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

Пятая конференция «Свободное программное обеспечение в высшей школе»

Переславль, 30–31 января 2010 года

Тезисы докладов

Москва,
Институт логики,
2010

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

Программа конференции

30 января

11:30 А. Е. Новодворский. Открытие. Информация оргкомитета

11:45 С. М. Абрамов. Приветственное слово

12:00–12:30 Н. Н. Непейвода

12:30–13:00 И. А. Хахаев

13.00–13.20 Кофе-брейк

13:20–13:50 И. В. Парамонов, А. М. Васильев

13:50–14:05 С. Э. Грегер

14:05–14:20 С. Э. Грегер

14:20–14:50 Е. А. Чичкарев

14.50–17.00 Обед и заселение

31 января

Вне программы

Н. Н. Непейвода
Ижевск, Удмуртский государственный университет

Принципиальные ограничения свободного софта

В последнее время я работаю над заказной статьей, которая должна дать обзор достижений и уроков конструктивной математики XX века, рассматриваемой в широком смысле слова — как спектр всех направлений от интуиционизма до польского полуконструктивизма. Поскольку одна из целей данной работы — получит уроки для современной информатики, в этом докладе приводятся некоторые выводы, к которым я пришел.

Для наших целей важны следующие характеристики конструктивизма:

1. Высказывание рассматривается как задача, а не как упражнение на абстрактное доказательство «истинности». Поэтому нас интересует не истинность, а реализуемость.
2. Эта реализуемость рассматривается как идеальное умственное построение. Следовательно, полученное построение может быть абстрактным высокоуровневым функционалом и вопрос о требуемых для его практического воплощения ресурсах ставится уже во вторую очередь.
3. В связи с этим полностью отвергается иллюзия всезнания, которая выражается в традиционной математике законом исключенного третьего. Это не означает введения новых истинностных значений. Скорее, сама концепция истинностных значений перестает быть основой для определения логики. Разные теоремы формально имеют одно и то же истинностное значение, но их реализации могут быть в корне различны.
4. В дополнение к этому в некоторых ветвях конструктивизма явно постулируется *незнание* и показывается, как использовать незнание в качестве основы для позитивных выводов. Например, беззаконные последовательности Брауэра являются конструктивной моделью физических датчиков, и основная аксиома Брауэра гласит, что мы в любой момент знаем об абстрактной

беззаконной последовательности лишь тип данных результата и конечное число ее результатов. Конечно же, применив конструктивное преобразование к беззаконной последовательности, можно получить последовательность, о которой известно несколько больше, например. что все ее члены — либо 0, либо 1. Но незнание остается существенно неустранимым.

5. Множества (или, что то же самое, предикаты) и типы данных — существенно различные сущности. Не всякий тип данных может быть описан как множество. Примером опять-таки служит тип беззаконных последовательностей.

Выяснилось, что варианты конструктивизма можно расположить в некоторый род иерархии по отношению к тому, какую долю знания они предполагают и какую долю незнания допускают.

Самый классичесский из них — польский полуконструктивизм, в котором все объекты сделаны алгоритмическими, но вся логика остается классической, из-за чего доказательства и построения проводятся отдельно. Для информатики этот конструктивизм практически ничего дать не может.

Советский конструктивизм А. А. Маркова и Н. А. Шанина — второй по степени знания об объектах. Незнание рассматривается как временное состояние. Предполагается аксиома религии прогресса, что творчески мыслящий человек может решить любую точно поставленную задачу, но неизвестно, захочет ли он этим заниматься, дойдут ли у него руки и хватит ли у него ресурсов на ее решение. Поэтому незнание хоть и временное, но неизбежное состояние. Но постулировать мы можем лишь знание.

В соответствии с этим предполагается, что для каждого эффективного функционала мы **знаем** вычисляющую его программу, и можем эту программу использовать в наших дальнейших построениях.

Получившаяся система оказалась замкнутой и последовательной в своих концепциях (как показал А. Г. Драгалин, в области арифметики советский конструктивизм *в принципе* действительно полон) и, соответственно, концептуально единой. Но в советском конструктивизме наряду с практически интересными примерами (напрмер, некорректными задачами и методами их превращения в корректные) возникает ряд монстров, в частности, вычислимая непрерывная всюду определенная на отрезке [0,1] функция действительных чисел, неограничен-

ная на данном отрезке. Построения в советском конструктивизме громоздкие и не очень практичные.

Прагматичный американец Э. Бишоп (E. Bishop) предложил следующее решение. Все наши преобразования — алгоритмы, но мы никогда в этом не признаемся, и, соответственно, не предполагаем знания их программ. Это положило начало целому букету конструктивных концепций, основанных на данной идее. В общем они оказались и технически более легкими, и концептуально более красивыми, чем советский конструктивизм.

Но самой красивой и самой мощной из конструктивных концепций так и остался голландский интуиционизм, в котором нет даже речи о том, что все эффективные преобразования — конечные алгоритмы, и в котором вовсю используется незнание как позитивный фактор. Техника преобразования некорректных задач, чисто технически более развитая в свое время в более практически ориентированном советском конструктивизме, полностью переносится в интуиционизм, и заодно появляется целый ряд новых красивых возможностей.

Сделаем теперь выводы из всего этого для информатики.

Первое, что здесь бросается в глаза — результаты теоретического анализа подрывают веру в универсальность концепции полностью открытого софта. Закрытость может иногда быть полезной.

Второе, на что обращал внимание, в частности, А. А. Шалыто, — подмена цели в свободном софте. Обращается внимание на открытость исходного текста программ, когда для практики очень часто важна не конкретная реализация, а полная и открытая программная документация к ней. Пользуясь этой документацией, желающий может переписать программу заново без всякого ущерба для всех, кто с ней работает, кроме хакеров того или иного сорта, ставящих на программу заплатки либо использующих недокументированные возможности (в данном случае и то, и другое равнозначно).

Далее, В. Ф. Чаусов, квалифицированный юрист в области патентного права и интеллектуальных прав, обратил мое внимание на то, что от требования открытости программного текста легко уйти легальными средствами, а неадекватная документация вполне может быть предметом преследования по закону. Я переделал его пример, который был частным, в общий алгоритм ухода от требования открытости. В частности, известны программы-обфускаторы. Но даже если не пользоваться примитивным обфускатором, выдающим формально открытый код, есть возможность прикрыться самыми лучшими

намерениями и сделать формально открытый код еще более закрытым, чем пропущенный через обфускатор. Для этого достаточно пропустить финальную версию программы, предоставляемую заказчику, через действительно мощный и интеллектуальный оптимизатор. Мы делаем как лучше, а получаемый код формально открыт, а фактически неперестраиваем.

