АНО «Институт логики, когнитологии и развития личности»
ALT Linux
НОУ «ИПС-Университет г. Переславля им. А. К. Айламазяна»
Институт Программных Систем РАН

Восьмая конференция «Свободное программное обеспечение в высшей школе»

Переславль, 26–27 января 2013 года

Тезисы докладов

Москва,
Альт Линукс,
2013

Восьмая конференция «Свободное программное обеспечение в высшей школе»: Тезисы докладов / Переславль, 26–27 января 2013 года. М.: Альт Линукс, 2013. — XX с. : ил.

В книге собраны тезисы докладов, одобренных Программным комитетом восьмой конференции «Свободное программное обеспечение в высшей школе».

ISBN 978-5-905167-13-3

Программа конференции

26 января

11.00-12.00 Регистрация участников и заселение

12.00 А. Е. Новодворский. Открытие. Информация оргкомитета

12.10–12.40 С. М. Абрамов, член кор. РАН.

12.40–13.10 В. А. Сухомлин, д.т.н, профессор МГУ

13.10–13.40 Н. Н. Непейвода, проф.

13.40–15.00 Перерыв на обед

15.00–15.15 В. А. Старых, А. Ю. Кузнецов, А. А. Карасёв и др.

15.15–15.30 В. А. Старых, М. А. Ушанов, В. И. Одинцов и др.

15.30–16.00 А. А. Якушин

16.00–16.20 М. А. Ройтберг

16.20–16.40 А. А. Харченко, Е. А. Роганов

16.40–17.00 Кофе-пауза

Вечернее заседание
17.00–19.30

17.00–17.20 Г. В. Курячий

17.20–17.40 В. Л. Симонов, С. А. Мартишин, М. В. Храпченко

17.40–18.00 Л. Н. Чернышов, В. Н. Лукин

18.00–18.20 А. Г. Михеев

18.20–18.40 А. А. Рябуша, С. А. Чунин

18.40–19.00 И. А. Хахаев

19.00–19.20 А. Е. Прудников, Е. Ф. Гребцова

19.20–19.30 А. А. Капнинский

27 января

Утреннее заседание
9.30–13.35

9.30–9.45 А. Н. Пустыгин, М. В. Зубов, Е. В. Старцев

Выделение типов в универсальном классовом представлении для статического анализа исходного кода 53

9.45–10.00 А. Н. Пустыгин, А. А. Ковалевский и др.

Прототип инструмента для получения и анализа графа потока управления открытых исходных текстов C/C++ на основе универсального промежуточного представления 58

10.00–10.15 Н. В. Ключко

Проблемы использования свободного программного обеспечения на практических занятиях в высшем учебном заведении 60

10.15 В. П. Иванников, академик РАН, директор ИСП РАН, зав. кафедрой Системного программирования ВМиК МГУ, зав. базовой кафедрой Системного программирования МФТИ.
«Использование СПО в образовании»

11.15–11.35 Кофе-пауза

11.35–13.35 Круглый стол: Задачи подготовки и переподготовки инженерных кадров в условиях расширения использования СПО в различных отраслях промышленности.

13.35–14.50 Перерыв на обед

Дневное заседание
14.50–18.30

14.50–15.10 В. В. Яковлев, Д. В. Хачко, А. Г. Кушниренко и др.

Кумир 2.0: компилятор и среда выполнения 64

15.10–15.30 И. В. Воронин, В. В. Воронина

Среда КУМИР для изучения алгоритмов управления сенсорными сетями роботов 67

Вне программы

Сергей Абрамов, член кор. РАН
Переславль-Залесский, Университет г. Переславля им. Алаймазяна

Сегодняшние проблемы высшего образования в России (по всем областям знаний и особенно в области ИКТ)

1. Как изучать стремительно развивающиеся IT-технологии? ==> только в кооперации образование+наука+бизнес... В реальных проектах.
2. Проблема 1: Пагубная мобильность ==> ведется целенаправленная и эффективная политика вымывания интеллекта из регионов
3. Проблема 2: Россия останется без специалистов ==> в главнейших отраслях (например, ИКТ) разрешено готовить только бакалавров (а это грамотные пользователи чужих решений) и магистров (а это ученые) и запрещено готовить специалистов (а это как раз разработчики отечественных решений).

   Целенаправленная и эффективная политика закрепления (на уровне образования) роли России, как потребителя чужих ИКТ-разработок.
4. Проблема 3: Наука в Университетах? Нет! Образование в Российской академии наук!

   Целенаправленная и эффективная политика разрушение Россиийкой академии наук и попытка (бесперспективная) проведения науки в Вузах... Хотя (см. пункт (1)) надо делать все в кооперации.

Владимир Сухомлин
Москва, МГУ имени М. В. Ломоносова

Виртуальный ИТ-университет для переподготовки кадров

Аннотация

На базе развития концепции Виртуального ИТ-университета разработаны удлиненные программы дополнительного образования (переквалификации) магистерского уровня, реализуемые в режиме дистанционного обучения и позволяющие получить дополнительную квалификацию разработчика компьютерных технологий к диплому о высшем образовании.

Введение

Предлагаемый проект представляет собой одно из воплощений развиваемой на факультете ВМК МГУ концепции Виртуального ИТ-университета [1] в виде системы дистанционной подготовки/переподготовки ИТ-кадров высокой квалификации на основе удлиненных программ дополнительного образования. Такая система, имеющая предварительное название «ИТ-Мастер» (ИТ-М), ориентирована на предоставление услуг в сфере дополнительного образования с акцентом на реализацию комплексных учебных программ магистерского уровня, позволяющих получить выпускникам соответствующую подготовку и диплом о дополнительной квалификации к высшему образованию. Все программы в системе ИТ-М реализуются в режиме дистанционного обучения. Они разработаны с учетом требований международных стандартов куррикулумов [2,3], многолетнего опыта реализации программ на дополнительную квалификацию «Разработчик компьютерных технологий», а также с учетом текущих запросов отечественного бизнеса. Учебные программы в системе ИТ-М характеризуются также гибкостью в диверсификации. В первую очередь они предназначены для тех, кто, получив непрофильное высшее образование, желал бы переквалифицироваться в ИТ-профессионала. Ниже описаны некоторые аспекты основных компонент образовательной услуги ИТ-М: типовой образовательной программы, образовательного контента, процесса реализации образовательной услуги, процесса аттестации.

1. Содержание и принципы построения образовательной программы

Образовательные программы ИТ-М включают следующие основные части:

– вводные дисциплины (ядро);

– базовые дисциплины;

– курсы профилизации и практико-ориентированная подготовка.

Первые две части являются обязательными для всех образовательных программ ИТ-М. Содержание блока вводных дисциплин разработано таким образом, что включает порядка 70% объема знаний ядра (core), определенного в стандарте куррикулума [3] для бакалавров компьютерных наук. Вторая часть включает пять семестровых курсов магистерского уровня. Третья часть является вариабельной, и разрабатывается в зависимости от целей профильной подготовки. В качестве примера типовой программы ИТ-М в 1 представлено описание программы, ориентированной на подготовку программистов по профилю «Разработка корпоративных приложений», разработанной автором совместно с Сетевой академией Ланит.

Как отмечалось, учебные программы ИТ-М реализуются в режиме дистанционного обучения. В качестве системы электронного обучения используется система Moodle. Важнейшим компонентом такой формы обучения является образовательный контент, который формируется из электронных учебных курсов, реализуемых в виде учебно-методических комплексов (УМК). На данном этапе проекта принята в качестве базовой структура УМК, включающая следующие основные компоненты:

– программа курса

– учебник или учебное пособие по курсу в электронном виде

– электронный учебник в формате системы электронного обучения (Moodle), включая тестовый материал и средства контроля знаний и умений для данного учебного курса

– видео-лекции учебного курса.

2. Процесс реализации образовательной услуги

На обучение принимаются слушатели, имеющие диплом о высшем образовании сопутствующего профиля (например, инженерного, есте-

Таблица 1:

Вводные дисциплины (ядро)	Базовые дисциплины	Курсы профилизации и практико-ориентированная подготовка
Технологии e-Learning Основы языков программирования Операционные системы Компьютерные сети Технологии баз данных Основы программирования	Анализ информационных технологий Компьютерная безопасность Основы параллельного программирования Разработка вэб-приложений на языке Java Разработка распределенных приложений баз данных	Построение приложений-клиентов с использованием JDBC 4.0 База данных Oracle 11g: Основы SQL База данных Oracle11g: Основы PL/SQL и разработка модулей Разработка сервлетов / JSP-страниц Разработка компонентов EJB 3 Мобильное программирование на Java для платформы Android Технология SPRING Шаблоны проектирования под Java ВКР — выпускная квалификационная работа

ственнонаучного, педагогического), или студенты старших курсов соответствующих направлений и специальностей. Набор осуществляется по результатам очного собеседования или видео сеанса по скайпу. С отобранными для обучения слушателями заключается договор об оказании им платной образовательной услуги по реализации соответствующей образовательной программы в режиме дистанционного обучения. Планируемая стоимость обучения 30 000 рублей за один семестр (полная стоимость программы за четыре семестра — 120 000 рублей). После оплаты стоимости первого семестра учебная часть дополнительного образования готовит приказ на зачисление слушателя с последующим оформлением ему студенческого билета и зачётной книжки. Далее выдаётся логин и пароль для доступа к электронному контенту программы.

Обучение ведется на основании утвержденных учебных планов.

Учебные мероприятия проводятся в форме:

– занятий в режиме on-line (сетевое обучение);
– самостоятельной работы;
– контрольных мероприятий (текущий контроль усвоения материала);
– аудиторных занятий (в период очных экзаменационных сессий);
– процесса аттестации.

3. Процесс аттестации

Процесс аттестации включает две формы проведения:

– итоговый контроль работы за семестр (экзаменационная сессия);
– итоговая аттестация.

Итоговый контроль работы за семестр осуществляется в конце каждого семестра в форме экзаменационной сессии и предполагает традиционную (очную) форму — экзамен или зачет. Продолжительность такой сессии до 14 дней. Во время сессии предоставляются консультации перед сдачей экзаменов и зачетов, а также, возможно, проведение мастер-классов и практических занятий для закрепления у слушателей практических навыком, если такие занятия предусмотрены учебным планом. В случае необходимости иногородним слушателям на время сессии предоставляется общежитие по отдельному контракту с соответствующей процедурой поселения (оплата проживания за счет слушателей).

Итоговая аттестация проводится в конце обучения аналогично экзаменационной сессии и реализуется в традиционной (очной) форме:

– государственный экзамен;
– защита выпускной квалификационной работы (ВКР).

Заключение

Данный проект нацелен на создание эффективной системы дистанционной подготовки/переподготовки высококвалифицированных кадров для ИТ-отрасли на основе комплексных программ дополнительного образования магистерского уровня, позволяющих получить выпускникам соответствующую подготовку и диплом/сертификат МГУ имени М.В.Ломоносова о дополнительной квалификации к

высшему образованию. Такая форма обучения может представлять интерес для многочисленной армии выпускников вузов по невостребованным в экономике профессиям и желающих переквалифицироваться в ИТ-профессионалов, спрос на которых не ослабевает. Также опыт реализации описанных выше образовательных программ позволит отработать учебно-методическое обеспечение для реализации других видов образовательных услуг, включая базовые образовательные процессы подготовка бакалавров и магистров на основе широкого использования технологий e-learning.

Литература

[1] *В.А. Сухомлин.* Виртуальный национальный университет IT-образования: от концепций к реализации. Прикладная информатика. №3(15), 2008, с. 89-115.

[2] *Computing Curricula 2005 (CC2005).* Association for Computing Machinery and Computer Society of IEEE.

[3] *Computer Science 2008 (CS2008).* Association for Computing Machinery and Computer Society of IEEE.

Николай Непейвода
Переславль-Залесский, ИПС им. А. К. Айламазяна РАН

Можно ли информатика, обучавшегося традиционному программированию, переучить на алгебраическое?

В связи с тем, что на переднем крае компьютинга диагностировано объективное исчерпание технических возможностей, возникают вопросы о том, что будет дальше и в области железа, и в области программирования?

Уже сейчас для суперкомпьютеров на гибридной базе (процессоры разного типа, работающие в одном устройстве: сеточные, видео, для быстрого поиска информации) программирование на клонах C++ с дополнительными возможностями ручного распараллеливания и указаний об укладке программы на процессоры становится мучительным

мазохистским процессом. Но все попытки что-то изменить наталкиваются на агрессивное невежество (термин не мой) программистов.

Таким образом, еще раз подтверждается диагноз: человек, который изучал в вузе языки программирования, а не концепции информатики, ничего нового в жизни уже знать не пожелает. Он будет добавлять чисто технические сведения о конкретных инструментах, но работать тем же топором, который ему дали.

Но намечается еще более радикальный поворот: переход и на уровне аппаратуры, и на уровне алгоритмики, и, соответственно, на уровне программирования от численного моделирования к структурному. При таком подходе программа становится алгебраической структурой. и данные нечисленными. В докладе будут показаны элементарные, но полностью выходящие за рамки традиционной парадигмы, примеры алгебраических программ [1, 2, 3].

Возникает вопрос: может ли человек, которого учили по традиционной схеме, быть переучен на структурное алгебраическое описание систем и программирование? Почти никогда.

И причина здесь не только в курсах программирования. Само фундаментальное математическое образование здесь должно быть иным, о чем я уже неоднократно говорил. Оно должно базироваться на алгебре и логике.

Возникает вопрос: а как же совместить фундаментализацию с катастрофическим падением уровня студентов? (Добили-таки школьное образование и добивают вузы). Ответ прежде всего в том, что на самом деле эта ситуация явилась вызовом для нашего привычного стиля изложения фундаментальной науки и прикладных аспектов. Если мы выживем в ней, то придется находить новые способы подачи и того, и другого. И прежде всего необходимо тесно увязывать курсы друг с другом и давать не сумму начитанных сведений, а строго отобранный базис, на который можно затем проектировать множество новых знаний. А искусство проекции нужно тоже отрабатывать сразу же: показывая самые абстрактные вещи во взаимосвязи с конкретными, а когда рассказываются конкретные, сразу же давая несколько вариантов их представления, чтобы отучить от зашоренности взгляда. Таким образом, нужны новые фундаментальные и прикладные курсы, причем базисные курсы ни в коем случае не должны привязываться к конкретике традиции, зато к конкретике жизни и ее многообразию.

Скажем, программировать лишь на C++ или Java — формировать тех, кто никогда не сможет и не захочет думать иначе, как в терминах

битов, команд и последовательных процессов. Поэтому очень полезный опыт на первом же курсе был бы переписывание практически всех программ на языках разных стилей (и заодно показ того, когда получается уродливо и не стоит впихивать программу в другую парадигму).

Литература

[1] *Nepejvoda N. N.* Reversivity, Reversibility and Retractability Third international Valentin Turchin Workshop on metacomputation Переславль 5–9 июля 2012 ISBN978-5-901975-28-6 p 203–216.

[2] *Непейвода Н. Н.* Абстрактные алгебры различных классов программ. Аппликативные Вычислительные Системы 3-я международная конференция АВС 2012 Москва, Институт «ЮрИнфоР-МГУ», 14–15 декабря 2012 г. ISSN 2304-7283. Стр. 103–128.

[3] *Непейвода Н. Н.* От численного моделирования к алгебраическому РАСО'2012 т. 1 М.: 2012 ISBN 978-5-91450-122-5 стр. 93–103.

В. А. Старых, А. Ю. Кузнецов, А. А. Карасёв, М. А. Ушанов
Москва, ФГАУ ГНИИ ИТТ «Информика»
http://www.informika.ru

Портал «Свободное программное обеспечение в образовании»

Портал «Свободное программное обеспечение в образовании» создан для информационного обеспечения и дистанционного обучения учителей школы и административных работников. Задача выполняемая порталом — реализация единой гармонизированной с международными стандартами, схемы метаданных электронно-образовательных ресурсов интегрированных из различных источников в едином хранилище с возможностью динамического добавления и изменения содержимого Портала, структурного представления и поиска информации. Критерии реализации разработки:

- Реализация механизмов поиска/фильтрации электронно-образовательных ресурсов в соответствии с заданными пользователем критериями.

- Реализация масштабируемости, обеспечивающая возможность расширения структуры Портала, создания новых разделов и функциональных сервисов.
- Обеспечение технологичности — использование современных открытых веб-технологий.
- Обеспечение модульности — использование решений, состоящих из обособленных замкнутых функциональных программных элементов, реализованных как сервисы.
- Программный инструментарий портала разработан и функционирует на основе кроссплатформенного свободно распространяемого программного обеспечения:
- Среда выполнения Java Runtime Environment 1.6.0 или выше.
- HTTP-сервер с поддержкой балансировки нагрузки для обеспечения возможности организации кластерной архитектуры подсистемы каталога и поиска.
- Сервер web-приложений, реализующий спецификации J2EE Java Servlet 2.4 и JSP 2.0.
- FTP-сервер

Организация и сопровождение учебных процессов обеспеченны встроенным функционалом платформы Moodle.

В. А. Старых, М. А. Ушанов, В. И. Одинцов, М. В. Крахин
Москва, ФГАУ ГНИИ ИТТ «Информика»
http://www.informika.ru

Интеграция мобильных устройств в составе учебного класса общеобразовательного учреждения

Планшетные компьютеры, или «таблетки», ноутбуки и нетбуки, коммуникаторы и Интернет в очередной раз меняют нашу жизнь подобно тому, как когда-то ее изменили радио, телевидение и мобильная телефонная связь. Единственным препятствием массовому внедрению таких современных мобильных средств вычислительной техники для отечественной школы является их стоимость. Одним из решений проблемы снижения стоимости является использование на таких

платформах свободного программного обеспечения (СПО) с нулевой стоимостью свободных лицензий. Разработанный программный комплекс «Информика Школьный» на основе свободного программного обеспечения предназначен для организации учебного процесса в образовательных учреждениях общего образования, посредством обеспечения беспроводного сетевого взаимодействия как обычных персональных компьютеров на платформе x86, так мобильных компьютерных платформ типа «нетбук», «электронный планшет» и «электронная книга». Преимущества комплекса:

- Комплекс разработан на базе свободно распространяемой операционной системы AltLinux 6.0 и специализирован для интеграции современных мобильных средств вычислительной техники (нетбуков, планшетов, стационарных компьютеров) учеников и учителей.
- Нулевая стоимость лицензий ПО. Загрузить весь программный комплекс можно по адресу school.informika.ru/download .
- Документация ко всем дистрибутивам представлена на русском языке.
- Мобильность. Это позволяет ученику и учителю легко включаться в образовательный процесс благодаря использованию нетбуков или планшетов как дома, так и в составе учебного класса со всеми его сервисами и системами.
- Для продукта доступна полная поддержка от ФГАУ ГНИИ ИТТ «Информика» и ООО «АльтЛинукс».

Пакет СПО **Информика Школьный** ориентирован на поддержку взаимодействия трех участников: Ученик, Учитель, Сервер. Роль каждого участника выполняется одним из компонентов комплекса:

1. Ученик. Реализуется двумя дистрибутивами — «Информика Школьный Ученик» или «Информика Школьный Планшет»;
2. Учитель. Реализуется дистрибутивом «Информика Школьный Учитель»;
3. Сервер. Реализуется дистрибутивом «Информика Школьный Сервер».

Пакет СПО **Информика Школьный Ученик** предназначен для установки на компьютеры учеников (за исключением планшетных компьютеров), содержит системное и прикладное программное обеспечение, необходимое ученику для выполнения повседневных задач.

Дистрибутив включает операционную систему АльтЛинукс 6.0 (Платформа 6), а также прикладное программное обеспечение. Пакет СПО Информика Школьный Планшет содержит те же приложения, что и «Информика Школьный Ученик», но предназначен для установки на планшетные компьютеры. В дистрибутиве «Планшет» установлена графическая система «Gnome 3» — одна из немногих, поддерживающих использование сенсорных устройств ввода. Пакет СПО **Информика Школьный Учитель** для обеспечения работы учителя. Включает в себя программы, входящие в состав модуля «Информика Школьный Ученик», но вместо клиентской части ПО для мониторинга iTALC, содержит ее серверную часть, позволяя использовать такие функциональные возможности как контроль содержимого любого экрана рабочего места ученика, включая возможность его блокировки, а также проекцию содержимого экрана рабочего места учителя на экраны рабочих мест учеников. Пакет СПО **Информика Школьный Сервер** является центральным, связующим звеном всего комплекса. Обеспечивает централизованную аутентификацию, синхронизацию данных, работу дополнительных образовательных сервисов (MediaWiki, Moodle, OwnCloud и РУЖЭЛЬ), фильтрацию контента, централизованное обновление программного обеспечения на учительских и ученических ПК, повторную установку (при необходимости системного и прикладного программного ПО).

Анатолий Якушин
Москва
Компания ALT Linux `http://www.altlinux.ru`

СПО — кризис среднего возраста?

Аннотация

В докладе рассматриваются особенности движения СПО на современном этапе.

Нынешний год является весьма заметных для всех, чья жизнь и работа связаны со свободным программным обеспечением. Тридцать лет назад Ричард Мэттью Столлман впервые сформулировал и опубликовал основные принципы проекта GNU. За эти тридцать лет фактически сформировлась всемирная отрасль информационных технологий и все знаковые события, которые происходили в отрасли, так или

иначе отразились на движении СПО, неоднократно меняли взгляды участников движения на положение свободных программ в быстроменяющемся мире IT.

Сегодня можно однозначно утверждать, что основные идеи СПО оказались востребованными отраслью, прошли проверку «священными войнами», пережили «кризис доткомов», воспитали несколько поколений блестящих программистов и грамотных пользователей.

Однако последние несколько лет появились и стали нарастать некоторые признаки снижения активности движения в целом. Уменьшается количество публикаций на тему СПО как в популярной, так и в научной прессе, стагнировало или уменьшается количество участников свободных разработок, отсутствуют яркие акции по продвижению идей свободных программ. Особенно негативные явления заметны в Высшей школе, особенно американской, где на смену парадигме «открытый код в обмен на гранты» пришло массовое использование стартапов и накопление патентного пула учебных заведений [1].

Являются ли эти явления полноценным кризисом движения СПО или это проявление зрелости сообщества, нахождение понимания роли и места сообщества в информационном пространстве? Проведенный анализ показывает, что скорее вернее второе утверждение, чем первое.

Основные положения СПО не изменились за последние 30 лет. Это всё те же 4 принципа свободы, сформулированные RMS. Однако за годы работы удалось развить и сформулировать все аспекты жизненного цикла свободных программных продуктов, вытекающих из классических принципов.

Сегодня мы знаем, что:

1. СПО — это коммерческое программное обеспечение, разрабатываемое и поддерживаемое в рамках соответствующей специализации бизнеса на заказ для конкретного конечного пользователя, либо с расчетом на заранее не определенный круг конечных пользователей с целью извлечения прибыли. [2]

2. Совокупная стоимость владения (TCO) свободного программного продукта всегда меньше, чем проприетарного, как минимум на стоимость проприетарной лицензии.

3. Существует значительное количество проверенных временем и рынком методов монетизации СПО.

4. СПО не в коей мере не противоречит авторскому и патентному праву, а базируясь на них, предоставляет пользователю наименее обременительную реализацию прав автора произведения.

Литература

[1] The MIT Technology Licensing Office Frequently Asked Questions. MIT pub. 2012.

[2] М. Отставнов «Перспективы свободного программного обеспечения в сфере государственного управления и бюджетном секторе экономики».

Михаил Ройтберг
Москва, НИИСИ РАН

ЕГЭ по информатике — итоги 2012 г.

Аннотация

В 2012 г. набор заданий в ЕГЭ по информатике был существенно изменен по сравнению с предшествующими годами за счет включения новых задач по алгоритмам и программированию, а также «олимпиадноподобных» задач. Будет представлен обзор результатов ЕГЭ.

