторе состоит из клеток. Существуют нередатируемые клетки (подсказка пользователю) и редактируемые клетки (содержательная часть задачи).

Активная клетка – это клетка, с которой пользователь может производить действия в данный момент. Клетка является активной,

если подсвечена желтым цветом. В нижней части редактора расположен блок возможных действий с клеткой. В нем расположены названия действий и горячие клавиши для их выполнения. Их также можно выполнять, нажимая левой кнопкой мыши на названия действий.

Существуют несколько типов редактируемых клеток.

1. Простые клетки, в которые вводится число, строка, логическое значение (да/нет) или выбирается значение из списка. Например: поле ввода заголовка задачи, поле выбора инструмента (конкретного элемента списка инструментов). По умолчанию такие клетки пусты. Возможные операции: очистить содержимое, выбрать из списка (при его наличии).

2. Клетки-структуры, которые состоят из последовательности клеток. По умолчанию эти клетки пусты (например, конкретный элемент списка предикатов). Раскрывающийся список (подобный списку простой клетки с выбором значений) позволяет выбрать тип структуры. Возможные операции:

очистить содержимое, выбрать тип содержимого.

3. Клетки-списки, которые состоят из набора клеток. По умолчанию такой набор может быть пустым (например, значение «не ограничивать участника» в списке инструментов) или состоять из одного пустого элемента (например, список предикатов). Список предикатов на рис. 1 состоит из 3-х элементов: предикат параллельности линий, предикат перпендикулярности линий и предикат равенства отрезков. Возможные операции: добавить элемент, удалить элемент.

7. ПОРЯДОК РАБОТЫ С ПРОГРАММОЙ

Порядок работы рассмотрен на примере следующей задачи:

Дана прямая a и точка D . Построить прямую b параллельную a , такую, что точка B лежит на прямой b , используя только инструмент построения прямой, перпендикулярной данной.

Условия задачи преподаватель создал в модуле учителя. Он ввел условие в структурированном редакторе (слева) и построил исходный чертеж (справа). Выбор инструментов производился из списка, изображен-

ного на рис. 3. Аналогично инструментам был осуществлен выбор предикатов (рис. 5).

Далее учитель заполнил параметры предиката и добавил еще один предикат. Окончательный вариант изображен на рис. 6.

Ученик при этом решил задачу следующим образом. Он построил прямую, перпендикулярную прямой a , проходящую через точку D , далее прямую, перпендикулярную построенной и проходящую через точку D . И наконец, он нажал на кнопку «проверить ответ» (рис. 7).

8. ЗАКЛЮЧЕНИЕ

Система написана на языке Java, что обеспечивает ее кросс-платформенность. Разработка осуществлялась при финансовой поддержке Фонда содействия развитию малых форм предприятий в научно-технической сфере.

Следующий шаг в разработке программы – ее внедрение в систему «1С: Образование 4.0. Школа» [4]. Такое внедрение станет возможным после разработки версии в виде Java-апплета. Система «1С: Образование 4.0. Школа» работает с цифровыми образовательными ресурсами (ЦОР). Планируется