Конечно же, в случае действительно коллективной работы над кодом программы в равной мере необходимы и открытая документация, и открытый код, но опять-таки я ставлю на первое место документацию и спецификации. Даже здесь открытый код не является догмой: можно сделать открытую надстройку над закрытыми модулями (что и приходится делать, если эти модули, скажем, полуаппаратные).

Таким образом, судя по всему, открытый софт оказался в ситуации, традиционной для сообщества прикладных математиков: поставлена некорректная задача, и требуется ее переставить как корректную.

И. А. Хахаев
Санкт-Петербург, Санкт-Петербургский торгово-экономический институт

LMS Moodle и видеоконференции

Аннотация

Рассматриваются варианты организации видеоконференций при обучении с помощью LMS Moodle с использованием только свободного ПО. Обсуждаются результаты экспериментов с различными вариантами видеоконференций в локальной сети.

Система дистанционного обучения (Learning Management System, LMS) Moodle имеет богатые коммуникационные возможности для обеспечения «живого» взаимодействия преподавателя с обучаемыми. Основными модулями являются форум и чат. Однако при наличии достаточно широкополосных каналов связи, а также при необходимости обращения преподавателя ко всей аудитории сразу целесообразно использовать технологию «вебинаров», т.е. видеоконференций.

Видеоконференция потенциально может быть реализована как с использованием модулей LMS Moodle, так и с помощью внешних по

отношению к Moodle сервисов. Интерес представляют варианты с количеством участников больше 2-х (т.н. «многоточечные» видеоконференции).

Среди многочисленных модулей Moodle имеется несколько средств для организации видеоконференций — модули *Covcell audio/video conference*, *OpenMeetings* и *WebClass*. Они используют сервер потокового видео *Red5*, написанный на Java.

Кроме того, имеются модули для интеграции с внешними службами видеоконференций — модули *Sclipo Live Web Class*, *DimDim module* и *Skype module*, однако принадлежность этих служб к СПО (особенно в случае Skype) вызывает сомнения (а DimDim имеет ограничения по количеству участников при бесплатном использовании).

И наконец, теоретически имеется модуль для интеграции Moodle с программной АТС (PBX) *Asterisk*. Однако этот модуль, судя по всему, заброшен и не поддерживается (с помощью Google удалось найти только японскую версию на sourceforge.jp).

Для организации видеоконференций как таковых имеется отдельный свободный проект — *OpenMCU* (возможны сборки под названием *OpenH323*).

Эксперименты проводились в локальной сети с использованием сервера на Debian 5.0 и клиентов с ALT Linux branch 4.1, оснащённых веб-камерами, микрофонами и аудиосистемами (колонки, наушники, встроенные динамики). При этом внешние службы видеоконференций (Sclipo, DimDim, Skype) не использовались.

На сервере был развёрнут Moodle 1.9, сервер *Red5*, *Asterisk* и *OpenMCU*. Сразу проявились ошибки в скрипте запуска *Red5*, поэтому в ходе экспериментов он запускался «вручную».

Модуль *Covcell audio/video conference* в настоящее время не поддерживается, в имеющейся версии не работает функция Whiteboard.

В целом эксперименты с использованием модулей Moodle, работающих с *Red5*, показали, что для «многоточечных» видеоконференций при количестве камер от 3-х и более наблюдается существенное (от 1 сек. и более) замедление в передаче изображений. Скорее всего, эффект связан с особенностями реализации сервера *Red5*.

Также проводились эксперименты с использованием *Asterisk* и *OpenMCU* с клиентами *Ekiga* параллельно с Moodle (хотя ничто не мешает использовать эти сервисы сами по себе). Настройка сервисов для описываемой задачи не представляет особой сложности, для этого достаточно имеющейся в Сети скудной русскоязычной документации.

Ekiga был выбран потому, что в ALT Linux это единственный «софтфон», поддерживающий передачу видео.

Ekiga с *Asterisk* позволяют прекрасно реализовать все возможности VoIP и видеозвонков, но количество участников конференции ограничивается одной парой.

Таким образом, для решения поставленной задачи («многоточечной» видеоконференции) в локальной сети остаётся только возможность использования *OpenMCU*.

И. В. Парамонов, А. М. Васильев
Ярославль, Ярославский государственный университет им. П. Г. Демидова

Опыт организации кроссплатформенной образовательной среды для обучения студентов программированию

Аннотация

В докладе описывается смешанная среда обучения, подразумевающая одновременное использование операционных систем GNU/Linux и MS Windows в рамках одних и тех же учебных курсов. Изложен опыт организации такой среды, указаны проблемы, которые возникли при её создании и эксплуатации, рассмотрены некоторые результаты.

В процессе модернизации учебных курсов на факультете информатики и вычислительной техники Ярославского государственного университета было принято решение о применении свободного программного обеспечения в образовательном процессе.

Помимо внедрения учебных курсов, посвящённых использованию и администрированию ОС GNU/Linux, был предпринят эксперимент по «погружению» традиционных курсов по программированию в смешанную среду, в которой одновременно использовались различные ОС. В числе затронутых преобразованиями дисциплин оказались программирование на языках C, C++, разработка приложений с графическим интерфейсом (I–II курсы), использование языка программирования Java и технологий Java Enterprise Edition (IV курс).

Организация кросс-платформенной среды предполагает, что студенты могут переходить из класса в класс, компилировать и запускать одни и те же проекты как под Windows, так и под Linux. Для обеспечения этого им предлагается использовать кросс-платформенные средства разработки: компилятор GCC (под Windows — его порт MinGW)

и фреймворк Qt. В качестве интегрированных сред разработки предлагаются Eclipse и NetBeans. Обе среды имеют свои преимущества и недостатки, как методического, так и технического характера. Ниже перечислены некоторые проблемы, с которыми мы столкнулись при их использовании.

- Основная методическая проблема Eclipse — сложность понятия рабочего пространства (workspace) для студентов. Среда устроена таким образом, что невозможно обойти необходимость импорта и экспорта проектов в IDE при переносе их между рабочими станциями. Дополнительная трудность возникает в связи с необходимостью ручного создания профилей компиляции и запуска. Всё это увеличивает издержки, отвлекая студентов младших курсов от собственно освоения учебных дисциплин. Отметим, что для NetBeans подобных проблем не существует.
- При организации смешанной среды изначально предполагалось использовать единое рабочее пространство IDE Eclipse. Это рабочее пространство должно было размещаться в личном каталоге студента на файловом сервере и быть доступным ему из любых компьютерных классов вне зависимости от установленной ОС. К сожалению, реализация данной концепции столкнулась с серьёзными трудностями. Оказалось, что плагин CDT, предназначенный для разработки приложений на C/C++, не информирует пользователя о неправильных настройках инструментов сборки проекта (toolchain), что приводит к появлению множества непонятных для студентов ошибок при открытии и сборке проекта, созданного под другой платформой. При смене платформы трудности возникают и при использовании Qt-плагина (одна из них связана с тем, что этот плагин хранит специфические для платформы пути к библиотекам и инструментам непосредственно в workspace).
- Чтобы обойти указанные выше проблемы, было решено создавать временное рабочее пространство каждый раз при входе в систему. Проекты в этом случае хранятся на файловом сервере, а в начале и при окончании работы студентам приходится выполнять процедуры импорта и экспорта проекта во временное рабочее пространство. Для того, чтобы упростить выполнение указанных операций для студентов было написано соответствующее руководство. Однако, данное решение является существен-

но менее удобным, чем то, которое планировалось реализовать сначала.