1. Общие сведения

Единый государственный экзамен по информатике в 2012 г. сдавали около 57000 человек. Это составляет примерно 6,5% от общего числа участников ЕГЭ текущего года.

Экзаменационная работа содержала 32 задания и состояла из трёх частей. В первой части работы (А) содержалось 13 заданий с выбором ответа (выбор одного правильного ответа из четырех предложенных). Во второй части (В) были собраны 15 заданий с краткой формой ответа. Третья часть (С) содержала 4 задания, подразумевавшие запись в произвольной форме развернутого ответа.

По темам задания можно разделить на 3 блока: «Математические основы информатики», «Алгоритмы и программирование», «Технологии». На блок «Алгоритмы и программирование» приходилось 13 заданий, в том числе — все задания части С.

2. Отличия от 2012 г.

С количественной точки зрения — уменьшение количества задач 1-й группы (с 18 до 13) и соответственное увеличение количества задач 2-й группы (с 10 до 15). Количество задач 3-й группы и общее количество задач осталось неизменным.

С качественной точки зрения следует отметить следующее.

1. Введение новых типов заданий по теме «Элементы теории алгоритмов» на позициях B7, B13, B14. Это задания на качественный анализ алгоритмов, их невозможно решить путем пошагового выполнения алгоритмов.
2. Общая тенденция на контроль неформального понимания различных разделов курса.
3. Уменьшение риска случайных ошибок. Была снижена «арифметическая» сложность заданий. Кроме того, многие задания наряду с решением «в лоб» имели и решение, требующее более глубоких знаний, в котором сделать случайную ошибку практически невозможно (см., например, задания A1, A2, B4).

3. Результаты.

Минимальную границу (8 баллов) не преодолели 11,1% сдававших Максимальную оценку в 100 баллов получили в 2012 г. 315 человек, то есть 0,5% участников экзамена. Результаты ЕГЭ 2012 г. отражают наличие среди сдававших нескольких групп, которые отличаются друг от друга по уровню подготовки. Были выделены четыре такие группы, к которым относятся ученики, набравшие соответственно 0–7 («минимальный уровень подготовки»; 11% сдававших), 8–19 («базовый уровень подготовки»; 32%), 20-30 («хороший уровень подготовки»; 34%) и 31–40 («отличный уровень подготовки»; 23%) баллов.

Литература

[1] `http://www.fipi.ru/binaries/1364/2.11.pdf`

Е. А. Роганов, А. А. Харченко
Москва, ФГБОУ ВПО МГИУ
http://www.msiu.ru

Опыт использования свободного программного обеспечения в Московском Государственном Индустриальном Университете

В восьмидесятых годах прошлого века Завод-ВТУЗ при ЗИЛе, как тогда назывался МГИУ, располагал вычислительным центром с машинами СМ-4, СМ-1420 и ЕС-1055, что было совершенно типичным для того времени. Затем в институте стали появились первые персональные компьютеры — IBM PC XT, работавшие под управлением MS DOS. Через несколько лет после списания морально устаревших «больших» машин серий СМ и ЕС их место занял класс компьютеров Intel 386, которые были бездисковыми и обслуживались сервером под управлением NetWare. С момента установки этого класса в 1995 году система не менялась и к 1997 году морально и физически устарела.

На рабочих станциях «Беста», созданных российской фирмой «Сапсан», изготовленных на базе процессора Motorola m68k, с мультиплексорами, к которым можно было подключить до 32 алфавитно-цифровых терминалов, а также с одним графическим адаптером, к которому подключался графический цветной дисплей, была установлена многопользовательская и многозадачная операционная система — немного урезанная версия UNIX System V. «Бесты» изначально были предназначены для решения задач САПР, но достаточно быстро стали использоваться преподавателями кафедры общей и прикладной математики также и для обучения студентов специальности «Системы автоматизированного проектирования».

В 1996–1997 годах на кафедре общей и прикладной математики началось освоение свободной операционной системы Linux, которую устанавливали на персональные компьютеры на базе процессора Intel. В 1997 году руководством вуза было решено создать компьютерные классы на основе бездисковых компьютеров, загружаемых по сети, с сервера для обучения старшеклассников подшефных школ информатике и информационным технологиям. С этого момента свободное программное обеспечение стало активно использоваться в учебном процессе в института.

Отметим, что к концу 2001 года в университете работало уже более десяти компьютерных классов общего доступа, рассчитанных на 25–28 человек каждый, а число зарегистрированных пользователей информационно-вычислительной среды превышало 6400 человек. Кроме студентов университета в классах занимались учащиеся подшефных школ — более 2000 старшеклассников изучали основы информатики и информационных технологий по 3 часа в неделю.

В 2002 году на одном из своих заседаний Учёный совет МГИУ принял принципиальное решение о необходимости широкого использования свободного ПО и финансирования работ, этому способствующих, а согласно приказу Минобразования наряду с такими вузами, как МГТУ им. Баумана, МИФИ, МФТИ, Санкт-Петербургский, Петрозаводский и Ростовский университеты, МГИУ был включён в число 13 вузов — участников первого этапа реализации федеральной целевой программы «Электронная Россия (2002–2010 годы)».

Недавнее постановление Правительства РФ «План перехода федеральных органов власти и федеральных бюджетных учреждений на использование свободного программного обеспечения» на период с 2011 до 2015 года в значительной мере уже может считаться выполненным МГИУ. Более того, богатейший опыт использования свободного ПО как при решении задач автоматизации, так и для организации учебного процесса может оказаться весьма полезным многим другим университетам и иным бюджетным учреждениям.

Георгий Курячий
Москва, ALT Linux, МГУ им М. В. Ломоносова, ф-т ВМиК
http://uneex.ru

Использование языка программирования Python в качестве базового при обучении специалистов

Аннотация

Язык программирования Python обладает рядом уникальных или просто полезных достоинств, позволяющих использовать его в качестве базового ЯП при обучении специалистов весьма широкого диапазона — от профессиональных программистов и инженеров до научных работников в области естественных и гуманитарных наук. Достоинства эти отчасти вполне объективного, а отчасти и весьма субъективного

толка. Как следствие, наш обзор не будет претендовать на сугубую объективность, опираясь, с одной стороны на личный опыт преподавания в различной аудитории, а с другой — на знания о возможностях самого языка. Цель обзора — показать поразительную универсальность среды Python в качестве возможной базы для программистской части ИТ-образования. Не является рекламой. Имеются противопоказания, проконсультируйтесь со специалистом.

Общие слова

Python до сих пор принято считать «молодым» языком программирования, хотя он слегка старше Java (официальный «год рождения» — 1989). Дело в том, что Python — очень динамично развивающийся язык, в котором непрерывно идёт процесс обновления и расширения как самого ЯП, так и инструментальной среды, входящей в стандартную поставку.

Это, кстати, сразу обращает наше внимание на первые три достоинства Python в качестве именно языка обучения.

(1) Всегда впереди. Во-первых, разработчики стараются удержаться на «передовом крае науки», видоизменять язык в соответствие с новшествами в области программирования (разумеется, дозированно и после тщательного обсуждения) [1]. Следовательно, в какой-то мере решён вопрос «устаревания инструмента», довольно остро стоящий в отношении, например, учебных программ, основанных на ЯП Pascal. Правда, возникает вопрос устаревания самой учебной программы... Впрочем, если по какой-то причине изменения программы надо подтормозить, в нынешней ситуации можно остановиться на Python [2], развитие которого заморожено в пользу Python [3].

(2) Батарейки внутри. Во-вторых, Python — это не только язык программирования, но и довольно полная инструментальная среда общего назначения, насчитывающая более полутора сотен модулей на все случаи жизни (связь с ОС, шаблоны и алгоритмы программирования, эффективные и сложные типы данных, работа с различными форматами данных, сеть, вычислительная математика и т. д.) [2]. Список не входящих в поставку, но зарегистрированных на сайте «Python Package Index» и доступных к использованию модулей в полтораста раз больше [3]. Это позволяет расширить круг учебных задач за счёт предметных областей, затрагиваемых в модулях, и не

привлекать при этом сторонние, не совместимые с базовым, инструментарии.

(3) С ним не пропадёшь. В-третьих, Python — очень востребованный язык программирования. Некоторые организации (RedHat4, NASA5. Google6) прямо заявляют, что ведут разработку или часть разработки на Python, а количество написанного на Python прикладного программного обеспечения любой сложности вообще вряд ли поддаётся исчислению. Стало быть, на одном только Python ученик без куска хлеба не останется. Хотя о том, что бы был не один только Python, да и масло на хлебе, стоит тоже позаботиться.

Python как первый язык программирования

Гвидо ван Россум, автор языка программирования и «бессменный великодушный диктатор» сообщества Python, в своём «Пособии к Python» замечает, что пособие это — для тех, кто уже умеет программировать [7]. В самом деле, изложение в нём плотное, с постоянной отсылкой на различные программистские реалии. Но структура «Пособия» довольно прозрачно намекает на то, каким может быть вариант учебной программы «с нуля».

(4) Плавно въезжать. Пользуясь тем, что Python — это интерпретатор со встроенной системой помощи, «Пособие» предлагает неожиданно традиционалистский подход к обучению программированию. Поначалу можно использовать Python как калькулятор всевозможных математических (а также и строковых) выражений. При решении, скажем, квадратного уравнения естественно возникает вопрос о промежуточном хранении вычисленных объектов — так возникает идея имён (переменных в строгом смысле в языке нет), а заодно возникает идея условного выполнения действий. Сами объекты обладают массой полезных свойства — методами. Полезную последовательность действий можно записать в функцию. Для прохода по последовательностям предусмотрены циклы... Клубочек можно ещё глубоко разматывать, добраться до исключений, классов, множественного наследования, функциональных элементов и многого другого. Не забыть только научиться записывать программу в файл.

(5) Легко читать. Из опыта работы со школьниками (и не только) известно, что для эффективности обучения очень важно соблюсти баланс между размером контекста и его плотностью. Говоря проще, реализация алгоритма должна умещаться на одном экране, но всё,

что на этом экране находится, должно быть — при известном умственном усилии — понятно до конца. Одну и ту же программу на Python можно написать и «под Pascal» (длинно), и «под Lisp» (в две-три немыслимые строки), и «под Python» (с использованием подходящих высокоуровневых конструкций и модулей). Отсутствие описаний и операторных скобок, множественное присваивание, выражения-конструкторы, лаконичные составные типы данных и операции над ними и многое другое позволяют писать программы коротко, но разборчиво [8]. И — да, отступы как обязательный элемент синтаксиса здесь тоже к месту.

(6) Понятно даже психологу. Не менее успешен был опыт «блиц-обучения» нескольких студентов факультета Психологии МГУ — не с целью подготовки программистов, а с целью дать в руки инструмент, достаточный для ввода-вывода и обработки данных. Здесь помогает то, что для многих простых подзадач в Python имеется готовая реализация в виде конструкции ЯП или модуля. Так, подавляющее большинство структур экспериментальных данных вписываются в питоновские списки или словари, интерфейс организуется из готовых интерфейсных блоков, а собственно анализ данных программировать и не надо, для этого есть специализированные инструменты. Кроссплатформенность инструментальной среды освобождает человека-непрограммиста от необходимости осваивать их несколько.

В «удобстве» Python кроется часто обсуждаемый недостаток. Считают, что, изучив «удобный» Python, человек не захочет переходить на «неудобный» низкоуровневый Си или многословный Java. Вот так же и автора этих строк учили сначала программировать на Фортране, а затем уже — на Си именно из соображений «не захочет потом Фортран учить, а надо».

Python как «продвинутый» язык программирования

Сказанное выше не означает, что Python остаётся в первую очередь учебным языком программирования, каким задумывался его прототип — язык ABC — более двадцати лет назад. Напротив, по истечении периода становления, язык быстро набрал популярность и в области высокотехнологичного, и в области промышленного программирования.

(7) Сахар и сливки. Первое, чем завлекает Python опытного программиста — это обилие «синтаксического сахара» и вообще вся-

ческих конструкций, сводящих синтаксический шум к минимуму и повышающих плотность программного текста. Большую роль играет и принцип «сведения к простому»: например, в Python нет отдельной перегрузки операций, потому что операции — это и есть соответствующие методы объекта. Ещё более показательный пример — использование «утиной типизации» (суть которой в следующем: если для работы с уткой используется всего два метода — .крякать() и .плавать(), то любой объект с такими методами — это утка, независимо от того, что ещё может объект (например, .летать() или .тарахтеть()). Отдельное удовольствие доставляют доведённые до логического завершения идеи языка Си о неполном вычислении и интерпретации логических выражений [9].

(8) Мультипарадигмальность. Традиционный императивный язык программирования (как, собственно, и любой другой) не может претендовать на полный охват различных парадигм программирования в рамках единого синтаксиса. К тому же философия Python требует, чтобы всякое расширение процедурной по сути парадигмы Python естественно следовало из имеющихся возможностей языка. Это не мешает «выводить» из свойств объектов весьма современную и гибкую объектную модель (см. п. **(4)**). Не менее естественно выглядят элементы функционального программирования [10], особенно в сочетании со списками, генераторами, функционалами и прочими стандартными средствами языка. Декларативная парадигма подразумевает несколько иную организацию данных и диалога с пользователем, и здесь на помощь приходит Python-интерпретатор командной строки. Реализация логического программирования упаковывается при этом в модуль, а диалог ведётся стандартными средствами (таких модулей в сети немало, но нам не доводилось анализировать их на предмет пригодности для учебного процесса). Наконец, событийное программирование явной поддержки в языке не имеет, зато, в силу специфики области применения (моделирование систем, сеть, GUI), представлено весьма мощными и многочисленными модулями в традиционном стиле «объекты + обратные вызовы + главный цикл + параллельность» с упором на эту самую специфику.

Нелишне заметить: когда учебная программа требует глубокого знакомства с парадигмами программирования и располагает соответствующим временем в сетке, разумнее изучать различные среды программирования, ориентированные на ту или иную парадигму. Но и здесь современное пространство языков программирования предла-

гает известную свободу манёвра. Например, весьма примечательным представляется функциональный язык программирования Pure (многими полезными свойствами он похож на Python).

(9) По-взрослому. Будучи не только языком, но и средой программирования, Python обладает всеми признаками инструмента разработки сложных систем. Многоуровневая модульность, динамическое документирование кода (это когда строка документации является частью объекта), поддержка модульного тестирования и технического документирования (как обычно, сразу несколькими модулями), собственная система пакетирования и развёртывания программных компонентов — всё это позволяет использовать Python в качестве базы для изучения синтетических дисциплин, наподобие технологии программирования, систем быстрого проектирования приложений, организации совместной разработки и т. п. Если область интересов ограничивается только технологическими вопросами, Python тоже вполне привлекателен, ибо обеспечен полной поддержкой систем индустриального программирования (как со стороны IDE Eclipse, NetBeans и Visual Studio, так и со стороны виртуальных машин Java и .Net). Система документирования исходного кода Sphinx, написанная на Python, оказалась настолько удобной, что теперь с её помощью создаётся штатная документация и к самому Python, и к многим проектам, на нём основанным, и к множеству других проектов, не имеющих к нему отношения [11].

Прикладные свойства

Рискнём повториться: универсальность и модульная структура среды Python делает её орудием не только програмиста, но и вообще всякого, в чью деятельность программирование входит только как один из инструментов решения прикладных задач. Причём задачи могут быть очень разной ширины и глубины: математика, естественные науки, моделирование процессов; и от поверхностного введения в область до профессионального инструмента.

(10) В игрушечки играете? Наглядность языка позволяет нещадно эксплуатировать его в режиме «напиши-используй-выбрось», так как появляющиеся при этом сценарии-однодневки выходят короткими и создаются со скоростью написания исходного кода практически без отладки. Самое частое применение — поспрашивать пользователя о входных данных, ввести их из какого-нибудь

файла, пожевать, переварить и выплюнуть в какой-нибудь другой файл. В *NIX-стилистике для этого существует командная оболочка (ввод-вывод — текст и командная строка, обработка данных — POSIX-утилиты в составе операционной системы), однако по нынешним временам это часто неудобно (как субъективно, с непривычки, так и объективно, из-за отсутствия нужных утилит и необходимости зрительного поиска). В кроссплатформенной python-стилистике — это несколько диалоговых окон (с полдюжины строк кода на каждое, включая сюда диалог выбора файла) или чуть более сложных форм плюс передача данных на обработку прикладному модулю (в том числе и операционной системе). Другой пример: опытный преподаватель не желает отказываться в учебном процессе от использования Turbo Pascal или даже Quick Basic, потому что в них «есть учебная графика» (SCREEN 13 и CIRCLE (20, 25), 5, 1, помните?). Важно чтобы первые написанные учеником программы делали что-нибудь ощутимое. Но вот программа, рисующая окружность на Java, вряд ли пройдёт в качестве первой. Между тем, для Python имеется некоторое количество графических модулей, от простейшей «черепашьей» графики (модуль turtle) до весьма многоплановой мультимедийной подсистемы PyGame (фактически повторяющей возможности специализированной библиотеки SDL) [12]. PyGame оказалась куда лучшей приманкой, чем старая добрая BGI-графика: как же! написать свою игру за пару дней! А ведь под это подтягиваются вполне серьёзные приёмы программирования: обработка событий, таймеры, использование производных классов...

(11) Академический код... Отдельное внимание хочется уделить большим научным пакетам, использующим Python для обработки данных, а то и просто написанным на нём. Например, проект SciPy [13] включает в себя модуль NumPy — эффективную реализацию многомерных массивов, и инструментарий к ним, а на базе NumPy реализует целое множество научных математических инструментов. Эдакий Matlab на Питоне. Главное достоинство таких проектов — научное и программистское сообщество, в котором становится нормой оформлять соответствующие результаты научной деятельности в виде программного модуля, публикуемого на специальном портале [14]. Сам модуль может быть примером использования SciPy, а может предоставлять и интерфейс к другим мощным инструментариям (например, к статпакету R). Введение таких инструментариев в учебную программу позволяет перекинуть довольно надёжный мостик меж-

ду чистой теорией того или иного раздела математики и зубодробительными методами её программной реализации. Кроме того, опыт пакетирования различных наукоёмких программных продуктов под Linux показывает, что т. н. «академический код» на Python выглядит куда менее ужасно, чем таковой же на C или Java. «Академический код» — это исходный код программы, написанной учёным, а не то — его аспирантами, обычно для подтверждения теории или реализации некоторых её положений; нередко, увы, являет собой образец довольно низкой культуры программирования. Как следствие, аналогов такая программа обычно не имеет, и, хочешь-не хочешь, приходится регулярно с академическим кодом возиться.

(12) . . . и все-все-все. Сказанное о SciPy и математике относится вообще к любой деятельности, требующей обучения с практической реализацией в виде программного продукта. Собственно, вокруг всякого популярного языка программирования достаточно быстро образуется обширная прослойка прикладных библиотек и систем, и ими можно вполне успешно пользоваться в учебных целях. С другой стороны, для популярных прикладных областей часто существует некий набор уже готовых пакетов прикладных программ, и их тоже можно с немалым успехом в тех же целях использовать. Но если наша цель — формирование восприятия области профессионального знания, понимания принципов работы профессионального инструментария, и — главное — умения этот инструментарий достраивать для решения собственных задач, то без упомянутого «мостика», позволяющего самостоятельно подходить к вопросам реализации теории, обойтись сложно. И здесь количество — наглядность и компактность кода вкупе с быстротой освоения языка и разнообразием прикладных модулей — переходит в качество: удаётся втиснуть в сетку занятий не только формулы и названия ручек, за которые надо дёргать для их вычисления, но и сам процесс организации этих вычислений. Думается, эффект от такого перехода будет напоминать эффект от изобретения языка Фортран, хотя и отнюдь, отнюдь не столь революционный. В естественных науках появляется возможность не только собирать и обрабатывать экспериментальные данные, но и публиковать окружение для воспроизведения эксперимента. В областях, ориентированных на информационные технологии — возможность анализа и сравнения.

Сложим кубики

Если немного продлить оптимистический пафос предыдущего параграфа, начинает выстраиваться довольно гармоничная картина.

1. Почти вне связи с содержательной стороной обучения (лишь бы требовалось программирование в нескольких различных контекстах) язык программирования Python выбирается в качестве базового. Это «язык, который знаются все, хотя бы теоретически». Обоснование: когда-то для этой цели применялся Алгол, затем настало время священной войны «сишников и паскалистов», современные же кандидаты — Java и C++ — в наглядности, но нашему мнению, Python проигрывают.
2. Основные программистские дисциплины — и в расчёте на собственно программистов, и для неспециалистов — ведутся также на Python, ибо он для этого дела ничуть не хуже других языков, а по нашему мнению — так и лучше.
3. Углублённое изучение как программирования, так и сопутствующих ему практик (технология программирования, совместная разработка и т. п.) практически полностью покрывается Python. Исключение — глубокое изучение различных парадигм программирования профессиональными информатиками.
4. Изучение программирования в приложении к актуальным предметным областям тоже с высокой вероятностью может быть поддержано Python и пакетами к нему. Исключение — обязательное изучение какого-то инструмента, уникального для данной предметной области.
5. ...
6. Профит!

P.S.

- Убедительный пример эффективности подхода «единой базовой среды» — работы Е. Р. Алексеева с применением математического пакета и языка программирования SciLab [15].
- Не менее убедительный пример доверия Python роли базового ЯП — переход большинства образовательных траекторий MIT с Lisp на Python [16]

- Свободное лицензирование самого Python и подавляющего большинства связанных с ним проектов обеспечивают лицензионно-правовую прозрачность материалов и разработок и известную свободу выбора инструментов. С другой стороны, оно сильно понижает коррупционную составляющую всего процесса.
- Всё вышеизложенное имеет смысл только при условии, что ВУЗ отказывается от практики эндорсмента программных продуктов, т. е. от повсеместного внедрения в учебный процесс курсов и методматериалов, в неизменном виде получаемых от разработчика ПО, и возвращается к традиционной академической практике самостоятельной разработки учебных программ и адаптации к ним программных продуктов.

Литература

[1] `http://www.python.org/dev/peps/`

[2] `http://docs.python.org/3.4/py-modindex.html`

[3] `http://pypi.python.org/pypi`

[4] `http://www.redhat.com/magazine/012oct05/features/python/`

[5] `http://www.python.org/about/success/usa/`

[6] `http://google-styleguide.googlecode.com/svn/trunk/pyguide.html`

[7] `http://docs.python.org/3/tutorial/index.html`

[8] `https://uneex.ru/LecturesVMSH/Python/2012-11-16#compact`

[9] `http://docs.python.org/3/reference/expressions.html#boolean-operations`

[10] `http://docs.python.org/3/library/functools.html`

[11] `http://sphinx-doc.org/examples.html`

[12] `http://www.pygame.org/`

[13] `http://www.scipy.org/`

[14] `http://scipy-central.org/`

[15] *Алексеев Е. Р., Чеснокова О. В., Рудченко Е. А.* Решение инженерных и математических задач в пакете Scilab. — М.: ALT Linux, 2008. — 257 с.

[16] http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/

Сергей Мартишин, Владимир Симонов, Марина Храпченко
Москва, Институт системного программирования РАН, Московский городской педагогический университет

Дипломное проектирование на СПО

Аннотация

Рассмотрены программные средства СПО для выполнения дипломного проектирования по специальности «Информационные системы и технологии»

Дипломный проект является выпускной квалификационной работой студента, подводящей итоги обучения в ВУЗе, и должен продемонстрировать теоретические знания и практические навыки выпускника и его готовность к самостоятельной работе по специальности.

В дипломном проекте по специальности «Информационные системы и технологии», связанном с разработкой или усовершенствованием информационной системы (ИС), выпускник должен:

- исследовать предметную область, проанализировать всю доступную информацию по функционированию подобных систем в данной или смежных предметных областях и показать актуальность решаемой задачи;
- выделить объект информатизации (автоматизации), сформировать требования к ИС;
- обосновать проектные решения по реализации поставленной задачи;
- разработать информационную модель системы и схему взаимодействия модулей;
- реализовать функционал и интерфейс, дающий возможность пользователю решать свои задачи в рамках ИС;
- оценить экономическую эффективность разработки.