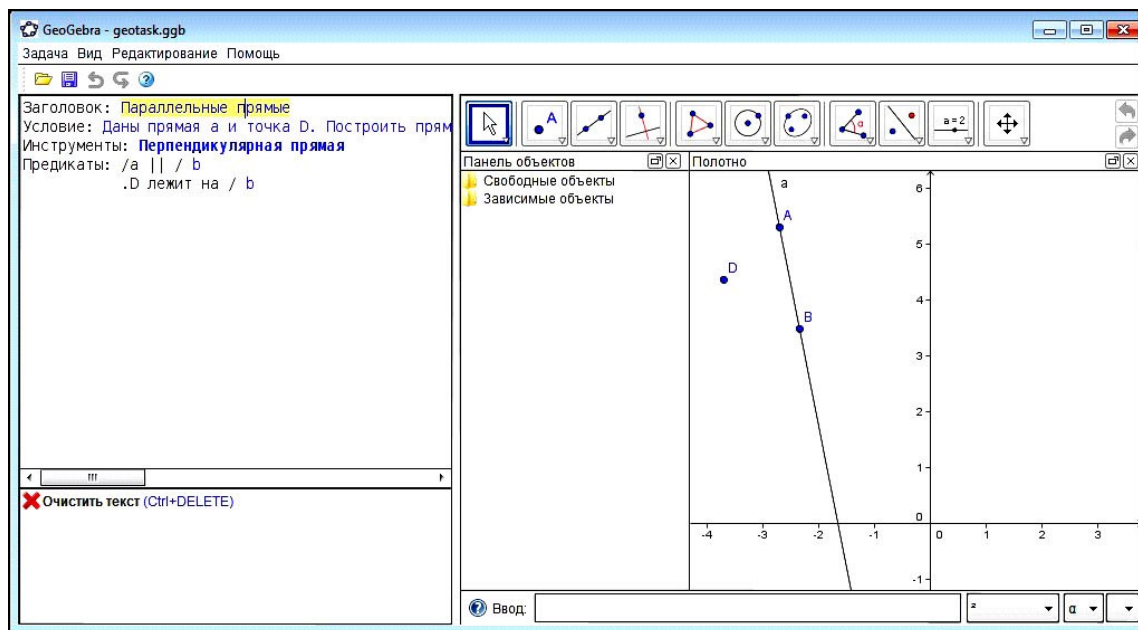


Рис. 6. Окончательный вид введенной задачи

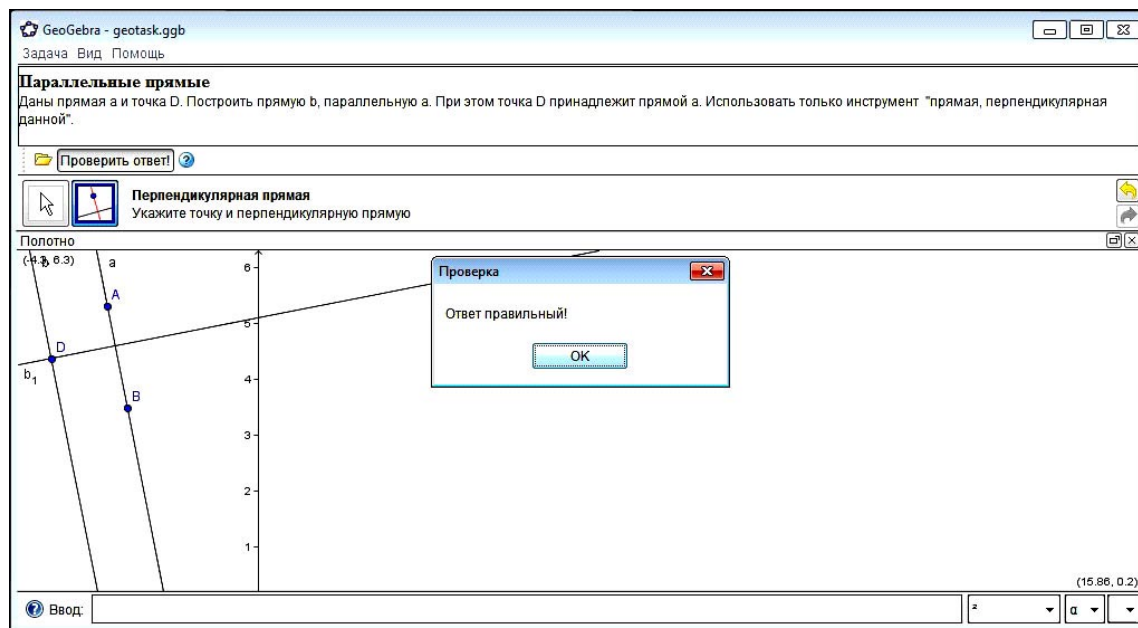


Рис. 7. Решение задачи в модуле ученика

архивировать каждую задачу в такой ресурс. Формат ресурса представляет собой архив, содержащий классифицирующую информацию, стартовый файл (в формате HTML) и вспомогательные файлы. В стартовом файле разрешено использовать апплеты. При вызове апплета в качестве параметра указывается имя XML-файла задачи. Взаимодействие между апплетом и системой «1С: Образование 4.0. Школа» происходит на языке JavaScript по стандарту SCORM [5]. Данное

внедрение позволит широко использовать программу в учебном процессе.

В настоящее время на Факультете компьютерных технологий СПбГЭТУ «ЛЭТИ» происходит разработка аналогичной системы в области цифровой обработки сигналов (на основе взаимодействия с инженерно-математическим пакетом MATLAB [6]). Планируется использование результатов разработки в учебном процессе для проверки знаний по курсу «Цифровая обработка сигналов».

Литература

1. Сайт системы динамической геометрии GeoGebra // <http://www.geogebra.org> (дата обращения 18.02.2012).
2. Богданов М.С. Автоматизация проверки решения задач по формальному описанию ее условия // Компьютерные инструменты в образовании, 2006. № 4. С. 51–57.
3. Сайт программы MPS // <http://www.jetbrains.com/mps/> (дата обращения 18.02.2012).
4. Сайт системы программ «1С:Образование 4.0. Школа» // <http://school-collection.edu.ru/catalog/rubr/3b9b31b6-31ee-4616-aaf1-10fc66f9879/102490/> (дата обращения 18.02.2012).
5. Статья о стандарте SCORM. Википедия – свободная энциклопедия // <http://ru.wikipedia.org/wiki/SCORM> (дата обращения 18.02.2012).
6. Потемкин В.Г. MATLAB 6: среда проектирования инженерных приложений. М.: Диалог-МИФИ, 2003.

Abstract

The article presents an approach to the development of the system that supports geometric problems with the formal verification of their solution. This is achieved by the description of problems' statements with the specially developed domain-specific language that specifies a set of predicates that should hold for the construction made by a student. The system has two modules: teacher's module and student's module. Several examples of problems are presented.

Keywords: domain-specified language, constructive geometric tasks, GeoGebra, Java.

*Перченков Олег Владимирович,
аспирант СПбГЭТУ «ЛЭТИ»,
факультет Компьютерных
технологий и информатики (ФКТИ),
кафедра ВМ-2,
olegperch@gmail.com,*

*Поздняков Сергей Николаевич,
профессор кафедры ВМ-2
СПбГЭТУ «ЛЭТИ»,
rozdnkov@gmail.com,*

*Посов Илья Александрович,
старший преподаватель кафедры
информационных систем в искусстве
и гуманитарных науках СПбГУ,
iposov@gmail.com*



Наши авторы, 2012.

Our authors, 2012.