- Eclipse не содержит свободного редактора UI приемлемого качества, а конфигурирование среды для работы с платформой Java EE и сторонними библиотеками (например, JUnit) крайне нетривиально даже для студентов старших курсов.
- Основным недостатком IDE NetBeans являются высокие требования к компьютерным ресурсам. Это затрудняет её полноценное использование на частично устаревших компьютерах университета. В качестве компромисса на таких машинах было решено использовать NetBeans только для разработки приложений на языках C и C++.
- Первоначально использование IDE NetBeans было ограничено ввиду отсутствия в ней поддержки фреймворка Qt. Данная проблема решилась летом 2009 г. вместе с выпуском новой версии NetBeans 6.7 с полноценной поддержкой данного фреймворка. Проведённое тестирование обнаружило, что открытие C/C++/Qt-проекта на другой платформе не приводит к существенным проблемам.

Проведение занятий в смешанной среде показало, что студенты достаточно легко воспринимают смену операционной системы. IDE Eclipse на начальном этапе освоения вызывает проблемы (многие из которых, впрочем, вызваны невнимательным чтением руководства), однако, на последующих этапах среда перестаёт вызывать трудности и успешно используется студентами несмотря на необходимость выполнения процедур импорта/экспорта и изменения toolchain'а при смене платформы.

Несмотря на это, мы едва ли можем рекомендовать Eclipse в качестве среды разработки для обучения студентов младших курсов, а для смешанной среды она представляется практически непригодной. IDE NetBeans великолепно проявляет себя в ходе занятий по Java-технологиям — студенты не испытывают ни малейших проблем при её использовании на любой платформе, а также при переходе от одной платформы к другой. В связи с этим есть надежда, что использование NetBeans в качестве среды разработки на языках C/C++ (в том числе с использованием библиотеки Qt) даст лучшие результаты, чем применение Eclipse.

С. Э. Грегер
Нижний Тагил, Нижнетагильский технологический институт. Уральский государственный технический университет им. первого Президента России Б. Н. Ельцина

Пакет компонентов обеспечения информационной поддержки образовательного процесса для учебного портала на базе CMS Plone

Аннотация

Включение компонентов e-learning в состав портала дает ряд преимуществ по сравнению со специализированными (интегральными) LMS. Такое решение позволяет позиционировать систему электронного обучения как часть корпоративной системы и использовать все возможности портала для решения задач электронного обучения. Plone является одной из самых известных Open Source систем построения порталов Для Plone созданы, разрабатываются или модифицируются различные модули расширения, предназначенных для решения самых разнообразных задач как общего, так и специализированного характера, в том числе и в области образования. К сожалению существующие образовательные компоненты Plone построены с учетом организации процесса обучения специфичного для европейского и американского образования. В докладе представлено описание комплекта средств разработки (SDK) для поддержки электронного обучения в составе портала на основе CMS Plone, поддерживающих российскую специфику организации обучения. Представленные компоненты предназначены для создания электронных учебных курсов, формирование учебных групп, а также экспорта-импорта содержимого портала. Совместное использование представленных и стандартных компонентов Plone позволяет решать различные проблемы в области электронного обучения.

Plone — система управления содержимым сайта (CMS), позволяющая строить на ее основе сайты самых различных типов — от простых сайтов-визиток до корпоративных порталов. Plone является свободным программным обеспечением и имеет лицензию GPL.

Расширение возможностей Zope и Plone производится через подключение дополнительных модулей — так называемых продуктов. В результате появляется появление возможности включать в состав портала нового типа данных (контент-тип).

К сожалению существующие образовательные компоненты Plone построены с учетом организации процесса обучения специфичного

для европейского и американского образования. Большинство этих компонентов не локализованы для использования в России и не предназначены для использования с актуальной в настоящее время версией Plone 3.XX. Указанные обстоятельства явились причиной разработки комплекта программных компонентов для поддержки электронного обучения — E-learning SDK.

Комплект состоит из трех пакетов, каждый из которых предназначен для решения определённого круга задач. На рисунке (рис. 1) представлена общая UML-диаграмма классов SDK.

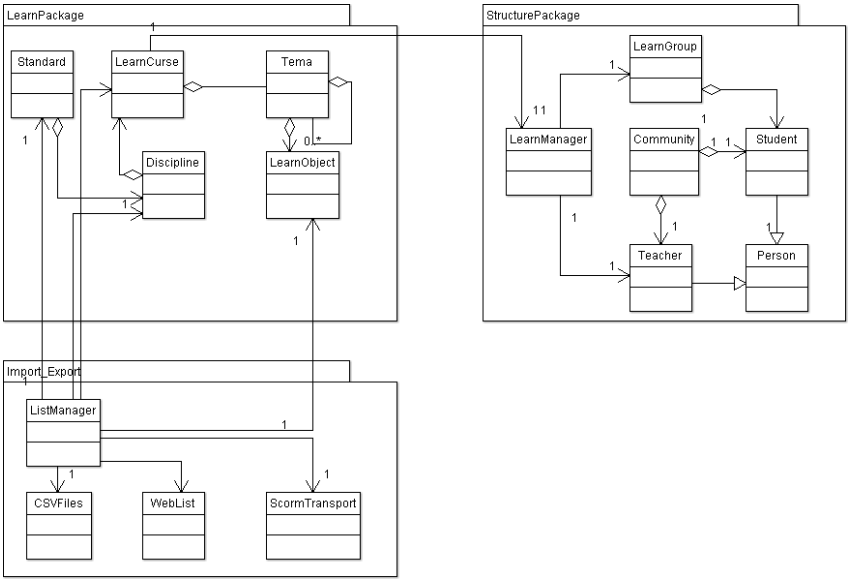

Рис. 1: Общая UML-диаграмма классов SDK

Пакет **LearnPackage** включает контент-типы, предназначенные для создание структуры учебных курсов, хранения электронных учебных объектов и организации различных стратегий обучения. Пакет включает в себя:

Контент-тип **Standard** (Стандарт) — хранит набор метаданных, регламентируемых требованиями Государственного стандарта по определенной специальности — продолжительность обучения, входные и выходные компетенции, список дисциплин специальности. От-

слеживает соответствие набора учебных дисциплин, представленных в портале списку дисциплин стандарта, предоставляет пользовательский интерфейс управления, формирует отчеты по устанавливаемым шаблонам.