Из сказанного выше следует, что основными и наиболее трудоемкими этапами дипломного проекта являются: формирование требований к разрабатываемому продукту, проектирование и реализация ИС.

Очевидно, что для выполнения дипломного проектирования необходимо использовать такое программное обеспечение, которое, во-первых, является доступным, во-вторых, обладает необходимой функциональностью, в-третьих, требования к аппаратным ресурсам должны быть как можно ниже. Необходимо также, чтобы инструментальные средства были согласованы между собой.

Среди множества программных средств для проектирования и разработки в последнее время появилось значительное число средств, являющихся свободным программным обеспечением (СПО), которые позволяют в значительной степени автоматизировать весь процесс создания программного обеспечения (ПО). СПО, как следует из самого понятия, является доступным и может быть получено путем скачивания через Интернет бесплатной версии для установки, скопировано с диска или просто взято у коллег.

Основной вопрос состоит в выборе операционной системы (ОС). В настоящее время все большую популярность в ВУЗах приобретает ОС Linux. Практически все Linux-системы являются СПО. Заметим, что для его установки достаточно обычного офисного компьютера с процессором, начиная с Pentium IV или аналогичного от AMD и 1 Gb оперативной памяти. В качестве web-сервера под ОС Linux используется Apache, который является свободным web-сервером.

Поскольку на этапе формирования требований проводится изучение объекта информатизации и предметной области, описание бизнес-процессов системы («AS-IS» и «TO-BE») в некоторой стандартной нотации, то необходимо выбрать соответствующие средства. В настоящее время для моделирования бизнес-процессов широко используется нотация BPMN (Business Process Modeling Notation), разработанная Business Process Management Initiative, поддерживаемая Object Management Group. В качестве СПО для моделирования бизнес-процессов можно использовать один из следующих программных продуктов: ARIS Express, Modelio Free Edition (open source), Intalio|BPMS Community Edition (open source) и пр.

На этом же этапе следует осуществить выбор ПО для реализации проекта. Как было описано в [1] для данных целей наиболее удобным оказалось использование LAMP — набора серверного программного обеспечения. Помимо ОС Linux и web-сервера Apache, LAMP включа-

ет в себя СУБД MySQL и язык программирования, используемый для создания web-приложений PHP (или Perl), то есть содержит все необходимые для реализации ИС компоненты Преимуществом использования LAMP является простота установки, полная согласованность используемого программного обеспечения и невысокие требования к ресурсам.

На этапе проектирования вне зависимости от того, принято ли решение о выполнении полностью уникальной разработки или частичном использовании готовых продуктов, необходимо произвести проектирование системы, включающее проектирование архитектуры системы и детальное проектирование. Основу системного проекта составляют модели проектируемой ИС, которые строятся на основе модели «TO-BE». На этом этапе также используются модели в нотации BPMN. Кроме того, на данном этапе выполняется проектирование базы данных, которая является ядром ИС.

Для визуального проектирования баз данных для СУБД MySQL имеется средство MySQL Workbench, которое позволяет проектировать, моделировать, создавать и эксплуатировать БД. Средство ориентировано на построение ER (Entity-Relationship) моделей, для которых поддерживаются две наиболее распространенные нотации: IDEF1X (методология структурного анализа для проектирования сложных ИС) и Information Engineering (IE), которая используется преимущественно в промышленности. С помощью MySQL Workbench также можно генерировать таблицы и связи между ними на основании построенной ER модели (прямой инжиниринг), восстанавливать структуры уже существующей на сервере БД (обратный инжиниринг), создавать SQL запросы и выполнять их.

На этапе реализации помимо баз данных и SQL-запросов, создаются программные модули, выполняющие требуемые функции и пользовательский интерфейс. Использование языка PHP и библиотеки jQuery (JavaScript) [2] позволяет реализовать полнофункциональный графический интерфейс.

Также в тех случаях, когда ИС является частью портала, может использоваться CMS-система (Content Management System). Среди CMS-систем есть СПО, которые хорошо зарекомендовали себя в качестве инструмента поддержки содержимого сайта, например, Wordpress.

Литература

[1] *Мартишин С.А., Симонов В. Л., Храпченко М.В.* Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench, учебное пособие, М: ИД Форум — Инфра-М, 2012, 160 с. -ил.

[2] *Ленгсторф Дж.* PHP и jQuery для профессионалов М: Вильямс — 2011, 362 с.

Владимир Лукин, доцент, Лев Чернышов, профессор
Москва, Московский авиационный институт, Государственный университет Министерства финансов РФ
http://www.gumf.ru

О подготовке специалистов в области ПО

Ситуация, связанная с нехваткой ИТ-специалистов в экономике, похожа на катастрофу. 5 лет назад выпуск студентов составлял около 40% от потребности, а в 2012 году превышение спроса в 6 раз [1]. С качеством выпускников ситуация ещё хуже. Опрос, проведённый в 2012 г. среди работодателей Аналитическим центром Rabota.ru, показал, что 74% их них не удовлетворены качеством образования молодых специалистов [2].

Похоже, вузовская подготовка не приводит и не может привести к получению полноценной квалификации для программирования. Для улучшения положения в [3] предлагается связать с кафедрой предприятие по разработке ПО. Но в учебном процессе это сложно: (а) нет практикующих программистов, желающих передавать свои знания студентам, (б) поддержкой такого предприятия должен заниматься специальный сотрудник, которого обычно нет. В [4] предлагаются варианты с магистрами, дополнительным образованием, сочетанием учёбы с работой. Но первый вариант уже продемонстрировал свою неэффективность, а два других требуют много времени.

Положение с облачными приложениями ещё хуже. Встает вопрос их интеграции, но специалистов, которых готовят для работы в этой области, практически нет. Со свободно распространяемым ПО (СПО) проблема похожая.

Можно справедливо говорить о невысокой квалификации преподавателей, но решающая причина массового брака на выходе — это брак на входе. Обучить недорослей, не имеющих элементарную базу, невозможно. Вуз не может влиять на вход, но можно из пришедшего материала выбрать желающих учиться и уделять им повышенное внимание.

Чтобы работать со студентом дополнительно, есть единственный ресурс — время, которого у преподавателя нет. Отбросим вариант оплаты: во-первых, не каждый способный студент располагает средствами, во-вторых, нужно дать образование тем, кто хочет его получить. Откуда взять дополнительное время? Это могут быть, например, консультации. Но их мало, да и на дорогу надо потратить 2-3 часа. Проводить же консультации на дому неудобно.

Естественная возможность сэкономить время — общаться в сети. Этим пользуются многие преподаватели, но они тратят на обучение своё личное время. Назрела потребность это время легализовать. Конечно, контролёрам будет неудобно, но не это главное. Более важно — как проводить занятия, какими инструментами пользоваться.

Преподаватель не может платить деньги за лицензионные продукты, а краденными пользоваться не хочет. Альтернатива — использовать СПО.

Если предмет обучения — технологии разработки ПО, а СПО используется как инструмент, слушатель должен этот инструмент знать. Итак, СПО становится предметом изучения. Отсюда шаг до создания СПО, ибо это не что иное, как пример того ПО, производству которого обучают.

Эффективный процесс обучения программным дисциплинам требует активного взаимодействия, что приводит к формированию общего образовательного пространства, поддерживаемого облачными технологиями. Таким образом, в сферу изучаемых дисциплин втягивается и эта область.

Для того, чтобы сдвинуть с мёртвой точки ИТ-образование, необходимо, чтобы на уровень стандартов и, что более важно, на уровень реально действующих дисциплин, вышли как коммуникационные технологии, так и технологии облачных вычислений, наряду с СПО и принципами его создания.

Ассоциацией производителей коммуникационно-информационных технологий (АПКИТ) созданы стандарты по профессиям в области ИТ, которые описывают требования к уровням образования, стажу

работы и т.п. Комиссия по профессиональным стандартам Российского союза промышленников и предпринимателей рассмотрела проекты новых стандартов в области ИТ. Но ни в одном стандарте явно не обозначаются компетенции в области СПО или облачных технологий. Понятно, что существует традиционная инертность учебного процесса, но ситуация подходит к критической черте, и дальнейшее отставание в этой области недопустимо.

Рамки работы не позволяют подробно говорить о полезных и достаточно простых решениях, но кое-что отметить следует. Конечно, это электронная почта и интернет-сайты. Эти привычные средства при умелом применении могут привести к весьма неплохим результатам. Полезны общие ресурсы, такие, как Google-диск, особенно при работе над курсовыми или дипломными проектами. Далее, платформы для развёртывания сервисов для совместной работы. Из СПО полезно использовать не только привычный Open Office, но и специализированные средства, такие как Star UML.

К сожалению, отсидеться в надежде, что всё рассосётся, не получится. В материалах АПКИТ прогнозируется, что количество ИТ-студентов, будет падать с 300 тыс. в 2009 до 120–130 тыс. в 2015 году. Соответственно, упадёт и количество выпускников. Потребность в новых кадрах только в 2012 году составляет 12-61 тыс. человек. Хуже, что при нынешнем качестве образования только малая часть выпускников будет востребована. Отсюда, собственно, и катастрофа, которую следует избежать, используя все возможные средства.

Литература

[1] *Трудовая статистика* // Федеральная служба государственной статистики // `http://www.gks.ru`.

[2] *С. Рукшин.* Новый закон об образовании тянет страну на дно. `http://www.mr-msk.ru/story/2012/12/11/story_8828.html`

[3] *Терехов А.Н.* Технология программирования. — М.: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2006.

[4] *Лукин В.Н.* О подготовке специалистов, способных создавать программные системы. В кн.: Материалы IX Международной конференции NPNJ. М.: МАИ-ПРИНТ, 2012.

Михеев Андрей Геннадьевич
Москва, RunaWFE
http://wf.runa.ru/rus

Методика обучения процессному подходу к управлению предприятием на основе СПО и ее апробация в НИТУ МИСиС

Аннотация

В докладе рассказывается про курс обучения процессному подходу к управлению предприятием. Курс посвящен исполнимым бизнес-процессам (бизнес-процессам, непосредственно исполняющимся в компьютерной вреде), состоит из теоретической и практической частей. Работы практической части выполняются студентами на свободном ПО RunaWFE. Курс был апробирован в НИТУ МИСиС в весеннем и осеннем семестре 2012 г. на студентах магистратуры первого и второго года обучения. В докладе представлен опыт использования свободного ПО при обучении студентов.

Изменение подхода к процессной автоматизации

Особенностью традиционных работ в области процессного управления (например, — [1–7]) является то, что они не затрагивают автоматизацию исполнения бизнес-процессов. Использование компьютерных систем в традиционных работах ограничивается моделированием бизнес-процессов и изменением построенных моделей. То есть, в этих работах предполагается, что после разработки или изменения бизнес-процесса его внедрение в организации будет происходить без реального исполнения этого процесса на компьютере.

В последние годы в области процессного управления происходят качественные изменения. Степень оснащения современных предприятий компьютерной техникой позволяет создать всем работникам предприятия автоматизированные рабочие места, позволяющие взаимодействовать с системами, непосредственно исполняющими бизнес-процессы в компьютерной среде (см. [8 – 10]). Это позволяет исключить из действий сотрудников процедуры, связанных с поиском и передачей информации, что существенно повышает производительность труда.

Обучение студентов процессному подходу, основанному на исполнимых бизнес-процессах

Для обучения студентов процессной автоматизации был разработан практический курс процессного управления на основе исполнимых бизнес-процессов. Курс был апробирован в НИИТУ МИСиС в весеннем и осеннем семестрах 2012 г.

Курс ориентирован на студентов старших курсов, обучающихся по направлениям подготовки 080801 — «Прикладная информатика (в экономике)» и 230102 — «Автоматизированные системы обработки информации и управления». В курсе студенты знакомятся с базовыми понятиями процессного подхода, в частности с понятиями «определение бизнес-процесса», «экземпляр бизнес-процесса», исполнение экземпляра бизнес-процесса. Определение исполнимого бизнес-процесса излагается на основе идей С. Яблонского и С. Бусслера [11].

В практической части курса отрабатываются вопросы построения схем и инициализации ролей бизнес-процессов. Схемы строятся так, чтобы второстепенные задания не блокировали выполнение основных заданий бизнес-процессов (Рис. 1).

Также в курсе изучаются и закрепляются на практике вопросы работы с переменными бизнес-процессов, правилами маршрута движения точек управления, возможности задания сроков выполнения заданий. Разработанные бизнес-процессы студенты исполняют под разными ролями в программной среде.

Использование в практической части курса свободного ПО с открытым кодом

Основные занятия курса выложены в свободный доступ на сайте проекта RunaWFE в разделе документация по адресу `http://wf.runa.ru/rus` под свободной лицензией GNU FDL. Для разработки и исполнения бизнес-процессов используется свободный программный продукт с открытым исходным кодом RunaWFE (LGPL лицензия). Использование свободного ПО позволяет легко внедрить курс в учебный процесс любого российского ВУЗа. ПО бесплатно, для установки системы RunaWFE не требуется каких-либо ключей или лицензионных файлов. Установить ПО, а также выполнять и проверять с его помощью практические работы курса можно не только в учебных классе, но и на домашних компьютерах студентов и преподавате-

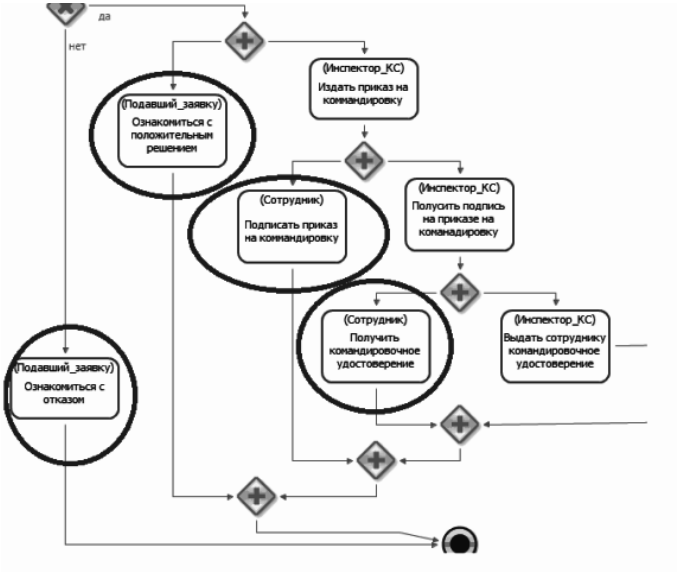


Рис. 1: (Михеев) Пример правильного расположения второстепенных узлов на схеме бизнес-процесса

лей. Количество инсталляций не ограничено. Разработанные бизнес-процессы можно свободно передавать в другие ВУЗ'ы без каких-либо затрат ВУЗ'ов на приобретение ПО.

Литература

[1] *Абдикеев Н.М., Данько Т.П., Ильдеменов С.В., Киселев А.Д.* Реинжиниринг бизнес-процессов. М.: Эксмо, 2005

[2] *Тельнов Ю.Ф.* Реинжиниринг бизнес-процессов: Компонентная методология. — М.: Финансы и статистика, 2004

[3] *Калянов Г.Н.* Моделирование, анализ, реорганизация и автоматизация бизнес-процессов. — М.: Финансы и статистика, 2006

[4] *Громов А., Каменнова М.С., Ферапонтов М., Шматалюк А.* Моделирование бизнеса. Методология ARIS. — М.: Весть-МетаТехнология, 2001

[5] *Репин В.В.* Бизнес-процессы компании. Определение. Анализ. Регламентация — М.: Стандарты и качество, 2007

[6] *Хаммер М., Чампи Д.* Реинжиниринг корпорации: манифест революции в бизнесе. — СПб.: Изд-во СПбУ, 1997

[7] *Кловпулос Т.* Необходимость Workflow. — М. Весть-МетаТехнология 2000

[8] *Вагнер Ю.* BPMS-эффект. Автоматизация в промышленности, 2009 №7

[9] *Куликов Г.Г., Михеев А.Г., Орлов М.В., Габбасов Р.К., Антонов Д.В.* Изучение методологии BPMN на примере программного продукта RunaWFE. Лабораторный практикум по дисциплине «Автоматизированные информационные системы в производстве» и «Автоматизированные информационные системы в экономике». — Уфа. УГАТУ. 2010

[10] *Михеев А.Г., Орлов М.В.* Система управления бизнес-процессами и административными регламентами. / Программные продукты и системы, № 3 2011

[11] *S. Jablonski and C. Bussler.* Workfow Management: Modeling Concepts, Architecture, and Implementation. International Thomson Computer Press, London, UK, 1996

[12] *Ссылка на сайт проекта RunaWFE:* `http://wf.runa.ru/`

А. А. Рябуша, С. А. Чунин
Санкт-Петербург
Институт Холода и Биотехнологий Национального Исследовательского Университета Информационных Технологий, Механики и Оптики.

Курсы переподготовки преподавателей ИХБТ НИУ ИТМО на базе Moodle

Аннотация

В период с октября по декабрь 2012 года в соответствии с приказом Минобрнауки РФ №330 от 24 апреля 2012 г. на базе ИДПО ИХБТ НИУ ИТМО проводились курсы профессиональной переподготовки педагогического состава по направлению «Применение системы дистанционного обучения "Moodle" в учебном процессе». Курсы проходили 30 слушателей, представлявших разные кафедры ИХБТ. В качестве обеспечения технической базы проведения курсов был развёрнут Moodle

2.2.3 на ALT Linux 6.0 Centaurus. В докладе так же будут разобраны технические подробности установки и настройки Moodle2 на ALT Linux 6.0 Centaurus, создание среды дистанционного обучения, освещен процесс подготовки материалов для проведения курсов и проанализированы результаты профессиональной переподготовки.

В период с октября по декабрь 2012 года в рамках приказа «Об организации повышения квалификации научно-педогогических работников института холода и биотехнологий в осеннем семестре 2012/2013 учебного года» в соответствии с приказом Минобрнауки России от 24.04.2012 N 330 «О контрольных цифрах приема слушателей в федеральные государственные образовательные учреждения высшего профессионального и дополнительного профессионального образования, подведомственные Министерству образования и науки Российской Федерации, для организации повышения квалификации научно-педагогических работников федеральных государственных образовательных учреждений высшего профессионального образования и государственных научных организаций, действующих в системе высшего и послевузовского профессионального образования, за счет средств федерального бюджета в 2012 году» на базе ИДПО ИХБТ НИУ ИТМО проводились курсы профессиональной переподготовки педагогического состава по направлению «Применение системы дистанционного обучения "Moodle" в учебном процессе».

Слушатели были разбиты на 3 группы. На занятия было выделено 72 часа. Списки групп были составлены по принципу удобства времени посещения, тем не менее слушателям разрешалось посещать занятия совместно с параллельными группами.

В качестве слушателей на курсах профессиональной переподготовки присутствовали преподаватели с разных кафедр, разных направлений и с разными базовыми навыками владения компьютером, среди них: ст.пр. — 6 чел., доц. — 17 чел., проф. — 5 чел., асс. — 1 чел., дек. — 2 чел., завкаф. — 2 чел, преп. ПЦ — 10 чел., преп. ОПЦ — 10 чел., преп. ООЦ — 8 чел.

Для проведения курса профессиональной переподготовки в локальной сети ИХБТ НИУ ИТМО был развернут Moodle 2.2.3 на базе ALT Linux 6.0 Centaurus.

Следующие компоненты были дополнительно установлены в Moodle с ресурса `moodle.org`:

- book (мультистраничный ресурс типа «книга»);

- WIRIS (математический визуальный редактор + онлайн платформа для математических расчетов);
- GeoGebra filter (фильтр, позволяющий включать в Moodle файлы GeoGebra и запоминающий последнее состояние файла);
- Jmol (фильтр, позволяющий включать в Moodle файлы, описывающие химические вещества, формата .pdb);
- mindmap (модуль, позволяющий создавать диаграммы связей в интерфейсе Moodle);
- Qestionnaire (модуль, позволяющий создавать анкеты);
- Moodle messenger (блок, отображающий последние уведомления и диалоги);
- Event reminders (автоматическая рассылка уведомлений о событиях календаря);
- темы оформления.

Следующие внешние инструменты были дополнительно рекомендованы преподавателям:

- GeoGebra (свободное мультиплатформенное динамическое математическое ПО, создающее интерактивные чертежи в планиметрии);
- QuickPDB, RasMol (ПО, предназначенное для визуализации структур молекул);
- `thinglink.com` (ресурс, позволяющий создавать интерактивные дейтаграммы);
- `wolframalpha.com`(база знаний и набор вычислительных алгоритмов);
- сервисы Google (диск, календарь, книги, академия);
- `mindmeister.com` (сервис для построения диаграмм связей);
- `slideshare.net` (сервис для создания, публикации и поиска презентаций);
- `screenr.com` (сервис для записи видео с экрана).

Предварительно для преподавателей был создан курс, содержащий в себе литературу по Moodle, методические рекомендации по использованию инструментов Moodle, ссылки на информационные ресурсы по Moodle и примеры использования Moodle в РФ. В дальнейшем курс

был дополнен встроенными примерами использования инструментов Moodle и наглядными материалами по использованию инструментов Moodle, в том числе переведенными Рябушей А.А. с английского языка.

В рамках курса было проведено 2 теоретических занятия, посвященных введению в e-learning и правовым аспектам использования СПО. Остальные занятия были практическими и были направлены на освоение инструментов Moodle для преподавателей, тем не менее форма проблемной дискуссии присутствовала. Достаточно часто вставал вопрос авторского права и охраны результатов интеллектуальной деятельности.

В итоге курс переподготовки успешно прошли все обучаемые, а 24 слушателя создали собственные электронные курсы в полном объеме предъявленных требований. Выявить какие-либо корреляции успешности освоения курса и понимания методической сути e-learning с педагогической направленностью, званием или уровнем навыков пользования ПК достаточно сложно, по этому в докладе будут рассмотрены отдельные яркие случаи.

Курсы переподготовки, кроме задачи обучения сотрудников ИХиБТ, организовывались для обеспечения информационного наполнения учебного сервера на основе Moodle2 с использованием ALT Linux 6.0 Centaurus и использования разработанных обучающимися материалов курсов в реальном учебном процессе. Решением Комиссии по информационным технологиям в обучении и управлении учебным процессом Учебно методического совета ИХиБТ, работы по созданию учебного сервера признаны удовлетворительными и авторам курсов рекомендовано использовать свои разработки в учебном процессе.

Иван Хахаев
Санкт-Петербург, Открытое акционерное общество
«Научно-исследовательский институт программных средств»

Свободные программы в проекте ГИС областного масштаба

Аннотация

Рассматривается предварительные результаты научно-исследовательской работы, проводимой Санкт-Петербургской академией ветеринарной медицины, по использованию свободного ПО в геоинформационной системе мониторинга и прогнозирования распространения болезней сельскохозяйственных животных.

Высшее учебное заведение «Санкт-Петербургская государственная академия ветеринарной медицины» (ФГОУ ВПО СПбГАВМ) проводит по госконтракту научно-исследовательскую работу по мониторингу распространения массовых заболеваний сельскохозяйственных животных (эпизоотий) в Ленинградской области. В рамках этой работы создается геоинформационная система (ГИС) для анализа текущей ситуации и прогнозирования возможных инцидентов. ОАО «НИИ ПС» является соисполнителем данной работы в части реализации ГИС и алгоритмов анализа, прогнозирования и моделирования.

В ТЗ на НИР предусмотрено использование свободного программного обеспечения для реализации ГИС. При этом запрещается публикация карт в Интернет, требуется создание централизованной базы для первичных данных, а также разграничение прав доступа к слоям ГИС и защита данных , передаваемых по сетям общего пользования (криптозащита). Кроме того, картографическая информация предоставляется не для всех пользователей системы. С учетом этих требований, а также целевой аудитории, было сформировано следующее решение:

1. Архитектура системы определена как комбинированная клиент-серверная: ввод первичной и оперативной информации в базу данных (БД), поддерживающую описания объектов с привязкой к географическим координатам осуществляется через web-интерфейс, в то время как визуализация значимых данных на картографической основе и анализ векторных слоев осуществляется средствами ГИС-приложения. Слои значимых данных формируются на основе таблиц БД.