Контент-тип **Discipline** (Учебная дисциплина) предназначен для хранения информации об учебной дисциплине, ее характеристиках, определяемых государственным стандартом.

Контент-тип **LearnCurse** (Учебный курс) обеспечивает вариативность учебных курсов, определяемую наличием различных целевых групп обучающихся, информационным содержанием курсов, их продолжительностью и временем проведения обучения по данному курсу. Фиксирует набор входных и выходных компетентностей слушателей курса.

Контент-тип **Tema** (Учебная тема) предназначен для построения иерархической структуры учебного курса, хранения информации о целях предъявленной темы, о наборах входных и выходных компетенций, продолжительности темы в составе курса. Является контейнером для учебных объектов.

Контент-тип **LearnObject** (Учебный объект) — является хранилищем для учебного контента или ссылкой на внешний ресурс. Такая возможность позволяет выделить весь учебный контент в отдельные репозитарии, предоставив его для совместного использования в различных курсах.

Пакет **StructurePackage** включает контент-типы, обеспечивающие построение организационной структуры ВУЗа, факультета, курса, учебных групп, а также структуру обеспечения внеучебной деятельности — секции, объединений и т.п. Позволяет проводить мониторинг учебной деятельности и распределение учебной нагрузки между преподавателями. *Контент-типы* **LearnGroup** (Учебная группа) и **Community** (Объединение) предназначены для образования организационной структуры. Контент-тип **LearnGroup** хранит ссылки на учебные курсы, определенные для учебной группы, формирует отчет о выполнении учебных заданий студентами, входящими в состав группы. *Контент-типы* **Teacher** (Преподаватель) и **Student** (Студент) предназначены для хранения различных данных о персоналиях, как общих данных, определяемых свойствами контент-типа Person, так и специальных, определяемых дополнительно. Структура портала определяется администратором портала через объект типа **LearnManager** (Учебный администратор).

Пакет **Import_Export** обеспечивает возможность взаимодействия с внешними, по отношению к порталу, системами. *Контент-тип* **WebLis**t, позволяет создавать различные справочники в составе портала. *Контент-тип* **CSVFile** служит для включения в состав портала файлов в формате csv, предназначенных в частности, для автоматического создания массивов объектов различных контент-типов. *Контент-тип* **ScormTransport** в настоящее время не реализован, предполагается что в его функции будет входитить импорт-экспорт учебных курсов в стандарте SCORM. *Контент-тип* **ListManager** предназначен для выполнения различных операций импорта-экспорта массивов экземпляров различных контент-тпов.

Представляется, что совместное использование представленных компонентов и стандартных компонентов Plone позволяет решать разнообразные задачи в сфере электронного обучения. В настоящее время проводятся исследования по разработке методики такого применения.

С. Э. Грегер
Нижний Тагил, Нижнетагильский технологический институт. Уральский государственный технический университет им. первого Президента России Б. Н. Ельцина

Опыт преподавания СПО при подготовке профессиональных программистов

Аннотация

В докладе рассматриваются итоги продолжительного периода внедрения учебных курсов по изучению СПО. Делается вывод о целесообразности интегративного подхода к преподаванию, приводится модульная структура курсов, рассматриваются вопросы технического и методического обеспечения преподавания.

Шесть лет на кафедре информационных технологий нашего института было принято решение о разработке курса «Web-дизайн» для студентов специальностей «Программное обеспечение вычислительной техники и автоматизированных систем» и «Информационные системы и технологии». Мною, как разработчиком этого курса, были выделены следующие ключевые моменты, определившие цели курса и его наполнение:

1. Действующий на тот момент (да и в настоящее время) стандарт подготовки по дисциплине «Web-дизайн» не соответствует реалиям ни с точки зрения научного содержания, ни с точки зрения практических навыков, приобретаемых студентами. Изучение только основ HTML, как это предписывается стандартом, явно не отвечало требованиям, предъявляемым к нашим выпускникам — профессиональным программистам.
2. Необходимо было обеспечить изучение современных технологий разработки, позволяющих создавать не только простые статические и динамические сайты, но и приложения корпоративного типа, такие как порталы.

Стремление повысит конкурентоспособность выпускников на рынке труда сразу отодвигало в строну такие платформы как Net и Java в силу ограниченности времени и невостребованностью таких знаний в сфере малого и среднего бизнеса, где в основном трудятся наши выпускники. В итоге был сделан выбор в пользу изучения сервера приложений «Zope» и приложений с ним связанных.

Модульная система курсов

Первый год чтения курса показал, что студенты испытывают сложности в изучении и связано это как с отсутствием учебных материалов на русском языке, так и с недостаточной практикой использования объектно-ориентированного подхода при разработке программного обеспечения. Оказалось, что концепции, используемые в Zope требуют пересмотра методики преподавания, отхода от парадигмы процедурного программирования и уделения большего внимания объектно-ориентированному программированию и проектированию. За недостатком места не буду описывать всю историю становлкеия а предъявлю лишь конечный результат.

Сейчас я читаю три отдельных курса, связанных в единое целое концептуально и технологически. Это:

1. Объектно-ориентированый язык программирования Python. Курс состоит из двух модулей. В первом модуле рассматриваются основы языка. При изучении модуля постоянно подчеркивается объектно-ориентированный характер языка, приводятся примеры использования такого подхода. Второй модуль посвящен вопросам создания графического интерфейса пользователя

на основе библиотеки wxPython, работе с базами данных, обработке текста.

2. Разработка web-приложений в среде CMS Django. Дается представление о системах разработки web-приложений, ориентированных на использование реляционных баз данных. Формируется представление о CMS, о их назначении, функционировании.

3. Разработка web-приложений с использованием сервера приложений Zope и CMS Plone. Рассматривается компонентный подход разработки приложений, использование объектной базы. Разрабатываются простые портальные решения на базе Plone.

4. Объектно-ориентированное проектирование информационных систем. Вопросы абстрактного проектирования на языке UML расссматриваются в контексте разработки информационных систем на основе Plone/

Хотя формально это независимые курсы, в реальности они сильно связаны между собой. Так, например, при выполнение курсовых работ по курсу проектирования ИС в большинстве случаев в качестве системы реализации выбираются Zope и Plone. Такой подход позволяет формировать у студентов единый комплекс компетенций, не разделять знания и умения по отдельным дисциплинам. Это обстоятельство привело к изменениям техники проведения лекционных и практических занятий. При проведении лекций часто используются примеры из других курсов, выявляется концептуальная общность терминологически разных понятий, подчеркивается необходимость системного подхода к решению поставленных задач.

Учебно-методическое обеспечение курсов

В течении почти шести лет преподавания был создан большой объем учебно-методического обеспечения. Кроме комплектов лекций в форме презентаций Power Point созданы:

- Учебный электронный курс «Язык программирования Python»;
- Учебный электронный курс «Использование сервера приложений Zope».