2. Прикладным программным обеспечением являются PostgreSQL в качестве сервера баз данных с расширением PostGIS для поддержки типа данных «координаты», QuantumGIS как пользовательская ГИС, а web-интерфейс обеспечивается web-сервером Apache2 и сценариями на Python (фреймворк Django с расширением GeoDjango). В качестве картографической основы используются открытые карты из проекта OpenStreetMap.
3. Система построена на основе ОС семейства GNU/Linux. Серверная часть основана на Ubuntu Server 10.04 или ALT Linux 6.0, ГИС-приложения функционируют в окружении Ubuntu 10.04/12.04 или ALT Linux 6.0.
4. Инфраструктура криптозащиты основана на OpenSSL.
5. Система разделения доступа для web-интерфейса ввода и редактирования характеристик значимых объектов и оперативных данных обеспечивается встроенными средствами Django.
6. Система разделения доступа для пользователей PostgreSQL организуется как на уровне пользователей/подсетей/баз (в конфигурационом файле), так и на уровне таблиц, соответствующих слоям ГИС (с помощью ACL).

Средствами ГИС и дополнительных модулей (расширений) обеспечивается решение следующих задач анализа эпизоотической ситуации:

- формирование охранных (буферных) зон вокруг прогнозируемых точек вспышки заболевания;
- формирование буферных зон вдоль автомобильных и железнодорожных магистралей;
- подсчет количества ветеринарно значимых объектов, попадающих в буферные зоны, формирование списка этих объектов;
- подсчет поголовья в хозяйствах, находящихся в буферных зонах;
- визуализация экономических связей для выбранных ветеринарно значимых объектов;
- построение путей транспортировки животных и продуктов переработки между экономически связанными объектами.

Таким образом, вся система построена с использованием только свободного ПО, при этом обеспечивается необходимый уровень защиты

информации. Кроме того, данное решение может быть тиражировано на другие регионы как в виде комплекса «под ключ», так и в виде образов виртуальных машин.

Александр Прудников, Екатерина Гребцова
Санкт-Петербург, Санкт-Петербургский Национальный исследовательский университет Информационных технологий, Механики и Оптики.

Интеграция компонентов свободного программного обеспечения для пространственной визуализации состояния элементов территориально распределённых компьютерных сетей

Аннотация

Пространственная визуализация состояния объектов с координатной привязкой — активно развивающееся направление в информационных технологиях. В данной работе рассматривается возможность совместного использования компонентов свободного ПО для построения геоинформационной системы мониторинга состояния компьютерной сети.

В крупных сетях часто появляется необходимость мониторинга IT-инфраструктуры, чтобы иметь перед глазами полную картину происходящего в сети для отслеживания неисправностей. Если не иметь вспомогательного инструментария, задача отслеживания параметров состояния интересующих объектов может оказаться вовсе непосильной для отдельного человека или даже группы людей. На помощь в этих случаях приходят системы мониторинга и контроля состояния разнообразных сервисов компьютерной сети и её проблемах. Это, например, такие системы как Nagios, OpenNMS, Zabbix и другие.

Из существующих в наше время систем мониторинга сервисов компьютерных сетей в реальном времени наиболее предпочтительнее система Zabbix, так как позволяет одновременно отслеживать около 100 000 устройств. Ещё одним аргументов в пользу Zabbix является то, что она является наиболее функциональной среди существующих open source систем мониторинга. К тому же для мониторинга постоянно растущей и меняющейся компьютерной сети в Zabbix предусмотрена такая немаловажная возможность как автоматическое обнаружение новых устройств в сети.

Информация о компонентах сети, их характеристиках и параметрах хранится в базах данных (Zabbix поддерживает MySQL, PostgreSQL. IBM DB2, SQLite и Oracle). Получение данных, их обработка и анализ, а также запуск скриптов оповещения осуществляет сервер мониторинга. Сам мониторинг может быть осуществлен несколькими способами: проверка доступности основных сервисом, например, SMTP или HTTP (simple check); использование агента Zabbix, который устанавливается на отслеживаемых объектах и посылает необходимые данные о состоянии серверу; выполнение внешних программ (external check); мониторинг через SNMP.

Для представления данных производительности и доступности IT — инфраструктуры используются таблицы, которые можно просмотреть в веб-интерфейсе, написанном на PHP.

Если предполагается держать под постоянным контролем достаточно крупную и растущую IT — инфраструктуру, то получившиеся таблицы могут оказаться чересчур большими для визуального анализа, несмотря на то, что Zabbix подсвечивает проблемные узлы. Кроме того, предоставляемая Zabbix карта сети, описывающая логическую структуру сети, не позволяет с ходу определить реальное месторасположение того или иного узла. Поэтому можно рассмотреть задачу визуализации сети на точном плане для того, чтобы иметь наглядное представление о расположении узлов сети и взаимосвязи между ними.

Для решения этой задачи предполагается использовать геоинформационную систему. Это позволяет нанести на карту устройства, из которых состоит сеть, а также прорисовать связи между этими устройствами. Таким образом будет получено представление и географическом распределении сети.

Такая конфигурация (Zabbix в качестве системы мониторинга IT-инфраструктуры в сочетании с геоинформационной системой) предоставляет нам следующие возможности:

- кроме подсвечивания строк с информацией о проблемных узлах в таблицах Zabbix, данное устройство также отображается на картах как неисправное;

- можно прописать адреса устройств непосредственно на карте, что позволяет оперативно отправить бригаду работников для устранения неполадок к этому устройству;

- непосредственно на карте геоинформационной системы можно видеть состояние сети: загрузка каналов, неисправные устройства и другие возможные проблемы;
- появляется возможность визуального анализа сети на основе географического распределенности ее узлов для поиска оптимального места для добавления нового узла.

Так как система визуального отображения информации об объекте на карте должна использовать данные информационного состояния системы Zabbix предполагается использовать QuantumGIS. QuantumGIS выводит пространственные данные в пригодном для просмотра на экране или распечатки виде, а также поддерживает множество различных форматов данных, в том числе пространственные таблицы PostGIS и данные OpenStreetMap, которые и были выбраны для работы.

В качестве базы данных для хранения описания и адресов узлов сети выбрана PostgreSQL. Основным аргументом при выборе базы данных было то, что её поддерживают как Zabbix, так и QuantumGIS.

Все компоненты системы будут размещены на Ubuntu Server или другой операционной системе из семейства GNU/Linux. При необходимости есть возможность разместить отдельные компоненты на различных серверах.

Таким образом, данная система позволяет осуществлять мониторинг IT-инфраструктуры на основе различных параметров и отображение состояния сети на карте, подсвечивая в режиме реального времени неисправные компоненты сети, проблемы с загрузкой каналов и прочие данные. Вся система основана на свободном программном обеспечении при обеспечении высокой функциональности и гибкости.

Артём Капнинский, Алексей Кухтинов
г. Калуга, ЗАО «Калуга Астрал»
Проект: www.astralnalog.ru

Проблемы и решения подключения учреждений профобразования к ФИС ЕГЭ и приёма.

Аннотация

В данном докладе рассматриваются возможные варианты решений для учреждений профобразования для подключения к Федеральной Информационной Системе Единого Государственного Экзамена (далее ФИС ЕГЭ) и приёма. А так — же разъяснения для осуществления правильного выбора необходимой схемы подключения в целях экономии средств учреждений при эксплуатации системы в дальнейшем.

ФИС ЕГЭ и приёма — что это?

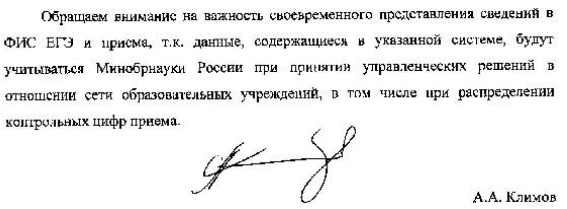
ФИС ЕГЭ и приёма — ФЕДЕРАЛЬНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА обеспечения ЕГЭ и приёма граждан в образовательные учреждения среднего профессионального образования и высшего профессионального образования, которая функционирует на базе защищённой корпоративной сети передачи данных Федерального государственного бюджетного учреждения «Федеральный центра тестирования» (далее ФЦТ).

Целевая модель ФИС ЕГЭ и приёма выглядит следующим образом:

Через Региональные центры обработки информации (РЦОИ) в ФИС ЕГЭ и приёма поступают данные ЕГЭ по выпускникам. ВУЗы и ССУЗы в свою очередь посредством запросов обмениваются данными с ФИС. Обмен данными происходит по защищённым каналам связи.

Построение системы и обмен данными в настоящее время является одной из ключевых задач Минобрнауки. Ниже приведена выдержка из письма Министерства «О предоставлении сведений ФИС ЕГЭ и приёма» от 29.08.2012 №АК 190/05 (см. рис. 1).

Так же Рособрнадзор с 01 февраля 2013 года планирует начать проверки крупных ВУЗов на предмет подключения их к ФИС ЕГЭ и приёма, с применением к нарушителям санкций, вплоть до отзыва лицензий у учреждений профессионального образования.



Обращаем внимание на важность своевременного представления сведений в ФИС ЕГЭ и приема, т.к. данные, содержащиеся в указанной системе, будут учитываться Минобрнауки России при принятии управленческих решений в отношении сети образовательных учреждений, в том числе при распределении контрольных цифр приема.

А.А. Климов

Рис. 1: Выдержка из письма Минобрнауки

В настоящее время, согласно техническим условиям подключения к защищённой корпоративной сети передачи данных (ЗКСПД) ФЦТ, согласованным с ФСТЭК (Федеральная служба по техническому и экспортному контролю) России разрешаются 3 схемы подключения, первые две из которых разработаны и предложены нашими специалистами.

Схема №1 предполагает подключение к ЗКСПД ФЦТ с использованием VipNet Terminal, стоящим на отдельно выделенном АРМ (Автоматизированное рабочее место), без аттестации сети учреждения. В финансовом выражении это порядка 51000 рублей.

Плюсы: Низкая стоимость и простота реализации.

Минусы: Невозможность интеграции с информационной системой подключаемой организации.

Схема №2 предполагает подключение к ЗКСПД ФЦТ с использованием защищённой сети VipNet учреждения. В таком случае сеть должна быть аттестована по классу безопасности К1. В финансовом выражении более затратный вариант, но в то же время, частично закрывающий потребность учреждения в исполнении требования ФЗ — 152. (Стоимость начинается от 272000 рублей, но конкретно можно сказать только после проведения начального обследования). На наш взгляд, самый оптимальный вариант.

Плюсы: Простота реализации. Автоматизированная обработка ПДн в ЛВС ВУЗа.

Минусы: Необходимость аттестации по более высокому классу К1.

Схема №3 подключения к ЗКСПД ФЦТ предполагает использования Программно — аппаратного комплекса (ПАК) «Шлюз ПДН». Требуется декларирование или аттестация сети по классу безопасно-

сти К3. Стоимость ПАК и работ по его внедрению в общей цифре начинается от 661000 рублей, а по срокам ввода в эксплуатацию от 24 до 66 дней.

Плюсы: Интеграция с любыми АС подключаемых организаций.

Минусы: Необоснованно высокая по отношению к схеме №1 и 2 стоимость реализации. Необходимость аттестации по К2 или К3

Встаёт вопрос, а какую схему необходимо применить в конкретном учреждении. В принципе, не углубляясь в детали, ответ прост: Если в учреждение во время работы приёмной комиссии обращаются до 500 абитуриентов, то ему подойдёт схема 1. Если же более 500, то схема 2 или 3.

Большинство образовательных учреждений, работающих с нами, выбирают схему 2 (ССУЗы как правило подключаются по схеме 1, ввиду малого количества абитуриентов и отсутствия финансирования).

А почему не схема 3? Ответы просты.

1. Новое, не проверенное устройство. Неизвестно, какие при работе с ним могут быть проблемы.
2. Необходимость ежегодного продления лицензии (не считая технического сопровождения).
3. Необходимость обновления типа лицензии при увеличении количества абитуриентов.
4. Необоснованно большая стоимость реализации.

Вы определились с выбранной схемой подключения и у Вас встал вопрос: «А что же дальше?». Дальше необходимо сделать несколько простых вещей.

1. Выделить сотрудника организации, ответственного за подключение к защищенной сети. Данный сотрудник ведёт взаимодействие с ФГБУ ФЦТ, а так же подписывает и согласует с ФГБУ ФЦТ документацию.
2. Подготовить и утвердить у руководителя организации схемы подключения.
3. Согласовать схему подключения с ФГБУ ФЦТ (форма 1, 2)
4. После согласования (положительного ответа от ФГБУ ФЦТ), провести закупку необходимого оборудования и/или программного обеспечения согласно требованиям, предъявляемым к подключаемой организации.

5. Получить в ФГБУ ФЦТ ключ шифрования VipNet, для подключения в ЗКСПД ФГБУ ФЦТ или произвести настройку межсетевого взаимодействия сетей VipNet.
6. Произвести пуско-наладку оборудования в соответствии с выбранной схемой. Утвержденный руководителем организации протокол приёмки направить в ФГБУ ФЦТ.

В результате Ваше учреждение снимает проблему предъявляемых требований и ряда вопросов со стороны контролирующих органов и регуляторов.

Обратившись в компетентную компанию, Вы сможете избежать многих ошибок и провести работы в кратчайшие сроки.

Очень надеемся, что приведённая информация будет полезна.

Литература

[1] *Брукс, Ф.,* Мифический человеко-месяц, или как создаются программные системы., `http://www.lib.ru/CTOTOR/BRUKS/mithsoftware.txt`

Зубов М.В, Пустыгин А.Н., Старцев Е.В.
Челябинск, Челябинский государственный университет

Выделение типов в универсальном классовом представлении для статического анализа исходного кода

Аннотация

Использование представления уровня классов не будет полным без анализа типов полей, возвращаемых из методов значений и аргументов вызовов. Получение типов для языков с динамической типизацией требует дополнительных затрат и специального подхода. Кроме того, дополнительно необходимо выделять связи агрегирования, порождаемые сложными типами данных: массивами, встроенными коллекциями и шаблонными типами.

Выполнение статического анализа с помощью универсальных промежуточных представлений (для нескольких языков) обычно выгоднее, чем с помощью частных (для одного языка). Чем выше уровень абстракции такого представления, тем точнее оно описывает несколько языков сразу, и тем проще выполнять на нем высокоуровневые

анализы. Для использования было выбрано универсальное классовое промежуточное представление[3].

Эффективность анализа универсального классового представления, не содержащего в себе типы полей и методов, достаточно ограничена. Сейчас выполняется визуализация представления в виде диаграммы классов без связей, а так же ищутся методы, которые можно вынести в суперкласс. Их введение позволит строить связи агрегации и композиции[2], которые несут в себе достаточно много информации. Кроме того, появляется возможность строить связи ассоциаций. Это всё придает классовому представлению уровень, максимально близкий к диаграмме классов UML[5]. Из такого представления можно получить гораздо больше информации, например, об используемых в коде шаблонах проектирования[4].

Задача выявления типов для языков со статической типизацией не представляет большой сложности, так как типы определены явно и их легко получать из AST[1]. Для языков с динамической типизацией ситуация совсем иная. Для Python был предложен подход на основе «утиной типизации». Название термина пошло от английского «*duck test*» («*тест на утку*»), который в оригинале звучит как: «*If it looks like a duck, swims like a duck and quacks like a duck, then it probably is a duck*»[6].

Именно такой подход выполняет интерпретатор Python при исполнении программы. В месте использования объекта класс, экземпляром которого он является может быть даже не виден в текущем пространстве имен. При статическом анализе подход «утиной» типизации будет выполняться «наоборот». При исполнении у класса проверяется наличие указанных полей и методов, а при анализе выбираются все использования класса и сравниваются с известными в исследуемом проекте «сигнатурами». Для этого на 1 проходе строятся эти «сигнатуры» полей и классов, а на втором проходе сигнатуры используемых полей и классов сравниваются с известными сигнатурами.

В листинге 1 приведен пример класса Zoo, который обладает 2 полями _rabbit и _duck, первое поле инициализируется в явном виде и, очевидно, объект какого класса с ним связан. О поле _duck мы можем судить лишь косвенно, потенциально функция get_duck из модуля dr_frankenstein может возвращать (возможно неочевидным способом) экземпляр класса CraftyDragon(коварный дракон), который помимо умения крякать(quack()) и плавать(swim()) будет обладать огненным дыханием.

Листинг 1. Пример демонстрирующий принцип «утиной» типизации

```
from my_aninals import MyRabbit
import dr_frankenstein

class Zoo(object):
    _rabbit = None
    _duck = None

    def __init__(self):
        self._rabbit = MyRabbit()
        self._duck = dr_frankenstein.get_duck()

    def life(self):
        while 1:
            self._rabbit.eat_carrot()
            self._duck.move_to_pool()
            self._duck.swim()
            if(self._duck.excitement):
                self._duck.quack()
```

Для проверки такого подхода было проведено исследование крупных Python-проектов: Django[8], Twisted[9] и Logilab[7]. В них количество успешно подобранных кандидатов для типа поля среди классов проекта составило *порядка 70%*, что достаточно велико, учитывая, что отбрасывались системные и библиотечные классы (внешние по отношению к проектам классы).

При формировании типов данных в универсальном промежуточном представлении учитываются только «*внутренние*» классы, являющиеся частью проекта. Системные, библиотечные и встроенные типы отбрасываются. Однако, это далеко не всегда правильно. Они могут содержать в себе коллекции элементов, являющихся классами проекта. Можно выделить группу таких элементов: массивы, встроенные в Python коллекции и шаблонные типы.

Исходя из того, что представленные типы содержат в себе множество элементов, встаёт необходимость в новой связи типа «*многие-к-одному*», так как простой случай использова-ния типа — это связь «*один-к-одному*». С точки зрения UML здесь идет речь об ассоциации с агрегированием и разной кратностью (рис. 1). Будем рассматривать все такие связи как агрегирование, без выделения композиции (когда

время жизни части равно времени жизни целого), так как на этапе статического анализа их невозможно разделить.

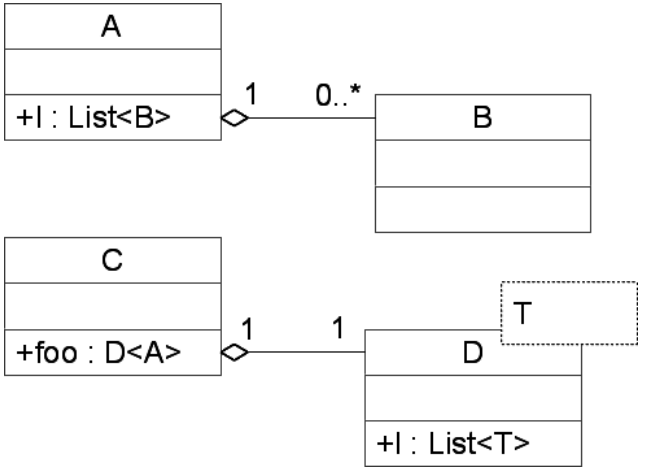


Рис. 1: (Пустыгин) Пример связей агрегирования с шаблонным типом.

Для Python необходимо рассмотреть встроенные в язык коллекции: списки, кортежи и словари. Они определяются по соответствующим узлам ASTNG[7]. Далее выполняется поиск использования найденной коллекции. Если найдено соответствие методам и полям коллекций, то тогда проверяются её элементы. Здесь идёт уже обычная «утиная» проверка.

Отдельно выделяется проблема шаблонных классов. Эта проблема актуальна для языков Java и C++(дополнение им находится в разработке). Здесь есть принципиально 2 различных момента: проектный класс находится под шаблоном и проектный класс содержит шаблон. В первом случае было решено считать эту связь как «*многие-к-одному*», потому что в большинстве случаев через шаблон определяются именно коллекции (рис. 2). Если всё же объемлющий тип не является коллекцией, то будет подразумеваться связь «*один-к-одному*» через него. Но эта связь лишь более строгий случай связи с коллекцией, поэтому здесь будет присутствовать неточность, но не ошибка.

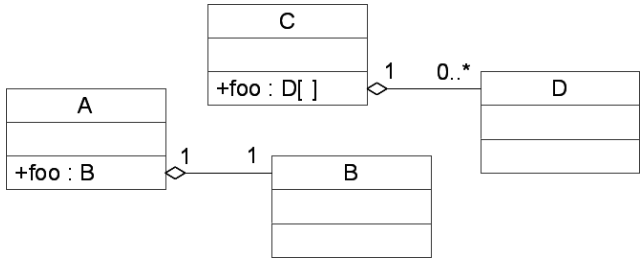


Рис. 2: (Пустыгин) Пример связей с агрегированием и разной кратностью.

Если же проектный класс содержит шаблон, то ситуация сводится к связи «*один-к-одному*» с этим классом. Его содержимое будет отражено в представлении, потому вводить тут дополнительных связей нет необходимости (рис. 2).

Таким образом, удалось расширить универсальное промежуточное представление уровня классов, выделив в нем 2 дополнительных узла, которые порождают 3 типа связей на диаграмме классов (ассоциация, агрегирование 1-1, агрегирование 1-0..*). Введение таких узлов позволит выполнять более сложные виды анализов, чем позволяет информация о наследовании классов.

Литература

[1] *А. Ахо, М. Лам, Р. Сети, Д. Ульман.* Компиляторы: принципы, технологии и инструментарий. — М.: Вильямс, 2010. - 1184 с.

[2] *Г. Буч, Д. Рамбо, А. Джекобсон.* Язык UML Руководство пользователя. — СпБ.: Питер, 2004. – 432 с.

[3] *Зубов М.В., Пустыгин А.Н., Старцев Е.В.* Подходы к статическому анализу открытого исходного кода. — Брест: Альтернатива, 2012. – 158 с. – с. 36–40.

[4] *Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес.* Приемы объектно-ориентированного проектирования. Паттерны проектирования. — СПб: Питер, 2007. - 366 с.

[5] *Диаграмма классов* — Википедия. `http://ru.wikipedia.org/wiki/`Диаграмма_классов

[6] *Утиная типизация* — Википедия. `http://ru.wikipedia.org/wiki/Утиная_типизация`

[7] *Logilab-astng (Python Abstract Syntax Tree New Generation)* (Logilab.org). `http://www.logilab.org/856`

[8] *The Web framework for perfectionists with deadlines* | Django. `https://www.djangoproject.com/`

[9] *Twisted.* `http://twistedmatrix.com/trac/`

Пустыгин А. Н., Ковалевский А. А., Огуречникова Е. А., Субботин А. А.
Челябинск, Челябинский государственный университет

Выделение типов в универсальном классовом представлении для статического анализа исходного кода

Аннотация

В докладе описывается прототип программного инструмента, предназначенного для получения эквивалентного представления открытого исходного текста на объектно-ориентированном языке программирования. Эквивалентные представления в виде диаграммы классов и диаграммы потока управления строится на основании промежуточного представления исходного текста. Использование универсального промежуточного представления позволяет использовать единый конвертор для получения диаграмм классов для текстов на разных исходных языках.

Целью описываемой работы было написание рабочего прототипа программного инструмента для получения эквивалентных представлений исходных текстов программ на языке C/C++. Для обеспечения возможности получения нескольких эквивалентных представлений — диаграммы классов, диаграммы потока управления, и возможности унификации части функциональных модулей для набора входных языков было реализовано универсальное промежуточное представления исходных текстов программ на языке C/C++.

Был выполнен обзор существующих программных инструментов, как свободных, так и проприетарных, используемых для статического анализа исходного текста, на основании которого сформулированы

требования к построению промежуточного представления и выбору инструментария разработки. Для детального сравнительного анализа были выбраны три программных инструмента: Dehydra, Treegydra, Clang. Каждый из этих инструментов позволяют получить промежуточное представление в форме абстрактного синтаксического дерева AST, которое является наиболее полным промежуточным представлением. Каждый из этих инструментов являются открытым проектом и достаточно документирован.

Был разработан, испытан и описан XML-формат промежуточного представления исходного текста, соответствующий предъявляемым требованиям. Далее был спроектирован и построен прототип конвертера промежуточного представления в эквивалентное, обеспечивающий построение диаграмма потока управления, связь полученной диаграммы с исходным текстом, возможность получения срезов потока управления по заданным параметрам фильтрация по идентификаторам, уровням и методам доступа, специальным условиям (рекурсивный вызов, неопределенная инициализация, циклическая синтаксическая конструкция), а также связь с другими эквивалентными представлениями, в первую очередь диаграммой классов. Диаграмма потока управления представлена в HTML, что позволяет свободно настраивать её внешний вид и расширять функционал

Прототип реализован на ОС Ubuntu 10.04, язык программирования C/C++, для генерации промежуточного представления используется библиотека libclang 3.2, входящая в состав проекта Clang[1], который разрабатывается как замена GCC, в частности, Google и Apple. Clang является фронт-эндом для языков программирования C, C++, Objective-C и Objective-C++(англ.), использующим для оптимизации и кодогенерации фреймворк LLVM и включен состав дистрибутива ОС FreeBSD.

Связь с исходным текстом настраивается на любой популярный текстовый редактор с помощью файла настроек (уже поддерживаются: Gedit, Geany, Vim, Nano)

Ниже приведены скриншоты главного окна графической оболочки инструмента и фрагмент эквивалентного представления диаграммы потока управления в HTML-формате.

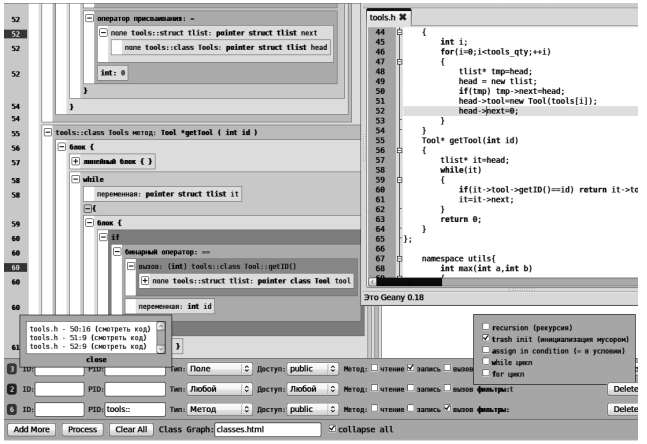


Рис. 1: (Пустыгин) Скриншот главного окна графической оболочки инструмента с открытым окном исходного текста проекта

Литература

[1] *Официальная страница проекта clang*: a C language family frontend for LLVM `http://clang.llvm.org/`

к.э.н., доцент Ключко Наталья Васильевна
Москва, Московский городской университет управления

Проблемы использования свободного программного обеспечения на практических занятиях в высшем учебном заведении

Использование свободного программного обеспечения в практической деятельности становится все более и более популярным. 17 декабря 2010 г. было выпущено Распоряжение правительства Российской Федерации №2299-р «О плане перехода федеральных органов исполнительной власти и федеральных бюджетных учреждений на использование свободного программного обеспечения (2011 — 2015

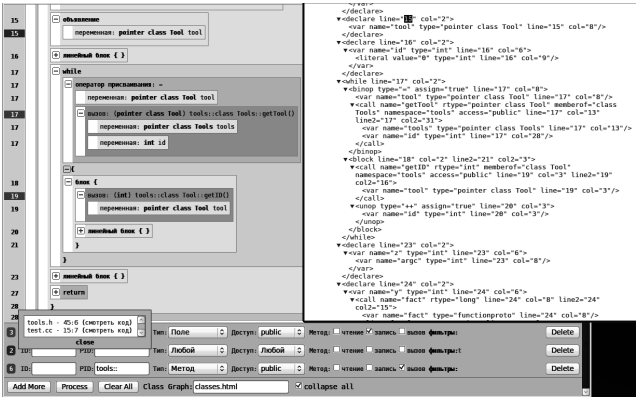


Рис. 2: (Пустыгин) Скриншот главного окна графической оболочки инструмента с открытым окном промежуточного представления текста проекта

годы)», согласно которому в 2013 году предусматривается внесение изменений в рекомендации о составе квалификационных требований к профессиональным знаниям и навыкам, необходимым для исполнения должностных обязанностей федеральными государственными гражданскими служащими, в области использования информационных технологий с учетом особенностей работы с обновленным пакетом базового свободного программного обеспечения и пакетами (обновленными пакетами) дополнительных прикладных программ, а также подготовка и утверждение методических рекомендаций для образовательных учреждений высшего профессионального образования о замене используемого в учебном процессе проприетарного программного обеспечения аналогичным свободным программным обеспечением[1].

Тем не менее, вплоть до настоящего времени свободное программное обеспечение широкой популярностью не пользуется ни в федеральных организациях, ни в ВУЗах. Тому есть ряд причин, основ-

[1]План перехода федеральных органов исполнительной власти и федеральных бюджетных учреждений на использование свободного программного обеспечения на 2011–2015 годы // Распоряжение Правительства Российской Федерации от 17 декабря 2010 г. № 2299.

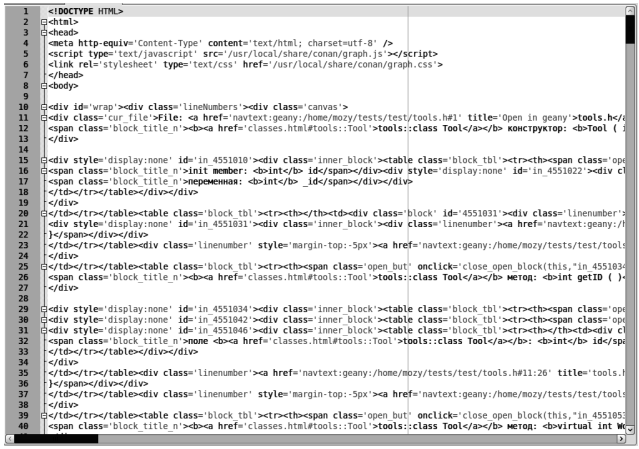


Рис. 3: (Пустыгин) Скриншот фрагмента эквивалентного представления диаграммы потока управления в HTML-формате

ной из которых, по нашему мнению, является профессиональная и агрессивная конкурентная маркетинговая деятельность корпорации Microsoft, которая со времени открытия своего представительства в России в 1992 году организовала широкое сотрудничество и с государственными организациями и высшими учебными заведениями. Достаточно сказать, что на в исследованиях и разработках, на которые Microsoft ежегодно тратит не менее 9 млрд. долл., принимают участие более 850 ученых и свыше 40 тысяч разработчиков по всему миру[2].

С 2002 по 2011 ежегодно Microsoft организовывала крупные конференции «Государство в XXI веке» с бесплатным участием государственых служащих.Начиная с 2006 года ежегодно проходят конференции «Образование в 21 веке» организованные Microsoft совместно с Министерством образования и науки Российской Федерации. С 1999 по 2010 гг. Microsoft ежегодно проводила в России технологические конференции «Платформа» по своим программным продуктам, в 2011–2012 гг. эти технологические конференции стали проходить под названием «Teched Russia».

[2]О Компании Microsoft [Электронный ресурс]. Режим доступа: `http://www.microsoft.com/ru-ru/news/inside_ms.aspx`

Но есть сила, которая уже сейчас вынуждает российские организации сокращать сотрудничество с этим мировым лидером в сфере программного обеспечения — это надвигающийся экономический кризис. 26 октября 2012 года Microsoft начала продажи новой ОС Windows 8 и начала сворачивать продажи Windows 7, которые только недавно начали устанавливаться в большинстве российских организаций. Нужно заметить, что ряд организаций работают еще на более ранних версиях Windows и офисных пакетов, для которых Microsoft уже не выпускает обновления.

И на этом фоне на первый план, как для государственного, так и частного использования начинает выдвигаться свободный офисный пакет OpenOffice.org, исходные тексты которого, после публикации компанией Sun Microsystems в 2000 г., стали доступны всем желающим. Формат файлов ODF (Open Document Format, или OpenDocument), который в 2006 г. был утвержден в качестве стандарта ИСО, в настоящее время является относительно «дешевым» форматом — большая часть поддерживающих его приложений (OpenOffice.org, Google Docs и др.) распространяется свободно или имеет бесплатные версии.

Начиная с OpenOffice.org версии 2.0, выпущенной в 2005 г., формат OpenDocument становится основным для этого офисного пакета, заменив свою предыдущую версию — OpenOffice.org XML. Важным достоинством формата ODF является то, что здесь представление документа изначально отделено от его содержания, что очень упрощает автоматизированную обработку, а также возможности прочтения и редактирования данных в этом формате в течение нескольких десятков лет. Одним из важнейших преимуществ для государственных организаций становится открытость формата и его независимость от поставщика.

В Московском городском университете управления в течение 2012 года в рамках курса «Информационные технологии» на практических занятиях бакалавры параллельно работали с приложениями Microsoft Office 2007 OpenOffice.org 3.2 под Windows. Это позволило выявить ряд проблем освоения работы в среде OpenOffice. Одной из важных проблем, по нашему мнению является сравнительная некомфортность интерфейса OpenOffice, относительно Microsoft Office 2007, напоминающего устаревшую версию Microsoft Office 2003. Но главной проблемой нужно конечно назвать недостаточную методическую проработку возможностей решения сложных задач вузовского уровня. Ката-

строфичной является ситуация с особенно актуальным для вузовского образования приложением OpenProj по управлению проектами, для которого невозможно найти практически никакой документации на русском языке.

В. В. Яковлев, Д. В. Хачко, А. Г. Кушниренко, М. А. Ройтберг
Москва, Пущино, НИИСИ РАН

Проект: Кумир `http://niisi.ru/kumir`, `http://gitorious.org/kumir2`

Кумир 2.0: компилятор и среда выполнения

Аннотация

Ранее была анонсирована разработка новой ветки системы «Кумир», где мы поставили перед собой цель создания с чистого листа расширяемой архитектуры программного комплекса, ориентированной на широкий круг задач — от начального обучения программированию до системы, применимой в школьных олимпиадах по программированию. Первый выпуск системы, основанной на новой архитектуре, запланирован на 24 января 2013 года и будет включать в себя две консольные программы: 1) компилятор языка Кумир в выполнимый байт код; 2) компактный интерпретатор байт кода для систем Linux и Windows, который может работать как на компьютерах с процессорами семейства x86, так и на некоторых устройствах с процессорами архитектуры ARM. Данный комплект предназначен, в первую очередь, для использования в системах автоматизированной проверки заданий (олимпиады и ЕГЭ).

Общие сведения

Система Кумир 2.x представляет собой набор модулей, исходные тексты которых находятся под открытой лицензией GNU GPL2 в ветке master репозитория [1]. Релизы системы выделяются в отдельные ветки, соответствующие номеру релиза, из которых исключаются те модули, которые не являются функционально завершенными.

На данный момент выделена ветка 2.0, которая будет включать два модуля — консольные программы: компилятор языка Кумир в выполнимый байт код, и интерпретатор байт кода для систем Linux, Windows и MacOS X.

Использование в системах автоматизированной проверки заданий

Приоритетность разработки набора консольных программ над разработкой пользовательского интерфейса определяется спецификой их области применения. В настоящее время проявляется интерес к использованию языка программирования Кумир в качестве олимпиадного, кроме того язык программирования системы Кумир (известный как «алгоритмический язык») является одним из языков программирования, применяемых при сдаче ЕГЭ и ГИА по информатике.

Для возможности использования в системах автоматизированной проверки заданий, набор консольных программ комплекса Кумир был разработан с учетом следующих требований:

- возможность работы как в Linux (в частности, совместно с ejudge [2]), так и в Windows (с использованием Contester [3]);
- система выполняется на сервере, поэтому должна быть реализована в виде консольных программ;
- скорость выполнения программ должна соответствовать уровню современных общеупотребительных интерпретаторов (таких как Python или Perl);
- порядок действий с исходной программой должен соответствовать общепринятой последовательности: скомпилировать исходный текст в программу, затем — выполнить её, указав потоки ввода и вывода данных, после чего — сравнить полученный результат с эталоном.

Возможность выполнения программ на широком спектре оборудования

Интерпретатор выполнимого байт кода системы Кумир реализован в виде одного выполняемого бинарного файла, который для своей работы требует только стандартную библиотеку языка C++. Размер релиз-сборки интерпретатора не превышает 400 Кб (для процессора ARM, система Raspbian Linux), при этом базовые возможности интерпретатора включают в себя полную поддержку всех возможностей языка Кумир, работу со стандартными функциями, с текстовыми файлами, поддержку кириллических символов Юникода и функ-

ции работы со строками. Сгенерированный компилятором байт код одинаково выполняется на процессорах различных архитектур.

Изначально предполагалась реализация минимально возможного интерпретатора для выполнения на устройстве Lego NXT 2.0, однако, для поддержки всех возможностей языка Кумир, в рамках эталонного учебника [4], это оказалось невозможным по двум причинам:

1. Отсутствие блока операций с плавающей точкой в используемом NXT процессоре [5], в то время как язык Кумир декларирует поддержку операций над вещественными числами в соответствии с IEEE 754.
2. Небольшой объем (64 Кб) оперативной памяти, что делает практически невозможным работу с таблицами языка Кумир.

Существующая на данный момент реализация системы Кумир 1.9 для работы с конструктором Lego NXT [7] предполагает использование управляющего хост-компьютера, который использует радиоканал Bluetooth для передачи телеметрических данных. Такой подход имеет два существенных недостатка: во-первых, требуется качественный Bluetooth-адаптер, во-вторых, это решение имеет большие временные задержки, оказывающие серьёзное влияние при реализации типовых алгоритмов управлением роботом. Данная проблема может быть решена только переносом выполняющей части на компьютер робота, и использованием хост-компьютера только для редактирования, загрузки и запуска программ.

Недавно компания Lego анонсировала [6] выход нового поколения компьютера серии Mindstorms, на который может быть портирован выполнитель системы Кумир без существенной адаптации. Поскольку данное устройство пока отсутствует в продаже, процесс портирования и отладки системы Кумир для процессора архитектуры ARM осуществляется с помощью устройства Raspberry Pi [8].

Отказ от использования сторонних библиотек (в частности, Qt) при реализации выполнителя, с одной стороны, усложнил процесс разработки, а с другой, – позволил снизить аппаратные требования для выполнения Кумир программ и обеспечить возможность портирования на те платформы, которые не поддерживаются разработчиками Qt. В частности, возможна реализация выполнителя для FreeDOS (после некоторой доработки стандартной библиотеки компилятора OpenWatcom) и на платформу Android (используя NDK). Существенным здесь является использование единых исходных текстов как для

«десктопной» версии Кумира, так и для версий, специализированных под конкретные устройства.

Литература

[1] http://gitorious.org/kumir2

[2] А. В. Чернов *Система тестирования ejudge.* Вторая конференция «Свободное программное обеспечение в высшей школе». М.: AltLinux, 2007.

[3] http://www.contester.ru/

[4] *Информатика: 7–9 кл.: Учеб. для общеобразоват. учр.* А. Г. Кушниренко, Г. В. Лебедев, Я. Н. Зайдельман. М.: Дрофа, 2003. 335 с.

[5] http://www.atmel.com/Images/6175s.pdf

[6] http://www.legoeducation.us/eng/product/ev3_intelligent_brick/2526

[7] А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, В. И. Хачко, Д. В. Хачко, В. В. Тарасова, В. В. Яковлев *Новые Миры в системе КуМир.* Пятая конференция «Свободное программное обеспечение в высшей школе». М.: Институт логики, 2010. с. 56–58.

[8] http://www.raspberrypi.org/

Игорь Воронин, Вероника Воронина
Шатура, ИПЛИТ РАН, СОШ 4 г. Павловов, Нижегородской обл

Проект: Образовательный проект — УМКИ http://umki.vinforika.ru/

Среда КУМИР для изучения алгоритмов управления сенсорными сетями роботов

Аннотация

В данной работе обсуждаются вопросы обучения на основе использования среды КУМИР базовым алгоритмам управления и дистанционной связи множества разнообразных устройств в единую сенсорную сеть. Приводятся примеры выполнения различных миссий , при помощи которых в игровой и увлекательной форме происходит обучение базовым навыкам программирования и управления различными роботизированными передвижными комплексами. Проект УМКИ основан

на СПО, использование в нем программной среды КУМИР увеличивает наглядность и повышает удобство в обучении программированию передвижных роботизированных платформ.

Как договориться с устройством, чтобы радиосигнал от одного пульта не заставлял сразу все машинки двигаться в одну сторону — вправо, например? Как добиться, чтобы затраты энергии на управление были бы минимально возможными, а сеть таких машинок просуществовала максимально долго? Как организовать взаимодействие между операторами машинок — если их много? Кто решает, кому и в какой момент времени подавать нужную команду, чтобы не было конфликтов?

Очевидно, что в ситуации где много управляемых элементов, люди могут очень легко запутаться. Поэтому нужно, чтобы команды управления по радиоканалу адресовались на конкретное устройство, точно в нужный момент времени. Даже если машинка находится вне зоны прямой видимости базовой станции, нужно уметь позиционировать, с достаточно высокой точностью местонахождение каждой, с тем, чтобы зная в каком месте обнаружен очаг задымления, другая платформа — с брандспойтом, могла подъехать точно в это место и потушить пожар.

Так вот, чтобы легко и просто — как бы играя, научиться управлять такими устройствами, был разработан образовательный конструктор — который называется: УМКИ. Это Управляемый по радио каналу Машинный Конструктор Инновационный. Достоинство конструктора в том, что он комплектуется роботизированными платформами, которые связываются между собой в единую сенсорную сеть, на основе протокола ZigBee. Изучив этот конструктор, ученики получают базовые знания по управлению сначала одним устройством, потом группой устройств объединенных в распределенную беспроводную сенсорную сеть. Каждая роботизированная платформа оснащается либо набором сенсоров — датчиков для различных физических величин, либо исполнительными механизмами. Вы сможете заставить машинку двигаться по программе, ориентироваться на местности и выполнять разнообразные задания. Курс составлен так, чтобы на каждом его этапе было максимально интересно получать знания. Выполняя шаг за шагом задания к занятиям — вы будете проходить разнообразные миссии. Занимаясь с УМКИ, в игровой форме вы овладеете серьезными инженерными знаниями которые сможете реализовать в дальнейшем, например в научных исследованиях.

Текущая версия УМКИ, базируется на элементах и различных инженерных схемах конструктора «Знаток». С ее помощью можно с интересом использовать конструктор. Она укомплектована методическими материалами для преподавателей, которые включают в себя поурочное планирование. Аудитория курса — это начальные классы дополнительного школьного образования. Документация по работе с роботизированными машинками УМКИ наполнена научно-популярной информацией для детей и представлена в игровой форме — как процесс изучения окружающего нас мира. Вдобавок, для большей наглядности в ней используются интерактивные технологии. Все миссии курса представлены в разных вариантах и связаны между собой логической составляющей.

Первым вариантом является «Миссия на Марс».

В ней ученикам необходимо реализовать миссию проводя как бы освоение Красной планеты. Первоначально собирается конструктор, при этом ученик получает базовые навыки по управлению роботом и попутно решает другие задачи (связанные со сборкой схем из технической документации). Цель миссии — создать колонию на Марсе.

Поскольку наборы конструкторов УМКИ, предполагается использовать в дополнительном образовании детей (чаще всего в форме кружков технического творчества), то вместе с изучением робототехники, предполагается коллективная работа нескольких детей и совместное решение задач, в форме стратегии. Таким образом, выполняя каждый этап миссии, дети собирают схему из конструктора, разбираются в принципе его работы, отвечают на вопросы по естествознанию (астрономии, физике, химии, биологии) и тем самым как бы увеличивают себе виртуальное жизненное пространство на Марсе.

К достоинству курса на базе УМКИ, можно отнести то, что это не отупляющая «стрелялка» или «бродилка», а интеллектуально развивающее пособие, которое позволяет через игровую форму, каждому ребенку, занимающемуся с конструктором УМКИ, под руководством преподавателя, или самостоятельно получить в увлекательной форме базовые знания и умения их использовать по основным естественным дисциплинам: физике, математике, механике, электротехнике. Кроме того, курс разработан таким образом, что преподаватель может общаться с учениками не только в очной форме, но и давать задания, и проверять результаты выполнения в форме дистанционного обучения, используя платформу MOODLE

В базовой комплектации, роботизированный комплекс — конструктор УМКИ состоит из:

1. 4-х колесного вездехода, или иначе говоря — передвижной платформы с модулем zigbee, который позволяет связываться множеству платформ по стандарту IEEE 802.15.4 в единую, распределенную самоорганизующуюся сенсорную сеть.
2. Радио шлюза — который по USB соединяется с ПК и служит для отправки команд по радиоканалу и приема ответов о выполненных процедурах.
3. Программного обеспечения (ПО) на ПК, для управления передвижной роботизированной платформой — одной конкретной — выбираемой по MAK адресу, или множеством сообщества.

Вместе с тем, конструктор УМКИ может быть до укомплектован набором различных датчиков — которые унифицированы таким образом, что они легко и просто подключаются к базовой платформе и становятся доступными для выполнения разнообразных миссий.

Программное обеспечение для управления вездеходом состоит из следующих модулей:

1. Серверной части, написанного на C++, и скомпилированной под GCC, которая загружается при запуске системы в оперативную память и находится там в постоянно активном состоянии — для формирования команд управления передвижной платформой, и приема ответов о выполненных командах.

2. Модуля внешнего интерфейса, написанного на QT и предназначенного для управления мышкой или с клавиатуры для управления передвижением поворотной платформы.

3. А так же свободно распространяемого пакета КУМИР — для программирования и задания пути движения передвижной платформы.

Код кумира

```
использовать Робот
алг
нач
. нц 3 раз
. . вниз
```

Рис. 1: (Воронин) Состав конструктора

```
. кц
. вправо
. вправо
. нц пока сверху стена
. . закрасить
. . вправо
. кц
кон
```

Следует также обратить внимание на то, что программный пакет КУМИР — рекомендован для использования в школах при подготовке к сдаче ЕГЭ

В дальнейших планах развития этого проекта — переход от обучения программированию перемещениями в плоскости — 2D, посредством использование алгоритмов, заложенных при разработке данного УМКИ, на управление объектами, перемещающимися в 3D пространстве, например беспилотными летательными аппаратами.

Литература

[1] *Хачко Д.В., Яковлев В.В, Субоч Н.М., Джелядин Т.Р., Ройтберг М.А., Кушниренко А.Г., Леонов А.Г.* Кумир 1.9. Практикумы и исполнители. Средства интенсификации обучения. Седьмая конференция «Свободное

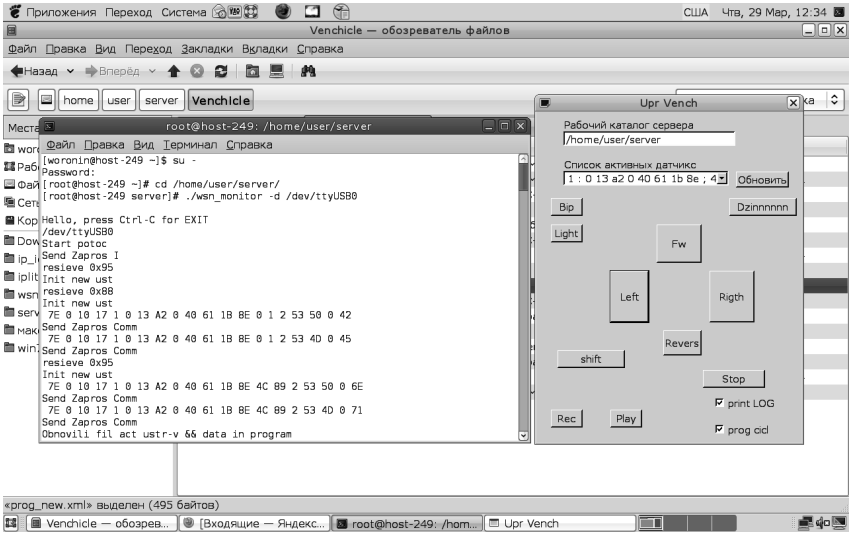


Рис. 2: (Воронин) Внешний вид монитора и программы управления

программное обеспечение в высшей школе». М.: Институт логики, 2012. с. 36.