Опубликованы:

- Методическое указания «Web-сервер приложений Zope. Разработка динамических сайтов. Методические указания к изучению курса „Web-дизайн“»;
- Методическое указание «Web-сервер приложений Zope. Установка и интерфейс. Методические указания к изучению курса „Web-дизайн“»;
- Сервер приложений «Zope». Учебное пособие для вузов (гриф УМО по политехническому образованию);
- Администрирование и интерфейс пользователя CMS Plone (монография).

Техническое обеспечение обучения

Технологические трудности были обусловлены проблемами установки программного обеспечения в среде Windows, используемого в ученых классах института и с корпоративной политикой безопасности, используемой нашим вычислительным центром. В результате, в настоящее время все учебные курсы, связанные с СПО, проводятся с использованием специально скомплектованной виртуальной машины. Поскольку, к сожалению, студенты не имеют практики использования операционных систем на базе Linux, виртуальная машина работает на основе Windows XP. В системе установлены:

- Интерпретатор языка Python
- Пакет wxPython
- Пакет SqlAlchemy
- CMS Plone
- Редактор UML ArgoUML
- Генератор классов ArchgenXML

Такой состав программного обеспечения используется как при обучении по представленным курсам, так и при выполнении курсовых работ и дипломных проектов.

Е. А. Чичкарев
Мариуполь, Украина, Приазовский государственный технический университет
http://www.pstu.edu

Разработка и использование расширений OpenOffice для работы с системами компьютерной математики

Аннотация

Представлен краткий обзор современных вариантов организации графических и web-интерфейсов для систем компьютерной математики. Показаны возможности и ограничения взаимодействия Maxima и SciPy со средой OpenOffice. Представлены результаты разработки учебной документации и лабораторного практикума с использованием расширений OpenOffice для взаимодействия с системами компьютерной математики.

В настоящее время в Linux-системах существует значительное число математических и вычислительных программ различного назначения и уровня сложности. Большинство из них являются интерпретаторами того или иного входного языка. Отдельные пакеты включают собственный графический интерфейс пользователя и средства отладки и разработки (Scilab) , но для большинства пакетов (Maxima, Axiom, Gap, Octave, R и т.п.) графический интерфейс и средства разработки создаются отдельно, причем в нескольких версиях.

При решении реальных задач научного или технического характера с использованием компьютерного моделирования наиболее сложными и трудоемкими стадиями являются постановка задачи, выбор и анализ вычислительного алгоритма, не требующие графического интерфейса.

Однако при организации лабораторного или вычислительного практикума в учебных заведениях работа исключительно в терминале или простом текстовом редакторе вызывает серьезные затруднения. Студенты делают вывод, что в данной сфере совершенно нет никаких ресурсов: ни денег, ни современных специалистов, поэтому изучать предметы, связанные с компьютерным моделированием и вычислениями, нет никакого смысла; никакая красивая теория их не убедит в обратном (причем они отчасти правы: вычислительная математика и компьютерное моделирование используются обществом только вместе с реализующими их программами).

Учитывая наличие проприетарных программ с хорошо проработанным графическим интерфейсом, наличием локализованной помощи, подсказок, примеров, организация учебного процесса на базе программного обеспечения с открытым кодом встречает серьезные трудности.

Отчасти принципы разработки интерфейса и технологии работы с математическим ПО связана с технологией подготовки конечных документов.

Для большинства указанных систем компьютерной математики или специализированных пакетов существует Emacs-интерфейс (несколько вариантов для Maxima, octave-mode и т. п.). Учитывая возможность вывода графических иллюстраций в eps-файлы, включенная в большинство пакетов, естественная технология подготовки окончательных документов с использованием Emacs — использование Latex с экспортом содержимого буферов, содержащих результаты расчетов, в отдельные файлы, легко включаемые в проект конечного документа. Достоинством работы с использованием Emacs в качестве интерфейса является возможность интерактивного режима работы с вычислительными пакетами, при котором сохраняется состояние рабочего пространства при передаче ему очередной команды.

Альтернативным вариантом является использование в качестве графического интерфейса пользователя редактора Texmacs, позволяющего включать в конечный документ командные файлы, допускающие повторное выполнение, и результаты расчетов. Однако специфичная локализация и сложность экспорта документов в какой-либо другой формат делает Texmacs «вещью в себе», которая реально может использоваться лишь для подготовки небольших документов в pdf-формате.

Вероятно, наиболее завершенной оболочкой системы компьютерной алгебры, позволяющей решать широкий круг задач, и готовой к широкому использованию, является пакет wxmaxima, позволяющий работать с документами, включающими комментарии, блоки кода, графики. Присутствует также возможность экспорта в формат tex или html.

Существует также несколько превосходных оболочек для работы с R (rkward, Rcmdr), обеспечивающих экспорт результатов в txt или html.

Кроме того, для работы с open-source вычислительными пакетами используются и различные варианты клиент-серверных технологий

(в первую очередь sage) с доступом пользователей к вычислительному ядру того или иного пакета через web-интерфейс. В этом случае на клиентской стороне необходим лишь интернет-броузер (обычно с поддержкой JavaScript), а вся вычислительная работа выполняется на серверной стороне. При этом результаты работы сохраняются в html-файле.

Однако студенты для подготовки конечной документации используется преимущественно OpenOffice, поэтому несомненный интерес представляет непосредственный экспорт результатов расчетов в данный пакет.

Учитывая возможность разработки расширений OpenOffice на python, естественным решением для разработки плагина, позволяющего импортировать результаты расчетов в документам, является не Octave или SciLab, а пакет SciPy, обеспечивающий практически ту же функциональность, но реализованный на python. Командные файлы Scipy в виде текста копируются из документа OpenOffice, интрпретируются python, а возвращаемые текстовые результаты вставляются в документ.

Взаимодействие с maxima организовано по классической схеме (посредством сокетов). Текстовые строки передаются maxima без дополнительного редактирования, результаты расчётов размещаются в документе без дополнительного редактирования.

Как показало опробование организации доступа к вычислительным пакетам из Oowriter, данное решение является вполне работоспособным, и может быть использовано для курсового и дипломного проектирования.

Для работы с основными командами пакетов разработаны элементы графического интерфейса.

Аналогичный подход может быть использован и для работы с Octave, Axiom и т. п.

Е. Р. Алексеев, И. Шевченко
Донецк, Украина, Донецкий национальный технический университет
www.teacher.ucoz.net

Выбор свободно распространяемого программного обеспечения для учебного процесса (на примере кафедры ВМиП ДонНТУ)

Аннотация

Представлен сравнительный анализ современных специализированных образовательных дистрибутивов Linux.