[2] *А. Г. Кушниренко, Г. В. Лебедев, Я. Н. Зайдельман.* Информатика: 7–9 кл.: Учеб. для общеобразоват. учр. М.: Дрофа, 2003. 335 с.

[3] `http://www.niisi.ru/kumir/`

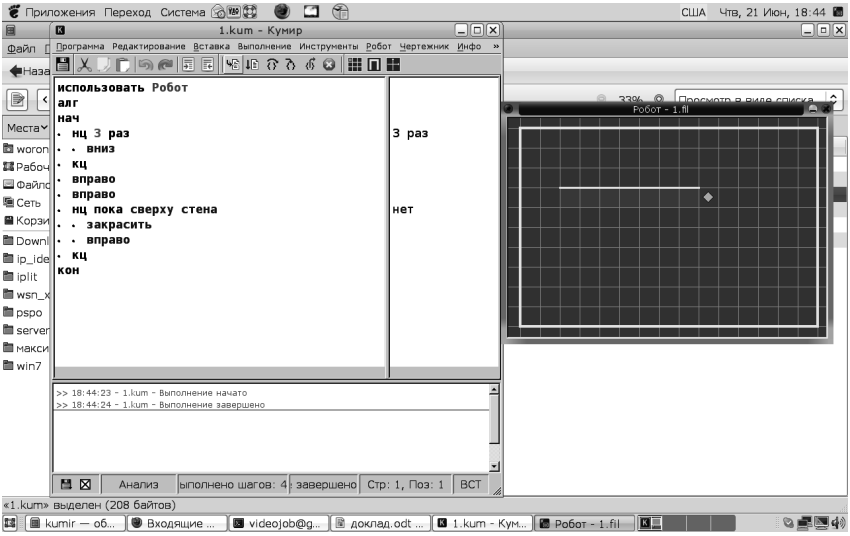


Рис. 3: (Воронин) Программа КУМИР

Валерий Руденко
г.Переславль-Залесский, Детский компьютерный центр при УГП

Свободное программное обеспечение для NXT

Аннотация
Использование NXT для изучения программирования.

1. Трудно переоценить значение выбора языка для обучения программированию. За 30-летнюю практику проведения таких занятий имеется опыт использования языков программирования от Фортрана и Паскаля до Лого и Пролога. При этом очевидно, что кроме языка необходимы компьютеры, надёжная реализация, разработанная концепция обучения, практическая методология используемой системы и т.д.

2. Будем исходить из простого принципа — программирование в любой форме это вид интеллектуальной деятельности, и для образования это своего рода современная «занимательная математика» (или

какая-то её часть). Отбросим миф о том, что таким образом (занимаясь программированием) можно обеспечить успех в современном мире, перенасыщенном компьютерными устройствами и интерфейсами. А если это и достигается, то тем, что занимаясь программированием, происходит развитие интеллектуальных способностей, необходимых для решения любых задач. Понимание алгоритма важнее средств выражения этого алгоритма.

3. Получается, что преподавание на Лого и Паскале осуществлять значительно проще, чем на C, C++, C# и на Java. Тем не менее иногда возникает ситуация, когда Лого и Паскаль недоступны. Имея несколько системных «серых блоков» (NXT Brick) возникло желание использовать этот перспективный аппаратный ресурс для обучения. Из того, что доступно для NXT имеется ряд программных систем: NXT-G, NXC/NBC, RobotC,...

4. Коммерческий визуальный язык NXT-G, основанный на LabView, и варианты C программирования создавали по разным причинам существенные сложности при обучении программированию на NXT. Запутавшись в очередном комплексе программ на C, попробовали систему LeJOS (NXJ). Эта система позволяет программировать для NXT с использованием JAVA технологии.

5. Кратко можно отметить некоторые особенности системы LeJOS:

- плагин для EclipseIDE,
- полна документации с уникальным описанием архитектуры NXT,
- доступ, инструкции и ПО для работы с репозитарием LeJOS.

Это делает проект LeJOS уникальным среди многих проектов свободного ПО.

Итоги.

Проект LeJOS использовался в течении года для занятий в детском компьютерном центре при УГП. Подходит для занятий и проектов с учащимися от начальной школы до студентов-старшекурсников и инженеров-исследователей. При этом можно изучать основы программирования на JAVA в распределённом режиме — один язык в проекте и на PC, и на NXT.

Тематика учебных и исследовательских проектов может быть довольно обширной и включать:

- знакомство с ООП — на серверах, планшетах и мобильных телефонах, других устройствах с поддержкой JVM,

- изучение на практике базовых концепций робототехники и искусственного интеллекта, с управлением различными моторами и датчиками,
- параллельное программирование и мультипроцессорные системы,
- коммуникационные проекты с использованием USB, Bluetooth, RS-485, I2c.

Евгений Чичкарев, Наталья Сидун
Мариуполь, Донецкая обл. Украина, Приазовский государственный технический университет
http://www.pstu.edu

Эффективность различных технологий распараллеливания при решении вычислительных задач

Аннотация

В работе исследовано влияние распараллеливания вычислений на время выполнения расчётов для различных типов задач и условий выполнения программ. Показана зависимость времени выполнения тестовых программ от вычислительной платформы (частота процессора, операционная система: Windows или Linux, используемый компилятор, версия OpenMP и др.).

Показано, что выигрыш по времени вычислений при работе кластера или вычислительного комплекса с многоядерным процессором под управлением ОС Linux по сравнению с ОС Windows достигает 10%. Проанализирована эффективность распраллеливания при использовании проприетарных (Intel Math Kernel Library, AMD Core Math Library) и открытых (Atlas) библиотек, а также систем компьютерной математики на их базе при работе на многоядерных процессорах. Установлены условия, влияющие на эффективность распараллеливания вычислений, предложены рациональные варианты параллельных алгоритмов решения ряда задач.

Выбор платформы для параллельных вычислений, включая архитектуру системы, операционную систему, языковые и программные средства в основном диктуется характером задач, техническими возможностями и порогом вхождения в работу для разработчиков,

а также такими материальными параметрами, как стоимость владения, включающая в себя стоимость на поддержку и модернизацию системы.

Основным требованием при выборе технологии и платформы для вычислений является ее производительность. Теоретически, реализации библиотек для параллельных вычислений как для систем с распределенной, так и для систем с разделяемой памятью, являются кроссплатформенными и не должны иметь различий в производительности. Однако, практическое сравнение производительности систем на базе различных операционных систем, а также с использованием разных компиляторов или версий библиотеки на одной и той же аппаратной базе и ОС показывают, что такие различия имеются и могут быть весьма существенными. Камнем преткновения в этом случае являются особенности реализации низкоуровневых процессов, таких как создание и уничтожение нитей выполнения, управление обменом сообщениями и синхронизацией потоков.

Традиционным является использование операционных систем на базе UNIX для организации параллельных вычислений, благодаря ее прозрачности и гибкости в настройке и администрировании. Рациональность такого выбора была исследована на примере систем с распределенной и разделяемой памятью. Было проведено сравнение производительности вычислений на одинаковых аппаратных платформах с использованием различных программных средств: ОС, компиляторов и т.д. В качестве вычислительной системы с распределенной памятью использовался вычислительный кластер следующей конфигурации: 7 вычислительных узлов
DualCore Intel Pentium, 1800 MHz cache 1 Mb;
RAM: 1Gb DDR2-667;
1 HDD 250Gb SATA 2
OC: Windows 2003 Server + MS Cluster Pack, или Ubuntu 11.10 64 bit
1 управляющий узел;
2 CPU Intel Core 2 Duo P8400, 2266 MHz cache 3 Mb;
RAM: 4Gb DDR2-667;
1 HDD 320Gb SATA 2,
Gigabit Ethernet для объединения вычислительных узлов, 100Mbit Fast Ehternet для связи кластера с внешним миром
коммуникационная среда — MPI(MPICH).

При проведении вычислительного эксперимента использовались два варианта вычислительных систем с разделяемой памятью: пер-

вый: AMD Athlon II X4 645, 4 Гб оперативной памяти DDR3; второй — Intel Core 2 Duo (2,2 ГГц) и 2Гб оперативной памяти. Использованное программное обеспечение — Windows 8 Release Preview Build 8400 64 бит или Ubuntu Linux 12.10 32 и 64 бит; компиляторы: gcc 4.7, g++ 4.7, MS Visual Studio 2010 C++ Compiler, Intel Parallel Studio XE C++ Compiler соответственно.

В качестве тестовых задач были использованы тесты LINPACK для систем с распределенной памятью и перемножение квадратных матриц произвольных чисел с плавающей запятой для системы с разделяемой памятью, а также разностные методы решения уравнений математической физики для обоих типов систем. В качестве меры производительности системы использовалось время выполнения программы (измерялось как общее время выполнения, так и время выполнения определенных участков кода).

Наиболее важным фактором при обеспечении стабильности и быстродействия являлся выбор и настройка а компилятора. Библиотеки для организации параллельных вычислений являются кроссплатформенными. Для систем с разделяемой памятью в качестве библиотеки для параллельных вычислений использовалась преимущественно OpenMP, которая манипулирует не потоками ОС, а облегченными вычислительными нитями. Теоретически, это должно было нивелировать различие в производительности, однако здесь на первый план выходит влияние компилятора на управление вычислительными нитями, а также, как и в случае с системой с распределенной памятью, с накладными затратами на обмен данными между потоками и синхронизацию.

Одни и те же тестовые задачи решались либо на Windows/Linux — кластере, либо с использованием многоядерных процессоров и технологий OpenMP или pthreads.

Кроме того, выполнялось сопоставление вариантов решения тестовых или реальных задач с использованием проприетарных библиотек (IMCL или ACML) и автонастраиваемой библиотеки Atlas. При тестировании с помощью комплектных бенчмарков библиотека Atlas продемонстрировала более высокую производительность при работе с функциями библиотеки LAPACK, прирост относительно проприетарной AMCL, оптимизированой для процессоров AMD особенно заметен для задач с матрицами большой размерности (более 200 элементов), разница может составлять от 15–20% до 100% и более. На задачах с матрицами меньшей размерности существенное премущество име-

ет проприетарная библиотека. При работе с функциями библиотеки BLAS ситуация несколько отличается. Производительность вычислений с помощью проприетарной ACML и Atlas в большинстве случаев сопоставима, однако сильно зависит от характерарешаемой задачи. Разница в производительности для большинства стандартных функций библиотеки не превысила 15%, при чем тенденция, согласно которой Atlas заметно проигрывает в производительности на задачах малой размерности сохранилась. Установлено, что в среднем различие скорости выполнения тестов (задачи линейной алгебры) не превышает 10–15% и зависит от характера решаемой задачи, размерности матриц, а также от тщательности настройки Atlas при сборке(в частности точность измерения времени выполнения вычислений).

Ряд задач решался с использованием систем компьютерной математики (проприетарный вариант — Matlab 2008; открытый вариант — Octave или SciPy/NumPy, пересобранные с использованием Atlas и с учётом многопоточности). Выигрыш производительности специально подготовленных программных средств по сравнению с проприетарными "из коробки"достигал 50–100%.

При сравнении вычислительных систем для различных типов задач было установлено, что системы под управлением UNIX при любом варианте распараллеливания показывают несколько большую производительность по сравнению с Windows–системами, а также характеризуются большей стабильностью. Ключевой для производительности системы особенностью со стороны ОС являлась общая загрузка, в частности количество потребляемой оперативной памяти при минимальной активности, а также специфика реализации управления процессами, потоками и нитями исполнения. Тестирование показало, что под управлением Windows именно более медленный механизм синхронизации вычислительных потоков непосредственно оказывает влияние на производительности системы в целом. Этот факт вместе с большими накладными расходами со стороны на создание потоков и вероятнее всего и становится определяющим для разницы в производительности.

Станислав Фомин
Москва, ИСПРАН, МФТИ
http://discopal.ispras.ru

SeminarAssembler — эффективная съемка, монтаж и публикация лекций и конференций

Аннотация

До сих пор почти во всех вузах доминирует догутенберговский формат «аудиторных лекций», с замученными лекторами и конспектирущими «не приходя в сознание» студентами. Казалось бы видеозапись лекций, семинаров, конференций — очевидное решение, может на порядки увеличить аудиторию, дав ей возможность посмотреть лекцию или доклад в удобное время, да и разгрузить от докладчика/лектора от повторов.

Однако, считается, что образовательную видеозапись можно сделать «дешево и плохо», либо, в самом лучшем случае, «дорого и долго». Лично мы, тоже столкнулись с всеми этими проблемами и при чтении курсов в МФТИ и ИСПРАН, и при организации множества IT/научных конференций, и при оптимизации процессов обучения и управления знаниями в коммерческой компании по разработке софта.

В результате, разработан open-source фреймворк SeminarAssembler для эффективной — дешевой и быстрой видеоконсервации произвольных лекций, семинаров, конференций, совещаний, с минимальными трудозатратами, и дешевым оборудованием.

Такой процесс можно внедрить в любом ВУЗе, даже страдающем от хронического недофинансирования, требуется лишь немного воли и времени на изучение.

Современная ситуация такова, что повсеместно распространен широкополосый интернет, у всех минимально образованных людей есть ноутбук или планшет, востребовано и становится популярным электронное дистанционное обучение (Coursera и т. п.). Но с другой стороны аудиторий российских вузов меняется мало — почти все курсы продолжают преподаваться в средневековой, догутенберговской манере — лекции по расписанию, с девизом «страдают все!».

Преподаватели пытаются, зачастую в неподготовленных аудиториях, втиснуть свой рассказ в академический слот ориентируясь на «среднего студента». Продвинутые студенты скучают, отстающие — отпадают, да и в целом, из-за отсутствия электронных материалов,

все сконцентрированы на конспектировании в режиме «не приходя в сознание». Пропускающие лекции студенты вовсе в пролете, и таких становится все больше — ранняя карьера, и честно говоря, трезвая оценка эффективности затрат личного времени. И весь этот средневековый сизифов труд повторяется из года в год для штатных преподавателей, чувствующих себя «роботами-плеерами», ежегодно повторяющими стандартный набор, без шансов углубить или расширить объем. Либо, если это были редкие лекции приглашенных специалистов, все это уходит в небытие, кроме смутных воспоминаний десятка бывших студентов. И кстати, аналогичная проблема с научными и образовательными конференциями — «напряжно и неэффективно».

Казалось бы, что мешает сделать долгоиграющие электронные курсы? Книги? Тесты? Разгрузить себя от работой «магнитофоном» и сконцентрироваться на семинарах, играх, симуляциях, проектах, и прочих интерактивных формах, требующих личного участия?

Увы, тут, как говорится, «есть нюансы». Издание «обычной книги» — дико трудоемко. И даже если выкинуть трудозатраты «бумажного издания» — возню с корректорами, издательствами, верстку под неудобный формат бумаги и т. п., и публиковать обновляемый электронный курс (в HTML/PDF) — это все равно долго и тяжело, а для преподавателей «старой школы», не освоивших текстовые процессоры или издательские системы, и вовсе невозможно. Плюс теряется куча возможностей «лекционного формата» — объяснение сложного простыми словами (в печатной форме это, увы, до сих пор не принято), демонстрация живьем физического опыта или компьютерной симуляции, да и в целом, очень эффективна идея правильных лекций: «визуально показать самое важное — ключевые слова, формулы, схемы/диаграммы и ключевую визуализацию» и завернуть это все в простую и понятную речь, чтобы все это, синхронно укладывалось в голову студента параллельно, по видео- и аудио-каналу. Лично я был дико удивлен, когда составлял стенограммы докладов, и понял, что получасовой рассказ нормального лектора — это 60 тыс. знаков, записать который — огромная работа даже для меня, отлично владеющего слепой печатью.

Видеолекции? Тут проблема в том, что мало кто умеет грамотно снимать, монтировать и публиковать. Проблемы низкого качества любительских записей на бытовые камеры/фото/телефоны понятны, но даже сьемка «профессионалом-оператором» на дорогое оборудование обычно бессмысленна — все навыки операторов и монтажеров,

полученные при обучении и теле- видео- кино- сьемках — «развлекательное» чередование планов и т. п. — бесполезны и даже вредны для лекций и конференций. Невозможно смотреть видео лекции, где крупный план на докладчике, а доску или слайд с формулами, о которых он рассказывает, показывают мимоходом, а не параллельно с его рассказом!

Нужно именно параллельное и постоянное присутствие «в кадре» всех активных информационных источников — слайдов демонстрации, доски с формулами, жестикулирующего докладчика, и может быть даже видео аудитории/зала — чтобы видеть реакцию, вопросы и ответы.

Но с такими требованиями, все это начинает выглядеть дико сложно. Нужны куча техники, «профессионалы», сложный видеомонтаж, в результате чего, пропадает общая экономия на лекторе — «так что, пусть рассказывает как обычно, он уже привык». Да и даже профессионалы сложный монтаж делают очень долго, в результате те, кто пропустил лекции, не могут в кратчайший срок просмотреть пропущенную лекцию.

Собственно, мы, столкнулись с всеми этими проблемами и при чтении курсов в МФТИ и ИСПРАН, и при организации множества IT/научных конференций, и при оптимизации процессов обучения и управления знаниями в коммерческой компании по разработке софта.

И мы их решили.

Разработан open-source фреймворк (открытые процесс и технология, и в частности — набор open-source инструментов) SeminarAssembler (см. `http://wiki.4intra.net/SeminarAssembler`, `http://wiki.4intra.net/Efficient-eduvideo-course`) для эффективной — дешевой и быстрой видеоконсервации произвольных лекций, семинаров, конференций, кстати совещаний.

Дешевой:

- Можно обойтись простыми любительскими miniDV-камерами, стоимостью от $40, или простыми фотоаппаратами ($100).
- Бесплатный софт (`http://wiki.4intra.net/`SeminarAssembler, `http://wiki.4intra.net/`MiniDV2AVI, `http://wiki.4intra.net/`ConferenceRecorder ...)
- В качестве операторов/монтажеров отлично подойдут сами студенты, после миниобучения. Оператором может быть студент-

участник, после пятиминутного инструктажа, и это не требует отрыва от лекции.

Быстрой:

- Можно публиковать на следующий день!
- При этом, активные трудозатраты на обработку — от нескольких минут (в случае простого монтажа в «матрешку»), в режиме «не приходя в сознание».
- Легко вносить доработки или исправления — автоматизированная система пересборки.
- Можно делать гипертекстовую видео базу знаний — ссылаться на конкретные моменты лекций, коллаборативно, без видеомонтажа составлять «вырезки» из старых (и возможно, частично устаревших материалов).

С помощью этой технологии было снято и смонтировано несколько тысяч образовательных видео (конференции, лекции, семинары, ...), некоторые из которых можно увидеть по ссылкам:

- `http://lib.custis.ru/It-talks`
- `http://discopal.ispras.ru/Video-lectures`
- `http://wiki.4intra.net/Category:OSDN-UA-2012`
- `http://lib.custis.ru/Seminars`
- `https://vimeo.com/channels/51004`
- `https://vimeo.com/channels/251742`

Т.е. такой процесс можно внедрить в любом ВУЗе, даже страдающем от хронического недофинансирования, требуется лишь немного воли и времени на изучение. При этом можно добится максимального масштабирования — софт бесплатен и открыт, можно ставить на множество компьютеров, оборудование дешево, персоналом смогут стать даже студенты-первокурсники, с элементарной IT-грамотностью.

Собственно, уже сейчас можно все скачать, поставить, и даже пройти дистанционное обучение технологии (См. Курс «Видео на конвеере», `http://wiki.4intra.net/Efficient-eduvideo-course`).

Дмитрий Костюк
Брест, Брестский государственный технический университет

Особенности использования виртуализованных окружений, внедренных в презентационные материалы

Аннотация

Рассматривается подход к использованию виртуализации для повышения наглядности и интерактивности учебных материалов за счет непосредственного встраивания окон виртуальных машин в слайды презентаций либо электронные учебные пособия.

Медийная насыщенность демонстрационных материалов и электронной документации обычно ограничивается копиями экранов и, возможно, вставками анимационных фрагментов. «Живая» демонстрация обеспечивается частым переключением между окном, отображающим слайды или страницы документа, и окнами демонстрируемых программ. При этом для более простого развертывания демонстрируемое ПО может помещаться в контейнер виртуальной машины (ВМ). Импортируемое виртуализованное окружение может содержать любую готовую конфигурацию системного и прикладного ПО, а механизм снимков (snapshots) позволяет быстро выполнить откат ВМ к нужному моменту работы для повторной демонстрации ключевых элементов либо пропуска длительных процедур.

Однако установленное ПО, в отличие от копий экрана, не составляет единое целое с сопроводительными и поясняющими материалами. Частое переключение между окнами ВМ и презентационной программы ухудшает непрерывность восприятия информации во время лекции, а если ВМ является приложением к электронной документации или учебному пособию, лишние действия приходится выполнять конечному пользователю.

В плане вычислительной мощности современных ноутбуков и десктопов замена копий экрана программы «живым» выводом изображения ВМ, интегрированным непосредственно в поясняющие материалы — вполне осуществимая задача. Современные ВМ способны работать в «невидимом» (headless) режиме, предоставляя вместо графического окна программу-сервер для удаленного доступа (например, по

протоколу VNC). Клиентское же ПО является сравнительно несложным, а некоторые клиенты написаны на скриптовых языках либо в виде легко встраиваемых аплетов.

Задача создания подобных интегрированных виртуальных окружений решалась нами в рамках подготовки учебно-демонстрационных материалов по истории графического интерфейса. Изначально созданное решение было использовано в качестве лекционного материала, а затем адаптировано к роли действующей электронной экспозиции (выбранная тематическая направленность делает интерактивность демонстрируемого объекта особенно выигрышной).

Анализ исходной задачи показал, что возможность внедрить вывод окна внешнего приложения в документ на платформе GNU/Linux в свободных пакетах для подготовки и показа презентаций недоступна без создания модулей расширения. В результате было принято решение в пользу презентационных фреймворков на базе HTML5. Рост популярности этого формата для показа презентаций, наблюдающийся в последние годы, вызван богатыми возможностями языка для создания визуальных эффектов и простотой редактирования контента, при использовании готового фреймворка, отвечающего за оформление, показ и смену слайдов. На текущий момент в свободном доступе находится более десяти готовых решений, различающихся богатством визуальных эффектов и стилей оформления, а также возможностями сложной компоновки материала.

В результате были выбраны следующие компоненты:

- виртуальные машины KVM либо QEMU с аппаратной поддержкой виртуализации;
- VNC-клиент noVNC, написанный на javascript и HTML5 (`http://kanaka.github.com/noVNC`);
- фреймворк reveal.js для показа слайдов (`http://github.com/hakimel/reveal.js`).

Как и другие подобные фреймворки, reveal.js обладает обширным функционалом, включая показ встроенных слайдов, упрощенную разметку содержимого, возможность экспорта в PDF. Для более эффектного отображения требуется браузер с поддержкой трехмерных CSS-преобразований, однако предусмотрен и упрощенный режим показа. Подготовка слайдов выполняется на HTML, в разметке markdown, либо в визуальном онлайновом редакторе, доступном по адресу `http://www.rvl.io`.

Возможность включения HTML-фреймов в содержимое слайдов была использована для вставки страниц, отображающих на элементе canvas передаваемое ВМ изображение. KVM и QEMU не поддерживают веб-сокеты, поэтому трафик передавался VNC-клиенту через прокси websockify, изначально разработанный в рамках проекта noVNC для работы с серверами, поддерживающими лишь TCP-соединение.