При переходе на свободно распространяемое программное обеспечение в вузах СНГ одной из наиболее важных проблем является проблема выбора ПО. При этом приходится решать две задачи:

1. Выбрать дистрибутив свободной операционной системы семейства Linux.
2. Подобрать свободное программное обеспечение для ОС Windows.

При выборе дистрибутива в университете желательно, чтобы один и тот же дистрибутив подходил и для организации учебного процесса, и для проведения научных исследований. Кроме того, при выборе дистрибутива следует учитывать, что в университетах эксплуатируется большое количество старых и довольно старых компьютеров.

На кафедре «Вычислительная математика и программирование» Донецкого национального технического университета есть компьютерный класс свободного программного обеспечения. В этом классе проводят занятия с преподавателями и сотрудниками в рамках курса «Использование свободного программного обеспечения в учебном процессе», а также занятия со студентами по теме «Свободное программное обеспечение» в рамках курса «Введение в информатику». Стояла проблема выбора дистрибутива для установки на довольно слабые компьютеры (Celeron 600 Мгц, ОЗУ — 512 МБ, жёсткий диск 10-40 Гб). Такая конфигурация компьютеров является типичной для образовательных учреждений Украины. Кроме указанных выше требований, рекомендуемый дистрибутив должен: быть русско(украино)язычным, обеспечивать работу с необходимыми в учебном процессе приложениями (современные версии OpenOffice, Gimp, web-браузеров, почтовых программ, gcc, fpc, geany, lazarus, maxima,

scilab), а также должен хорошо работать на более современных компьютерах (на стационарных компьютерах и ноутбуках студентов и преподавателей). Рассматривались следующие дистрибутивы Linux:

- GOS (`http://www.thinkgos.com/gos/`) на основе Ubuntu Linux 8.04;
- Ubuntu 9.10 (www.ubuntu.ru);
- Ubuntu 9.10, версия UA Linux (http://ualinux.com);
- Runtu Office (`http://runtu.org/runtu-office/runtu-office.html`);
- Linux Mint 5 XFCE, Linux Mint 5 FluxBox (www.linuxmint.com);
- EduMandriva 2010 (www.edumandriva.ru);
- Debian 5.03 (www.debian.org);
- дистрибутив на базе Debian SkoleLinux (`http://www.slx.no/en/downloads`);
- Альт Линукс 5.0 Школьный Лёгкий (http://altlinux.org/Альт_Линукс_Школьный).

С учетом особенностей использования старых ПК для учебного процесса на кафедре были рекомендованы дистрибутивы *Debian 5.03* или *Альт Линукс 5.0 Школьный Лёгкий*. На более мощных компьютерах (процессор от 1 Ггц, ОЗУ $>$ 512 Мб) в ДонНТУ рекомендовано использовать Debian, Ubuntu или Mint.

Набор свободного программного обеспечения для каждого подразделения университета свой. Для решения проблемы выбора свободного программного обеспечения был разработан сайт «Свободное программное обеспечение в университете», на котором представлены описания многих свободных программ, а также ссылки на страницы загрузки программ. Кроме того, собран набор portable-версий свободно распространяемых приложений для вуза на базе оболочки RocketDoc.

Среди проблем, которые предстоит решать при дальнейшем внедрении свободного ПО в университетах, следует выделить следующие:

1. Недостаточное количество учебно-методической литературы. Может быть, ALT Linux следует от издания литературы, посвященной дистрибутивам ALT Linux, перейти к изданию литературы, посвященной свободному программному обеспечению и основать специализированное издательство. Наличие документации в Интернете не решает проблему учебно-методической литературы.

2. Отсутствие кафедр, специализирующихся на свободном ПО, в университетах СНГ. Пора всерьёз задуматься об организации кафедры свободного программного обеспечения в одном из университетов СНГ.
3. Инертность мышления преподавателей IT-дисциплин.

В. Г. Слугин
ФГОУ СПО «Нижегородский Радиотехнический Колледж»

Проект: cunewebform, elkartoteka, sadko
http://fireforge.net/projects/cunewebform/,
http://fireforge.net/projects/elkartoteka/,
http://fireforge.net/projects/sadko/

Внедрение, использование и сопровождение СПО в учебном заведении на примере ФГОУ СПО «НРТК»

Аннотация

Сейчас широко обсуждается вопрос перехода учебных заведений на свободное программное обеспечения. Говорят о технических возможностях, аналогах программ используемых в учебном процессе, совместимости с уже имеющимся оборудованием. Но, как показывает опыт Нижегородского Радиотехнического Колледжа, это важные но не основные вопросы, возникающие при переводе учебного заведения на свободное ПО.

Гораздо более важным оказался вопрос взаимодействия с государственными структурами и министерствами, управляющими учебными заведениями, такими как РАНО, Министерство Образования и органами, действующие по их указанию. Зачастую они слабо представляют что такое свободное программное обеспечение и лишь в редких случаях заботятся о совместимости присылаемых ими материалов и программного обеспечения со свободными операционными системами и свободными офисными пакетами.

Мы рассмотрим способы решения всех этих вопросов на примере перехода Нижегородского Радиотехнического Колледжа на использование свободного программного обеспечения как в учебном процессе, так и в работе подразделений и администрации.

Причины перехода

Зачастую это высокая стоимость коммерческого ПО. Но даже при наличии некоторого бюджета на покупку программного обеспечения вы сталкиваетесь с рядом других проблем, а именно:

- Запутанность ценовой политики компаний-производителей и их дистрибъютеров;
- Сложности с использованием копий ПО студентами на домашних ПК;
- Сложности с лицензированием дипломных и курсовых проектов студентов, созданных с использованием коммерческого ПО.

К тому же не стоит забывать о быстром моральном старении коммерческого по и стремлении производителя подтолкнуть покупателя к обновлению, а следовательно и покупки новых версий своего продукта.

Если ко всему этому добавить широкое распространение вирусов на коммерческих платформах выгода от использования свободного программного обеспечения и свободных платформ становится вполне ощутимой.

Преимущества, полученные от перевода HPTK на свободное программное обеспечение

В ходе перевода компьютерных классов, используемых в учебном процессе, и компьютеров сотрудников колледжа практически полностью удалось избавиться от угрозы вирусной атаки как на локальные компьютеры так и компьютерную сеть колледжа.

В учебном процессе используются различные операционные системы. За счет использования свободного программного обеспечения на серверах колледжа удалось создать интегрированную систему централизованной аутентификации и хранения данных пользователей на сервере. При этом, система работает абсолютно прозрачно, вне зависимости от загруженной на клиентском компьютере операционной системы. Студенты имею возможность использовать любой компьютер в колледже, загружаясь в любую нужную в данный момент операционную систему и входя под своим логином и паролем, получать доступ к своим файлам в их домашней директории, хранимой на сервере.

Широкое использование свободного программного обеспечения в учебном процессе дает возможность обеспечить студентов необходимым программным обеспечением на легальных основаниях. Раздавая LiveCD с необходимым набором ПО для выполнения домашних заданий нет необходимости контролировать дальнейшее распространения дисков.