Выбор средств виртуализации продиктован желанием использовать полностью свободный набор ПО, возможностью эффективной работы большого числа ВМ, а также широким спектром платформ, поддерживаемых проектом QEMU. Эксперименты показали, что достаточна производительность среднего процессора для ноутбуков либо десктопов, имеющего аппаратную поддержку виртуализации (отчасти низкие требования вызваны тем, что в каждый момент предполагается активное использование только виртуальной машины, отображаемой на текущем слайде). Требуемый объем ОЗУ более критичен и определяется нуждами конкретных гостевых ОС. Однако при большом числе однотипных гостевых систем он может быть уменьшен использованием KVM в связке с технологией Kernel Samepage Merging (KSM), позволяющей объединять одинаковые страницы памяти для различных приложений (одно из типовых применений — как раз сервера виртуальных машин). KSM существенно снижает потребление памяти на системах, на которых затруднительным является одновременный запуск всех ВМ, необходимых для демонстрации, однако требует их предварительного запуска, и возможно, не в один этап (т. к. дедупликация страниц выполняется в фоновом режиме). Таким образом, этот способ снижения системных требований более удобен для постоянно действующей экспозиции, чем для лекционных материалов.

Запуск всей системы, включая нужные ВМ, прокси и браузер, выполняется с помощью тривиального управляющего скрипта (рис. 1).

В целом, получаемое презентационное окружение, несмотря на определенную ресурсоемкость, оказывается вполне работоспособным на типичном оборудовании, а также обладает рядом преимуществ: добавляет в материалы необходимую интерактивность и при этом не требует от пользователя или лектора переключения программ, фрагментирующего процесс преподнесения и усвоения материала.

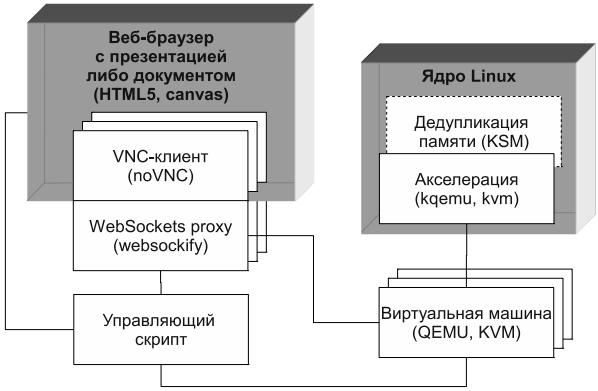


Рис. 1: Структура системы

Михаил Быков
Москва, diglossa.ru
http://diglossa.ru/morpheus

Morpheus — морфологический плагин к браузеру FireFox

Аннотация

Morpheus — плагин к браузеру ФайрФокс, обращаюшийся в режиме ajax к морфологическому сервису с тем же именем на ресурсе diglossa.ru. Позволяет получить морфологический анализ латинского слова. (Древнегреческий в процессе доработки и тестирования,). Доступен по адресу https://addons.mozilla.org/ru/firefox/addon/diglossa/

Morpheus-плагин к браузеру FairFox — плагин, построенный по образцу известного плагина Dict к сайту dict.org, с теми же возможностями. Но само наличие Морфей-плагина повлекло за собой перестройку и Морфей-сервиса. Если раньше Морфей определял конечное и небольшое количество словоформ, а именно только те словоформы, которые находились на страницах сайта http://diglossa.ru (85 тыс. словоформ), то теперь возникла необходимость отвечать на любой произвольный запрос. Это повлекло за собой изменение архитектуры приложения, и сейчас анализатор Морфей, его серверная

часть находится в состоянии перестройки под эту задачу. Несмотря на это, в настоящее время Морфей обрабатывает уже около 800 тыс словоформ, т.е на порядок больше прежнего.

Евгений Кондратьев, Сергей Оплетаев
Москва, Институт математики и информатики Московского городского педагогического университета

Метод Эйлера в Calc

Метод Эйлера — наиболее простой численный метод решения обыкновенных дифференциальных уравнений с начальным условием (задачи Коши). Этот метод был впервые описан Леонардом Эйлером в 1768 г. в работе «Интегральное исчисление». Метод Эйлера является явным, одношаговым методом первого порядка точности и основан на аппроксимации интегральной кривой кусочно-линейной функцией, ломанной Эйлера (рис. 1).

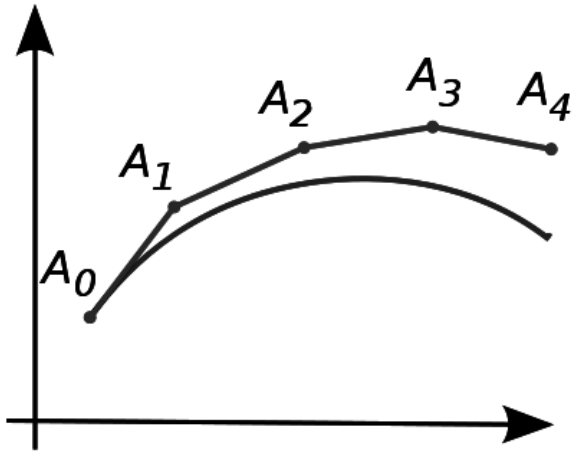


Рис. 1: (Кондратьев) Ломанная Эйлера (верхняя линия) — приближенное решение в пяти узлах задачи Коши и точное решение этой задачи (нижняя линия)

Решение задачи Коши на компьютере методом Эйлера требует её программирования и выполнения вычислений по разработанной программе. Расчет по методу Эйлера ведется на ряде последовательных равноотстоящих шагов, и при алгоритмической реализации расчета обычно используется цикл с конечным числом шагов. Например, для следующей задачи Коши:

$$\begin{cases} y' & = 2ty, \\ y(0) & = 1 \end{cases}$$

Блок-схема реализации метода Эйлера на компьютере показана на рис. 2.

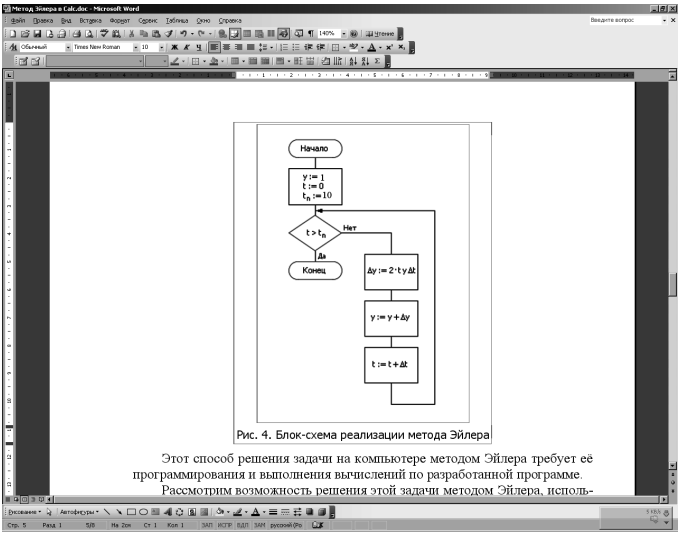


Рис. 2: (Кондратьев) Блок-схема реализации метода Эйлера

Далее выполняется программирование задачи, и для получения числовых значений осуществляются расчеты по разработанной программе. Решение задачи Коши методом Эйлера этим путем требует написания отдельной программы для каждой задачи Коши и умения программировать.

Предложено реализовать метод Эйлера в электронной таблице Calc. Электронная таблица Calc являются составной частью сво-

бодного программного обеспечения (СПО) Apache OpenOffice.org и LibreOffice. LibreOffice входит в последние дистрибутивы Linux Ubuntu и Linux Edu Mandriva.

Для демонстрации этого метода было выполнено решение рассмотренного примера задачи Коши (1) в электронной таблице Calc русифицированной версии СПО Apache OpenOffice.org. 3.4.1 (рис. 3–5).

Разработанный метод позволяет получать числовые результаты решения любой задачи Коши методом Эйлера и графически представлять их без программирования задачи и без использования коммерческих электронных таблиц.

Рис. 3: (Кондратьев) Стартовое окно Apache OpenOffice.org

А.В Апанасевич, В.Н. Лукин
Москва, Московский авиационный институт

Мобильный клиент оперативного доступа к ИАСУ МАИ

Аннотация

Предлагается новый подход сбора данных о результатах контроля знаний. Мобильный клиент предоставляет прямой доступ к информационной системе для преподавателей. Данный клиент упрощает работу преподавателя, позволяет полностью избавится от бумажного документооборота при проведении контроля знаний.

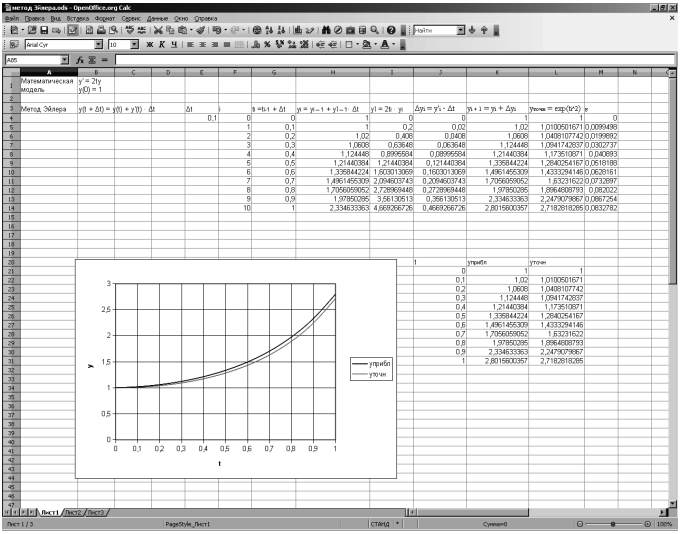


Рис. 4: (Кондратьев) Реализация метода Эйлера в OpenOffice.org Calc при $\Delta t = 0.1$

Цель разработки данного клиента — упрощение работы преподавателя, а также автоматизация работы деканата. Существующее сейчас решение связано с ручным вводом данных результатов контроля знаний студентов. Данные поступают в деканат в виде первичных документов на бумажных носителях. После обработки в системе создаются их электронные образы. Задача — убрать существующее промежуточное звено и разработать мобильный клиент, который позволит вводить данные напрямую в систему ИАСУ МАИ.

Мобильный клиент представляет собой компьютерный аналог бумажной ведомости. Данное решение предоставляет следующие преимущества:

- Автоматическое формирование ведомости для проведения экзамена.
- Автоматическая отсылка в систему ИАСУ МАИ заполненной закрытой ведомости, чтобы к ней был доступ у сотрудников деканата.

Рис. 5: (Кондратьев) Реализация метода Эйлера в OpenOffice.org Calc при $\Delta t = 0.05$

- Автоматическое формирование дополнительной ведомости.

При проектировании был проведен предварительный анализ известных архитектур клиент- серверного взаимодействия и выбран вариант реализации в виде WEB- ориентированного клиента. Web-клиент, в отличие “толстого” и “тонкого” клиентов, функционирует в среде браузера, тем самым достигается необходимы уровень универсальности: пользователь может работать с системой через свои привычные средства интернет- навигации. Данный подход позволяет упростить администрирование: не требуется установка клиента на каждом рабочем месте, отсутствует необходимость обновления ПО для всех рабочих мест — в любой момент времени пользователи работают с самой последней версией клиента.

Существенной проблемой является обмен данными между модулями системы. Задача импорта/экспорта данных решается применением XML-файлов. Для описания структуры документа используется язык

XSD. Данный подход упрощает создание объектов в памяти, упрощает разбор XML-документа.

Важный момент — вопрос безопасности. Данный клиент работает через сеть интранет. Применяется дайджест аутентификация, которая использует шифрование для отправки пароля через сеть. Рассматривается вариант применения аппаратной аутентификации.

Для повышения безопасности используются уникальные пароли для проведения каждого экзамена/зачета, централизованное хранение ведомостей, резервное копирование ведомостей, журналирование процесса заполнения ведомости.

Для проведения экзамена/зачета генерируется уникальный пароль, предоставляющий преподавателю доступ к документу — ведомость. Для передачи пароля преподавателю есть возможность использовать смс-рассылку. Пароль действителен только на период проведения экзамена/зачета до момента закрытия ведомости. Таким образом, не только улучшается безопасность системы, но и от преподавателей требуется определенная аккуратность и оперативность внесения информации. После закрытия ведомости есть возможность вывести ее на печать.

В будущем планируется расширение функциональности: предоставить механизм регистрации результатов рубежного контроля знаний; предоставить преподавателю доступ к справочной информации о студенте (например, какие оценки получил студент на лабораторных работах, на промежуточных контрольных точках); ведение контроля посещаемости занятий студентами.

Владимир Атаманов
Переславль-Залесский, Аспирант ИПС РАН

Проект: Поиск минимальных существенных замкнутых классов в P_k

Использование СПО в задаче поиска замкнутых классов

В моей работе рассматривается задача поиска минимальных существенных замкнутых классов многозначных логик. Доказано, что данная задача сводится к рассмотрению конечного числа функций. Тем не менее уже для случая трехзначной логики полный перебор

не представляется возможным. Но в некоторых случаях поиск можно свести к разбору относительно небольшого набора функций, который проще всего произвести при помощи ЭВМ.

На языке C был разработан ряд программ, позволяющий решать задачу поиска замыкания некоторого набора функций с фиксированным числом аргументов. Произведена сложностная оценка алгоритма замыкания функции и доказана его остановка через конечное время. В процессе исследования алгоритмы существенно изменились, что позволило за приемлемое время полностью решить поставленную задачу для случая трёхзначной логики.

Григорий Злобин, доцент
Львов, Львовский национальный университет имени Ивана Франко

Сравнительный анализ исплользования СПО в высших учебных заведениях Беларуси, Российской Федерации и Украины

На основании материалов научно-практических конференций «FOSS Lviv-2011», «FOSS Lviv-2012» сделан сравнительный анализ использования свободного программного обеспечения в высших учебных заведениях Беларуси, Российской Федерации и Украины. Примеры использования свободного программного обеспечения сгруппированны по направлениям: ПО поддержки учебного процесса, дополнительное ПО, используемое студентами в самостоятельной работе, ПО для использования в учебных курсах.

Ключевые слова: свободное программное обеспечение, операционная система GNU/Linux, офисный пакет OpenOffice.org.ukr, системы компьютерной математики.

Создание в 1981 г. фирмой IBM персональной ЭВМ IBM PC с открытой архитектурой привело к появлению IBM-совместимых ПЭВМ, которые производили во многих странах мира. Не отстали от этих стран СССР и страны Совета экономической взаимопомощи, которые стали выпускать целый спектр таких ПЭВМ:

ЕС 1840, ЕС 1841, Искра 1030, Нейрон (СССР);

ЕС 1834, ЕС 1835 (ГДР);

ЕС 1839 (НРБ).

Для ПЭВМ советского производства были созданы русскоязычная операционная система АльфаДОС, текстовый редактор Лексикон, текстовый редактор Text tip (Болгария), текстовый процессор Нейрон-текст, табличный процессор Нейрон-счет, СУБД Нейрон-база. Трудно определить, насколько лицензионно чистыми были АльфаДОС, Нейрон-текст, Нейрон-счет, Нейрон-база, ведь благодаря «железному занавесу» применить к СССР санкции по поводу нарушений авторских прав собственников программ было непросто. Вскоре после распада СССР во многих странах СНГ начали сборку IBM-совместимых ПЭВМ из комплектующих, которые ввозили преимущественно из стран Юго-Восточной Азии. На эти ПЭВМ устанавливали преимущественно пиратские версии как системного, так и прикладного программного обеспечения. Очевидно, что стоили эти ПЭВМ значительно дешевле аналогичных ПЭВМ европейского и американского производства, не говоря уже о ПЭВМ фирмы Apple. Поэтому операционная система MS DOS и офисный пакет Microsoft Office стали стандартом де-факто в высших учебных заведениях стран СНГ. Способствовало ли распространению пиратского ПО в высших учебных заведениях стран СНГ отсутствие законодательства о защите авторских прав собственников программ, сейчас сказать трудно, однако почти десять лет мы без ограничений копировали и устанавливали пиратские копии проприетарного ПО. В Беларуси, Российской Федерации и Украине законы о защите авторских прав собственников программ приняты в период с 1996 г. (Беларусь) по 2001 г. (Украина). В Российской Федерации с 1993 г. вступил в силу закон об авторском праве и смежных правах, который утратил силу с 1.01.2008 г. в связи с вступлением в силу четвертой части Гражданского кодекса РФ. Однако это мало повлияло на ситуацию с пиратским ПО в высших учебных заведениях этих стран. Случаи преследования высших учебных заведений за нарушение авторских прав в области программного обеспечения были немногочисленными и не всегда их проводили с целью защиты авторских прав владельцев программ. А вот применение законов о защите авторских прав собственников программ к субъектам хозяйственной деятельности начало создавать давление на высшие учебные заведения — «учите своих выпускников того, с чем они будут работать на наших рабочих местах». Ведь многие фирмы переходили на СПО с тем, чтобы уменьшить сумму лицензионных выплат владельцам проприетарного ПО. Еще одним аргументом в пользу изменения ситуации с использованием СПО в высших учеб-

ных заведениях Беларуси, Российской Федерации и Украины стало начало эры мобильных рабочих мест — трудно предсказать, какая ОС и какое прикладное ПО будет установлено на нетбуке, планшете или смартфоне сотрудника фирмы. Появление мобильных рабочих мест и быстрая смена версий системного и прикладного ПО побуждает высшие учебные заведения к отказу от технологической направленности лекционных курсов, связанных с компьютерными технологиями, в пользу фундаментальной составляющей. А это приводит к появлению рассуждений типа «если мы должны научить студентов основам работы с графическим интерфейсом в любой ОС, то почему это должна быть дорогая Microsoft Windows? Может, целесообразнее делать это в свободной и бесплатной GNU/Linux?». Однако отказ от наработок методического обеспечения для преподавателей высших учебных заведений оказался достаточно непростым процессом, особенно в условиях безнаказанности за использование пиратского ПО. За время от подписания Беловежского соглашения о прекращении существования СССР Беларусь, Российская Федерация и Украина прошли каждая свой путь развития и было бы интересно сравнить состояние с использованием СПО в высших учебных заведениях этих стран.

I. Использование СПО в высших учебных заведениях Беларуси

Сегодня рынок труда Беларуси нуждается в изучении многих проприетарных программ, начиная от Microsoft Windows и заканчивая специализированными CAD/CAM-системами. До недавнего времени риск от использования нелицензионного ПО был минимален, что не способствовало распростанению СПО. После создания в 2010 г. белорусского представительства Microsoft началась работа по преследованию нарушителей авторских прав Microsoft [8]. В первую очередь проводится работа разъяснительного характера с компаниями и частными лицами, нарушающими авторские права. Если она не приводит к какому-то результату, то в этом случае белорусское представительство Microsoft обращается в правоохранительные органы и суды. В настоящее время в работе находится около десятка дел по отношению к организациям, по некоторым из них рассматриваются дела об административных правонарушениях, по некоторым — вопрос о предъявлении гражданских исков.

Доля легального ПО выросла в последние годы благодаря специальным скидкам поставщиков и высоким экономическим показателям в 2011 г. Однако экономический фактор не является решающим для выбора СПО. Поэтому использование СПО в высших учебных заведениях обычно обусловлено его техническими преимуществами по сравнению с проприетарными аналогами или требованиями рынка труда. Выбор ПО сервера может быть единственным исключением, поскольку он значительно зависит от личных предпочтений системных администраторов.

В последние годы наблюдается рост интереса корпоративных работодателей к GNU/Linux, преимущественно для встраиваемых и серверных систем. Поэтому крупные фирмы активно заявляют о своих тренингах и семинарах, посвященных этой теме.

Использование СПО в высших учебных заведениях Беларуси можно разделить на три направления:

1. ПО поддержки учебного процесса (преимущественно системное ПО на серверах и рабочих станциях). В основном системное СПО на рабочих станциях представлено GNU/Linux в режиме двойной загрузки как альтернативной ОС в компьютерных классах кафедр, которые обучают программированию студентов инженерных специализаций. В педагогических учебных заведениях Linux на настольных компьютерах используют изредка из-за недостаточной распространенности GNU/Linux в школах Беларуси. В то же время в некоторых университетах зафиксировано использование Linux в тонких клиентах с терминальным Windows-сервером (например, Гродненский государственный университет имени Янки Купалы);

2. дополнительное ПО, используемое студентами в самостоятельной работе. К этой группе ПО можно зачислить офисный пакет OpenOffice.org и браузер Firefox;

3. ПО для использования в учебных курсах. В этом направлении СПО преимущественно используют в инженерных высших учебных заведениях, особенно тех, которые осуществляют обучение ИТ-специалистов, а именно: СПО для обучения программированию на языках Ассемблер, Java и PHP, SciLab для выполнения математических расчетов, QCAD/LibreCAD, Blender, cirquit CAD для изучения систем автоматизированного проектирования, использование свободных систем виртуализации

VirtualBox и KVM для изучения операционных систем, использования Moodle и iTest для тестовой проверки знаний студентов.

Отдельно отметим использовании СПО для кластеров и национальной ГРИД-системы Беларуси, в которую вовлечены ресурсы ведущих университетов (Белорусский государственный университет, Гродненский государственный университет имени Янки Купалы, Белорусский государственный университет информатики и радиоэлектроники, Белорусский национальный технический университет), научных учреждений и предприятий страны в рамках совместной российско-белорусской программы СКИФ-ГРИД.

Использование СПО в высших учебных заведениях Беларуси отражено на рис. 1.

Рис. 1: Использование СПО в высших учебных заведениях Беларуси

II. Использования СПО в высших учебных заведениях Российской Федерации

В отличие от Беларуси, в Российской Федерации в 2008 г. принята концепция развития разработки и использования свободного программного обеспечения. В рамках этой концепции в 2008-2010 гг реализована программа использования СПО в школах Российской Федерации (в 35% школ СПО установлено на более 50% компьютеров). Отметим, что, в отличие от Беларуси и Украины, в Российской Федерации прослеживается значительная активность контрольных органов по поводу лицензионности программного обеспечения. Как следует из обзора судебных дел [11] в Российской Федерации вынесены приговоры:

в 2012 г. 30 приговоров; в 2011 г. 43 приговора; в 2010 г. 70 приговоров; в 2009 г. 92 приговора; в 2008 г. 127 приговоров. Наиболее резонансным было дело А. Поносова, которое и привело к созданию в 2008 г. общественной организации «Центр свободных технологий». Как следует из [1-3], в большинстве высших учебных заведений Российской Федерации используют как Microsft Windows, так и GNU/Linux. Лишь в некоторых учебных заведениях руководство приняло волевое решение о полном переходе на СПО (Санкт-Петербургский торгово-экономический университет, Томский государственный педагогический университет, Нижегородский радиотехнический колледж). Как и в Беларуси, использование СПО в высших учебных заведениях Российской Федерации можно разделить на три направления [3-5]:

1. ПО поддержки учебного процесса (преимущественно системное ПО на серверах и рабочих станциях). В основном, системное СПО на рабочих станциях представлено GNU/Linux в режиме двойной загрузки как альтернативная ОС в компьютерных классах кафедр;

2. дополнительное ПО, используемое студентами в самостоятельной работе (к сожалению, авторы не имеют данных об этой группе СПО);

3. программное обеспечение для использования в учебных курсах. В этом направлении спектр СПО значительно шире, чем в Беларуси. Здесь можно упомянуть использование СПО для изучения программирования на языках C/C++ (в Ярославском университете есть интересный опыт обучения программированию с ис-

пользованием СПО), Pascal (Free Pascal, Lazarus), Java, Haskel, Пролог; SciLab, Octave, Sage для выполнения математических расчетов (значительный опыт применения свободного математического программного обеспечения накоплен в университетах Новосибирска, Барнаула, Бийска) организация систем дистанционного обучения, использование свободных систем виртуализации для изучения операционных систем; инструментарий для филологического анализа текстов, использование инструментария верификации ПО в обучении магистров, создание электронных образовательных ресурсов поддержки учебного процесса для заочной формы обучения (авторы осознают, что реальный список используемого СПО значительно больше, однако в открытом доступе данных об этом пока нет)

В высших учебных заведениях Российской Федерации довольно активно используют вычислительные кластеры с СПО. По инициативе ректоров Московского государственного университета им. М. В. Ломоносова, Нижегородского университета им. Н. И. Лобачевского, Томского государственного университета, Южноуральского государственного университета создан «Суперкомпьютерный консорциум университетов России». В список TOP500 от декабря 2011 входят четыре российских суперкомпьютера (№ 18, 107, 119, 121).

Отметим, что в Российской Федерации накоплен значительный опыт разработки свободного программного обеспечения. В России разрабатывают дистрибутивы Linux: ALT Linux (`http://altlinux.ru`), Calculate Linux (`http://www.calculate-linux.ru`), ROSA (`http://rosalab.ru`). Наличие компаний, ведущих разработку СПО, позволяет создавать специализированные свободные программы и значительно упрощает реализацию проектов по внедрению Linux в школы и университеты.

III. Использования СПО в высших учебных заведениях Украины

В Украине «Государственная целевая научно-техническая программа использования в органах власти программного обеспечения с открытым кодом» утверждена в 2010 г., однако до реального ее исполнения пока не дошло. Как следует из [9] в Украине в отличие от Российской Федерации случаи нарушения авторских прав собствен-

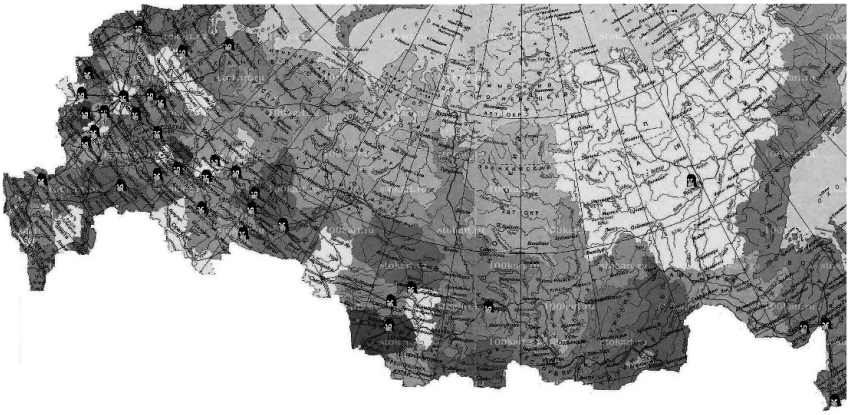


Рис. 2: Использование СПО в высших учебных заведениях Российской Федерации

ников программ соответствующие государственные органы проверяют в значительно меньшем объеме и преимущественно в хозрасчетных организациях. Особенно активными были проверки в 2006-2007 гг. В 2012 г. началась вторая волна проверок лицензионности ПО от Microsoft. В этом году впервые руководители вузов получили официальные письма из Microsoft с предложением легализировать используемые в вузах копии Microsoft Windows и Microsoft Office. В предновогоднем интервью [7] генеральный директор Microsoft Ukraine Д. Шимкив заявил о высокой вероятности инициирования нескольких показательных судебных процессов в Украине в 2013 г.

После приобретения ПЭВМ с преимущественно лицензионным Microsoft Windows и Microsoft Office на них устанавливают большое количество нелицензионного ПО, чем сводят на нет фантастически большие затраты средств на первичное приобретение ПО (Львовский национальный университет имени Ивана Франко до экономического кризиса 2008 г. каждый год покупал примерно 1000 ПЭВМ. Суммарная стоимость лицензий только на Microsoft Windows (OEM-версия) и Microsoft Office составляла почти 300000$ в год — достаточно большая сумма, как для учебного заведения!). В большинстве случаев выбор именно проприетарного ПО обусловлен даже не потребительскими

качествами этих программ, а фактом поверхностного ознакомления преподавателя с этой программой или даже наличием у него какой-либо книги с описанием программы. Выступления преподавателей и ученых на первой и второй конференции FOSS Lviv [1, 2] свидетельствуют о широком спектре использования СПО в высших учебных заведениях Украины.

Как и в Беларуси и Российской Федерации, использование СПО в высших учебных заведениях Украины можно разделить на три направления [1, 2]:

1. ПО поддержки учебного процесса (преимущественно системное ПО на серверах и рабочих станциях). В основном, системное СПО на рабочих станциях представлено GNU/Linux в режиме двойной загрузки как альтернативной ОС в компьютерных классах кафедр;

2. дополнительное ПО, используемое студентами в самостоятельной работе (к сожалению, авторы не имеют данных об этой группе СПО);

3. программное обеспечение для использования в учебных курсах. В этом направлении спектр СПО значительно шире, чем в Беларуси. Это использование систем компьютерной математики, организация систем дистанционного обучения, использование свободных систем виртуализации для изучения операционных систем, применение СПО для тестирования аппаратного обеспечения ПЭВМ; использования офисного пакета OpenOffice.org.ukr в курсе информатики высших учебных заведений, использование открытых средств программирования в обучении и научных исследованиях (авторы осознают, что реальный список применяемого СПО значительно больше, однако полные данные об этом авторам неизвестны).

В высших учебных заведениях Украины эксплуатируют вычислительные кластеры с СПО, наряду со специализированными установками широко используют распределенные кластерные системы и системы с выполнением вычислений на графических процессорах.

Учитывая вышеизложенное можно констатировать как широкий спектр использования СПО в украинских высших учебных заведениях от дистанционного обучения к разработке программного обеспечения, так и широкую географию использования СПО от Луганска на

востоке до Львова на западе и от Чернигова на севере до Одессы на юге

Рис. 3: Использование СПО в высших учебных заведениях Украины

Итак, независимо от наличия или отсутствия концепции использования СПО его используют в высших учебных заведениях Беларуси, Российской Федерации и Украине; количественно объем использования СПО в образовании выше в Российской Федерации, во всех трех странах министерства образования занимают отстраненную позицию в процессе внедрения СПО в университеты и школы, во всех трех странах уровень использования СПО в высших учебных недостаточен. Свобода выбора ПО, которой пользуются преподаватели, как правило, не приводит к выбору лучшего инструментария, а обусловлена привычками или стереотипами (часто ошибочными) преподавателей.

Литература

[1] Тези доповідей міжнародної науково-практичної конференції FOSS Lviv-2011. — Львів: Львівський національний університет імені Івана Франка, 2011. — 196 с.

[2] Матеріали другої міжнародної науково-практичної конференції FOSS Lviv-2012. — Львів, 2012. — 160 с.

[3] Алексеев Е. Р., Брагилевский В. Н. Использование свободного программного обеспечения в университетах и исследовательских учреждений Российской Федерации // Друга міжнар. наук.-практ. конф. FOSS Lviv-2012. — Львів, 2012.

[4] Брагилевский В. Н., Гуда С. А., Худолей Г. В. СПО на мехмате Южного федерального университета // Свободное программное обеспечение в высшей школе: Тез. докл. — М.: Альт Линукс, 2012.

[5] http://lists.raspo.ru/Plone/publichnye-drafty-dokumentov/dokumenty-komiteta-po-obrazovaniyu-i-vysshei-shkole/spo-v-rossiiskih-vuzah

[6] Kostiuk D. A., Pynkin D. A. Free/Libre software usage in the Belarusian system of higher educational institutions // Друга міжнар. наук.-практ. конф. FOSS Lviv-2012. — Львів, 2012.

[7] http://expert.com.ua/75297-microsoft-v-prednovogodnem-neformate-zhelezo-i-lyudi.html

[8] http://www.faeton.by/newspub/art_1.htm

[9] http://www.microsoft.com/ukraine/news/archive/default.mspx

[10] http://www.microsoft.com/rus/antipiracy/archive.aspx

Сергей Дочкин,
доктор педагогических наук, доцент, член-корреспондент Академии педагогических и социальных наук.
г.Кемерово, ГОУ «Кузбасский региональный институт развития профессионального образования»

Мониторинг внедрения СПО в образовательные учреждения профессионального образования региона

Аннотация

В статье рассматриваютсярезультаты исследования, проведенного с целью определения уровня внедрения продуктов свободного программного обеспечения в учреждения профессионального образования. С учетом того, что в основном данные вопросы рассматриваются в аспекте использования СПО в общеобразовательных учреждениях (школах) нами изучались учреждения начального профессионального и среднего профессионального образования, как основные компоненты образования, решающие задачи подготовки кадров для экономики региона

Применение информационно-коммуникационных технологий (ИКТ) в профобразовании ведет к появлению нового поколения образовательных технологий, позволяющих повысить качество обучения. В Кемеровской области уделяется большое внимание процессам информатизации именно в образовательных учреждениях профессионального образования (ОУ ПО), так как использование ИКТ способно радикально изменить существующую систему образования, и внедрение свободного программного обеспечения (СПО) в ОУ ПО являются его очередным этапом. Нами проводится периодический мониторинг процессов внедрения СПО в данный компонент системы образования, так как обычно процессы и результаты внедрения СПО в образование рассматриваются только для школ.

В целом использования характеризовали процесс внедрения свободного программного обеспечения в учреждениях профессионального образования (УПО) как слабым. Не обеспечивался доступ педагогического сообщества к объективной информации о спектре СПО и возможностях его использования: в вопросах оснащенности лицензионным и свободно распространяемым программным обеспечением

более трети респондентов показали свою низкую осведомленность, большая часть респондентов не владеет терминологической базой, и не понимает сущности задаваемых вопросов. Малое количество отечественных образовательных программных продуктов для использования с программами СПО отметили 87% опрошенных. Отсутствие достаточного количества учебно-методической литературы по тематике отметили 69% опрошенных. Сделан вывод о необходимости продолжения повышение квалификации ППР в области СПО, разработки механизма информирования педагогической общественности о существующих разработках, а также разработки модульных методик, программ и курсов, направленных на более активное использование свободно распространяемого ПО.

В исследовании 2012 года приняли участие 33 ОУ НПО и 37 ОУ СПО, всего 70 ОУ ПО региона (72,2% от общего числа). В данных ОУ профессионального образования по состоянию на май 2012 года использовалось 6101 компьютер, и на 90,9% компьютеров в ОУ НПО и 89,7% компьютеров в ОУ СПО установлена Windows, в то время как под GNU/Linux функционирует только 8,5% ПК в учреждениях НПО и 9,4% компьютеров в учреждениях среднего профессионального образования. Таким образом, в настоящее время только 8–9% компьютеров в ОУ профессионального образования функционируют под ОС GNU/Linux. И это при том, что современное коммерческое программное обеспечения продолжает оставаться недоступным по финансовым причинам для большинства ОУ ПО. Обращает внимание на себя и тот очевидный факт, что активность использования операционных систем из состава СПО больше для ОУ среднего профессионального образования.

В тоже время ситуация с коммерческими программными продуктами также достаточно любопытная. Как показало исследование, только 78,8% ОУ НПО и 83,8% ОУ СПО смогли указать наличие лицензий на использование операционных систем, 60,6% ОУ НПО и 56,7% ОУ СПО — на использование офисных пакетов. Дальнейшее исследование позволило сделать вывод, что в ОУ ПО региона существует серьезная проблема по обеспечению компьютерной техники лицензионным программным обеспечением и обеспечение точного учета приобретаемых программ и лицензий. Сложившаяся ситуация подтверждает необходимость оперативного решения данного вопроса, и один из путей решения заключается в постепенной и частичной замене проприентарного программного обеспечения на СПО. При этом

количество ОУ ПО использующих какие-либо программы из состава СПО не велико, но в регионе ОУ ПО имеют опыт использования иных операционных систем, отличных от Windows. Среди ОУ НПО наибольшей популярностью пользуются системы GNU/Linux на основе дистрибутива Mandriva, Ubuntu и ALT Linux (по 18,2% среди опрошенных ОУ НПО), на втором месте по популярности дистрибутивы Debian и Linux Mint (по 12,1% среди ОУ НПО). Всего в системе НПО региона в той или иной степени используется 5 дистрибутивов, один из которых — российская разработка. Перечень дистрибутивов GNU/Linux, используемый в ОУ СПО более широк, и насчитывает — 10 разнообразных дистрибутивов, лидируют операционные системы GNU/Linux на основе дистрибутива Fedora (35,1%), ALT Linux (29,7%), затем — дистрибутивы Mandriva (21,6%). Всего 48,5% ОУ НПО и 72,97% ОУ СПО в той или иной степени используют программы из состава свободного программного обеспечения.

Программы СПО используются для преподавания всего цикла дисциплин из состава образовательных программ. Наибольшее количество программ СПО используется на занятиях по информатике — 27,3% ОУ НПО и 43,2% ОУ СПО. Однако, использование данных программ в ходе освоения других дисциплин не значительно меньше, примерно на треть и составляет в среднем для ОУ НПО — 15%, для ОУ СПО — 32-33%. Это свидетельствуют о том, что персонал ОУ ПО региона знаком с продуктами на основе СПО, тем более их использование не свидетельствует о полном отходе от Windows-приложений. Наибольшей популярностью среди пользователей ОУ ПО пользуется пакет OpenOffice.org, веб-браузер Mozilla Firefox и растровый графический редактор Gimp. Но этими программами не ограничивается спектр используемых программ, среди которых Inkscape, Lazarus, 7-zip, Opera.

Подключение ОУ ПО к Интернету открыло возможность получения свободно распространяемых программ и их использования, выбора того перечня программ, которые соответствуют имеющемуся оборудованию и потребностям. При этом полный переход на свободные программы не планирует ни одно УПО. Частично планируют заменить Windows на их «свободные» аналоги только 69,7% ОУ НПО и 81,08% ОУ СПО, при этом в остальных УПО вопрос не обсуждался, и решение еще не принято.

При этом одной из основных причин отказа в рассмотрении программ из пакета СПО в качестве полноценной замены проприентар-

ных программ указывалась невозможность использования ряда справочных и расчетных систем. Однако в ходе исследования было выявлено, что перечень информационных систем, которые используют УПО ограничивается небольшим набором бухгалтерских и справочных программ, по которым есть «свободные» аналоги, функционирующие в GNU/Linux.

На данный момент СПО пока частично вводится в образовательный процесс ОУ ПО области, при этом наши исследования показали, что более 75% учреждений готовы внедрять СПО, 76% ответственных за информатизацию в своих ОУ ПО, имеют опыт использования СПО; ведется работа по адаптации Windows-ориентированных электронных изданий, накопленного контента на свободную платформу. В 8 УПО развернуты «опытовые» зоны, в которых ведется отработка различных вопросов использования программ СПО.

Вторая часть проведенного мониторинга была направлена на определение их готовности к использованию СПО в своей деятельности. За счет принятых мер в области удалось существенно повысить уровень ИКТ-компетентности ППР в области коммерческих продуктов и в среднем 75–80% преподавателей УПО используют средства ИКТ в своей деятельности. Однако СПО используют в среднем 11–12% ППР ОУ НПО и около 30% ППР ОУ среднего профессионального образования. При этом представители администрации имеют более высокий уровень подготовленности и большую заинтересованность в переходе на использование данных продуктов. Одним из способов решения данной проблемы является плановое обучение всего персонала УПО применению программ из сферы СПО и подробное информирование об особенностях их применения. На период проведения исследования около 60–70% всех административных работников и 60–65% ППР ОУ ПО нуждались в повышении квалификации в области СПО; углубленное изучение данных продуктов требовалось для 20–30% работников администраций и 15–25% ППР.

Проведенный мониторинг позволил сделать вывод, что в настоящее время образовательные учреждения начального и среднего профессионального образования региона имеют незначительный опыт использования в своей деятельности программ из состава СПО. Однако существующий уровень развития материально-технической базы и подготовки кадров требует постепенной и систематической работы по подготовке к миграции данных учреждений на свободное программное обеспечение, организованной органами власти, и в качестве основ-

ного направления деятельности следует выбрать целенаправленную подготовку персонала учреждений профессионального образования.

Алексей Ильченко, Михаил Шигорин
Киев, FS-Comm

IT-структура частного образовательного учреждения

Поднимая вопрос о IT инфраструктуре частного образовательного учреждения необходимо, для начала, определить, что же отличает его от учреждений государственных и очертить общее понятие IT структуры, точнее, требования к ней и их производные.

Из основных отличительных особенностей частной школы можно выделить:

- меньшие территориальные размеры
- меньшие топологические размеры
- меньшие размеры штата и количество учащихся
- иные критерии формирования бюджета
- более высокий (в т.ч. профессиональный) уровень персонала
- более жесткие требования к уровню внедрения и интеграции
- бОльшая готовность к внедрению новых технологий
- необходимость дельнейшей капитализации внедряемых технологий
- неготовность к содержанию собственных (мощных) IT подразделений
- ...

Все эти, равно как и опущеные в списке, причины обуславливают требования к IT структуре. Она:

- максимально оптимизирована в соотношениях цена / качество / коммерциализируемость
- создается и сопровождается сторонними организациями и специалистами
- к ней предъявляются высокие требования по уровню качества

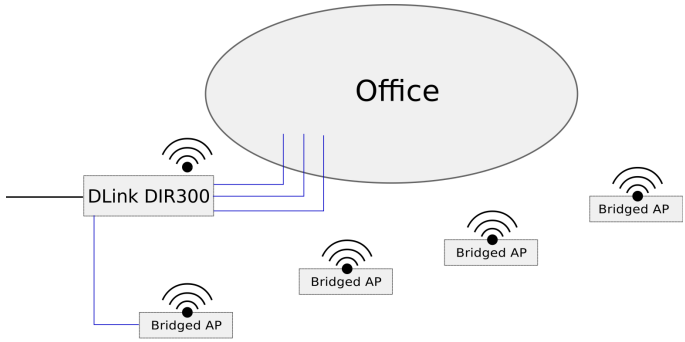


Рис. 1: Старая архитектура

- имеет меньшие топологические показатели при больших технологических

Рассмотрим проект частной школы-пансиона, находящейся на Украине, в Киевской области.

Как вполне видно из иллюстрации, говорить о промышленном использовании наличествующих средств не приходится – оборудование и структура с трудом справляются с задачами административного аппарата.

Для построения новой инфраструктуры были заявлены следующие пожелания:

- простота обслуживания
- функционирование в режиме 24/7
- высокая отказоустойчивость
- современная офисная инфраструтура
- возможности синхронного и асинхронного оповещения, в т.ч. с учётом приоритетов
- возможность интеграции с телефонией и системой оповещения
- унификация работы с учебным материалом
- возможность потокового аудио- и видеовещания
- возможность проведения интерактивных уроков с использованием проекторов

- возможность проведения индивидуальных интерактивных занятий
- возможность блицконтроля успеваемости
- возможности удаленного обучения, проведения конференций и семинаров
- возможности формировать гибкую аудиторию за счет мобильных пользовательских станций
- возможность принудительной физкультпаузы
- возможность организации видеонаблюдения
- чёткие бюджетные рамки
- возможность поэтапного внедрения и запуска частей функциональности

При участии специалистов компаний FS-Comm, Massive Solutions и ALT Linux был разработан проект инфраструктуры образовательного учреждения, позволяющий реализовать все заявленные пожелания практически в полной мере. Стоимость проекта: около 2 млн рублей.

Начало работ — сентябрь 2012. Ориентировочный срок запуска в эксплуатацию в полном объеме — август 2014 года.

Описание оборудования:

- центральный коммутатор (1 шт) – управляемый гигабитный 16-портовый
- крыльевой коммутатор (3 шт) – управляемые гигабитные 16-портовые
- этажный коммутатор (9 шт) – неуправляемые гигабитные 16/24-портовые
- рабочее место преподавателя (РМП) (по количеству классов) — стационарный ПК/ноутбук/терминальный сервер (требование по оборудованию – поддержка сетевой загрузки и WoL)
- локально-стационарный коммутатор (по количеству классов — для каждого РМП и перемещаемой стойки)
- впускной маршрутизатор (1 шт) — межсетевой экран (поддержка VPN, 2 порта WAN + 4 портами LAN 1000Base-TX, поддержка SNMP, опционально точка доступа)

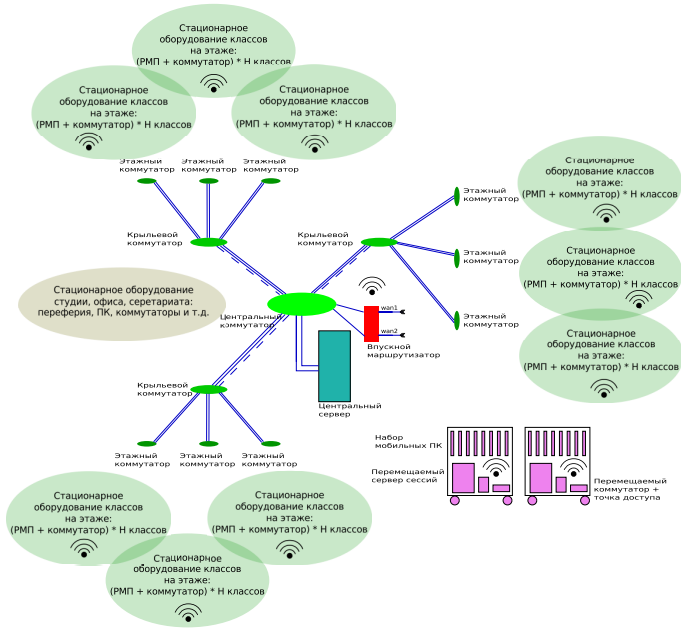


Рис. 2: Новая архитектура

- центральный сервер (1+ шт) — 64bit 2.5GHz+ 4+Gb RAM с запасом свободных слотов и поддержкой аппаратной виртуализации, 2x1Tb SATA HDD с запасом для еще двух, 2x1Gbps Ethernet
- шкафчик для размещения крыльевого оборудования 4UhR + UPS (3шт)
- перемещаемая стойка (1+ шт) — набор оборудования для локального развертывания инфраструктуры обучения (перемещаемый сервер сессий, источник бесперебойного питания, коммутатор + точка доступа, набор мобильных пользовательских ПК)

Описание программного обеспечения (основные моменты):

Серверная ОС — Linux (предпочтение отдано ALT Linux «Школьный сервер» для узловой ОС, как наиболее дружелюбной к администратору с данном сегменте)

ОС мобильных пользовательских ПК — Linux (предпочтение отдано ALT Linux в специальной модификации терминальных станций)

ОС РМП — Linux/Windows (Для ОС Linux предпочтение отдано ALT Linux за локализацию и развитое пользовательское комьюнити на территории СНГ).

Поддержка работы с ОС оборудования остальной инфраструктуры:

- Linux
- Windows
- iOS
- Android

Образовательные процессы — Moodle.

Видео и аудио конференции и вещание: Mumble / Sipwitch / Livemedia

Асинхронное оповещение: основанное на XMPP, e-mail

Центральный и крыльевые коммутаторы соединены транковыми соединениям для повышения надёжности и пропускной способности.Такая схема позволяет организовать резервирование наиболее нагруженных и сложных в обслуживании каналов, что является критерием для выбора управляемого оборудования.

За сентябрь — ноябрь 2012 года выполнено и введено в эксплуатацию:

- Оборудована необслуживаемая серверная.
- При содействии специалистов украинского представительства компании D-Link удалось выбрать бюджетное решение корневого оборудования серии DGS и DSR1000 в роли барьерного распределителя
- добавлена резервная линия повышеной надежности
- проложена и скоммутирована базовая основная и базовая резервная кабельные инфраструктуры
- покрытие сети Wifi полностью охватывает 3 крыла здания изолированной общей сетью
- административный сектор вынесен в отдельный сегмент центральной части с повышенным приоритетом
- введены минимальные политики безопасности.