Использование открытого формата ODF для работы с документами различного типа дает более широкие возможности по формированию необходимых документов из On-Line систем, используемых в колледже. А так же, дает возможность автоматической обработки присылаемых студентами работ.

Характерные вопросы при переходе на свободное программное обеспечение

На первоначальном этапе был проведен анализ используемого программного обеспечения и подобраны аналоги из свободных программ для обеспечения учебного процесса. Были заменены наиболее дорогие продукты такие как P-Cad, ElectronicWorkbanch, SpiderProject, Visio, MS VisualStudio, Dreamwaver, MS Sharepoint.

При внедрении была проведена большая работа по подключению а настройке различного периферийного оборудования. Различные типы сканеров и принтеров. Часть из них оказалась не совместима. Полученный опыт был использован при замене старого оборудования. При покупке нового оборудования обязательно производится предварительная проверка на совместимость со свободными платформами и программным обеспечением. Это дало возможность более легкой миграции оборудования по классам и подразделениям и уменьшило время настройки систем при плановом обновлении программного обеспечения.

Сложности, возникающие в процессе перехода и использования СПО в учебном заведении

На первоначальных этапах перехода на свободное программное обеспечение все сталкиваются с вопросом обучения сотрудников и преподавателей использованию нового ПО. Но этот процесс прохо-

дит достаточно легко в виду близости функционала и внешнего вида свободных программ с используемыми ранее аналогами.

Более сложно обстоит ситуация при работе с органами управления учебными заведениями. ФАО ведет переписку с учебными заведениями исключительно в форматах doc и xls, а в последнее время приходят документы и в форматах docx.

Многие on-line системы сбора данных об учебных заведениях ориентированы на работу лишь с web-браузером Internet Explorer и не выполняющие свои функции корректно в других.

Программное обеспечение присылаемое из РАНО по сбору данных для проведения ЕГЭ ориентировано исключительно на проприетарную платформу. Не смотря на то, что ранние версии этих программ работали в среде Wine, новые версии перестали корректно выполнять свои функции в этой среде. К тому же для правильного формирования отчетов требуется MS Office определенной версии.

Все эти сложности усугубляются обязательностью представления данных в сжатые сроки, санкциями в отношении учебного заведения при нарушениях сроков и отсутствия какой либо поддержки по данным продуктам при вопросах об их использовании в средах с набором свободного программного обеспечения.

Ю. Азовцев
Нижний Новгород, Нижегородский РадиоТехнический Колледж

Проект: перевод книги «Blender Basics 3-е издание»
`http://b3d.mezon.ru/index.php/Blender_Basics_3-rd_edition`

3D-моделирование и анимация в Blender для школ и ВУЗов

Аннотация

Основным сдерживающим фактором в широком использовании Blender в курсе «Компьютерная Графика» в школах и ВУЗах являлась отсутствие методического материала на русском языке. В конце 2009 года был завершен перевод книги Джеймса Кронистера «Blender Basics 3-е издание», которая является готовым методическим материалов рассчитанным на 40 часовой курс с планом по 2 академических часа на каждую тему (главу), включая практическое занятие. Книга распространяется абсолютно свободно для использования в учебном процессе.

Развитие пространственного мышления является одной из наиболее важных вещей в учебном процессе школы и ВУЗа. Это незаменимый навык как для студентов технических специальностей, так и для будущих архитекторов и художников. Раньше этим целям служили такие предметы как «инженерная графика» и «черчение». С появлением компьютерной техники в курсах многих специальностей появился предмет «компьютерная графика». Но на нем преподавали различные CAD системы продолжая строить плоские проекции объектов.

Но ни для кого не секрет что большинство школьников спят и видят себя авторами очередного «Шрек 3» или «крутой трехмерной игрушки». Почему бы не использовать этот уже существующий интерес для преподнесения нужного материала?

Blender относится к классу программ трехмерной визуализации и анимации. Программа позволяет создавать сразу трехмерные объекты, задавать им цвет. Создавать для каждого объекта свой материал и накладывать текстуру для придания вида стекла, камня или дерева (либо любого другого материала). Для анимирования персонажей Blender предлагает систему костей (арматура). В некоторых случаях удобно использовать физический движок, встроенный в Blender, для создания падающих объектов, дождя либо потока реки. Для архитекторов особенно удобным окажется возможность использования систем рендеринга с реальным просчетом падения и преломления лучей. Blender поддерживает порядка 8 рендеров, включая свободные Yaf(a)Ray, бесплатный LuxRender и, хоть и коммерческий, но всемирно признанный V-Ray. Так же. В Blender вы сможете сделать полный цикл создания анимации. Начав с создания модели, текстурировав и анимировав ее вы сможете наложить титры, эффекты и переходы на уже готовый видеофайл во встроенном в Blender видеоредакторе.

Blender, являясь свободным программным продуктом, позволяет проводить учебный процесс без ограничения доступа учащихся к среде моделирования ни дома ни в учебном заведении. Вы сможете обеспечить всех студентов необходимым количеством копий программы, при этом, давая им полную свободу в выборе операционной системы для работы с ней. Blender прекрасно работает в Linux, MacOSX и Windows. При этом интерфейс и функциональность программы абсолютно одинаковы во всех операционных системах.

Но при всех прелестях Blender до начала этого года у него был один значительный сдерживающий фактор перед широким использованием в учебном процессе школы и ВУЗа — это крайне малое количе-

ство документации и методического материала на русском языке. Для решения этой проблемы русскоязычным сообществом Blender был начат проект по переводу книги «Blender Basics» на русский язык. Эта книга была написана Джеймсом Кронистером (James Cronister) как методическое сопровождение к курсу трехмерного моделирования и анимации, который он ведет в Central Dauphill High School. Книга представляет собой готовую методическую разработку по курсу 3D моделирования. Каждая глава рассчитана на один либо два академических часа. Содержит теоретическую и практическую часть. При этом, объем теоретической части подобран очень сбалансировано, позволяет выполнить практику и не перегружает излишними подробностями. Перевод книги распространяется свободно и доступен как в web-варианте на сайте b3d.mezon.ru, так и в PDF версии, подготовленной для печати на бумагу (`http://b3d.mezon.ru/index.php/Blender_Basics_3-rd_edition`). Лицензия позволяет изготавливать любое количество копий, как на бумаге так и электронных для использования в учебных целях.

Возможности Blender, свободные условия распространения программы и наличие методического материала в виде книги «Blender Basics 3-е издание» делают эту среду 3D-моделирования просто незаменимым инструментом преподавания компьютерной графики школьникам и студентам.

Д. А. Костюк
Брест, Брестский государственный технический университет

Изучение низкоуровневого программирования и вычислительной архитектуры на базе платформы GNU/Linux

Аннотация

Рассмотрены преимущества обучения студентов низкоуровневому системному программированию и архитектуре ЭВМ в ОС GNU/Linux. Проанализированы сложности, связанные с отличиями платформы, выбор программного обеспечения, доступных информационных источников. Оценены результаты включения в учебный процесс.

Изучение языка машинных команд важно для студентов специальностей информатики и радиоэлектроники с двух позиций. Во-первых,

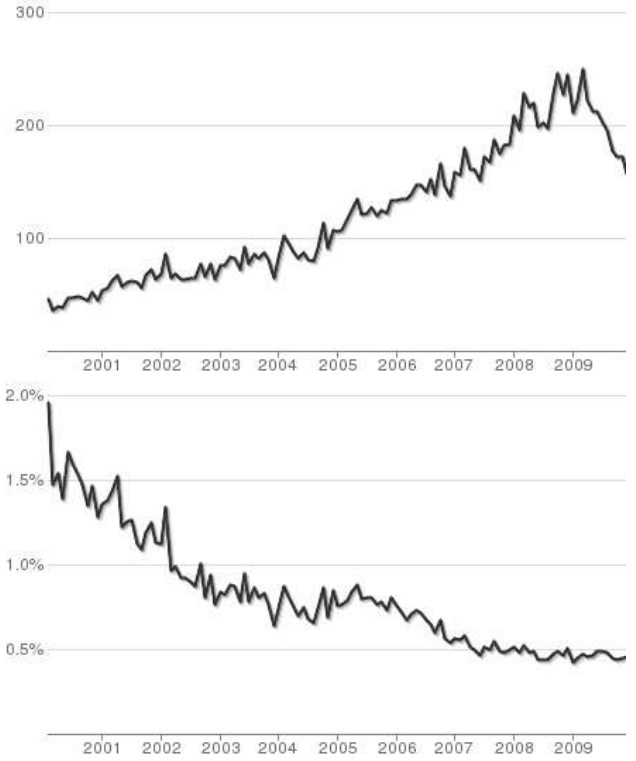

Рис. 1: Динамика использования ассемблера: число активных проектов и их доля от общего количества (по данным `http://ohloh.net`)

ассемблер актуален при разработке драйверов и другого аппаратнозависимого кода, трансляторов, оптимизации критических секций, а во-вторых — играет важную образовательную роль. Без машинных команд невозможно освоить устройство и принцип работы вычислительной техники. Пробелы в данной области служат источником неоптимального и уязвимого кода, приводят к заведомо неверным архитектурным решениям.

Можно выделить две объективные проблемы в изучении ассемблера. Во-первых, рост числа актуальных программных технологий и языков не позволяет уделять ему значительный процент учебной

нагрузки. Во-вторых, крайне архаична базовая платформа DOS, массово используемая для его изучения уложить из-за невозможности курс низкоуровневого программирования под Windows в отведенное число часов.

Программирование на ассемблере нельзя назвать характерным для Linux, но как платформа для его изучения эта ОС — идеальное решение. Она проста в освоении для системных программистов, предоставляет доступ к исходным кодам, демонстрирует неуклонный рост в ряде сегментов рынка.

С точки зрения ассемблера Linux характеризуется строго унифицированным интерфейсом системных вызовов, одинаковым для всех функций ядра (в отличие от «зоопарка» функций DOS). Упрощает освоение материала и то, что язык С знаком студентам к моменту изучения ассемблера (что закреплено белорусскими образовательными стандартами для технических вузов). Студенты обнаруживают многочисленные аналогии системных вызовов с функциями С и консольными командами.

В углубленном курсе различия аппаратных архитектур вносят некоторые сложности. Прикладной процесс не имеет полного доступа к аппаратным ресурсам, что не позволяет писать простые учебные программы в стиле DOS, взаимодействующие с устройствами на уровне портов ввода/вывода. Для освоения привилегированных инструкций студенту необходимо научиться создавать модули ядра — в первую очередь драйвера (учитывая востребованность подобной квалификации, это скорее плюс). В целом написание драйверов для Linux — несложная задача, а среди доступных руководств можно встретить даже ориентированные на использование чистого ассемблера [1].

Linux позволяет работать с устройствами на нескольких уровнях абстракции, которые могут быть основой для сокращенных курсов архитектуры ЭВМ — например, доступ к пространству портов ввода/вывода через /dev/port.

При апробации начального практикума нами использован терминальный Linux-сервер. Однако при низкоуровневом программировании устройств этот вариант неприемлем, т.к. студенты нуждаются в правах root и в возможности регулярно обрушивать систему. Решение на базе виртуализации показало некоторые отличия от поведения программ на реальных устройствах (из-за особенностей QEMU и его производных). Разумной альтернативой рабочим станциям с общедо-

ступным root-акаунтом представляется использование специализированного LiveCD-подобного дистрибутива.

Следует также отметить сложности, связанные с фрагментарностью и разрозненностью технической документации. Преподавательский состав нуждается в более подробных источниках, чем предлагаемые студентам методические материалы; кроме того, не все свободно владеют английским языком. Можно отметить качественные переводные руководства по ряду утилит GNU (make [2] и особенно gdb [3]), транслятору nasm [4], более привычному преподавательскому составу, чем GNU assembler, русскоязычное руководство [5], отвечающее тематике учебного курса, но не вполне применимое из-за выбора синтаксиса ATT. Из полезных информационных порталов можно назвать `http://opennet.ru/docs` и сайт программы «СКИФ» `http://skif.bas-net.by/bsuir`.

Однако, рассмотренные сложности не являются блокирующими, а апробация разработанных курсов показывает высокую усвояемость материала студентами и легкую адаптацию преподавательского состава.

Литература

[1] *Hyde R.* Linux Device Drivers in Assembly Language. `http://homepage.mac.com/randyhyde/webster.cs.ucr.edu/LinuxAsm/index.html`

[2] *Stallman R.M., McGrath R.* GNU Make. Апрель 2000 г. `http://linuxlib.ru/prog/make_379_manual.html`

[3] *Столмен Р., Пеш Р., Шебс С. и др.* Отладка с помощью GDB. Март 2000 г. `http://mitya.pp.ru/gdb/gdb_toc.html`

[4] Расширенный ассемблер: NASM. `http://www.opennet.ru/docs/RUS/nasm/` 06.02.2002.

[5] Ассемблер в Linux для программистов C: викиучебник. http://ru.wikibooks.org/wiki/Ассемблер_в_Linux_для_программистов_C

Г. В. Курячий, А. А. Панюкова
Москва, ВМиК МГУ, ALT Linux
http://uneex.ru/LecturesCMC

Опыт преподавания курса «Сопровожнение пакетов в Linux» на ВМиК МГУ

Волею судеб один из авторов доклада прочитал спецкурс по теме «Сопровожнение пакетов в Linux», а другой — принимал экзамен на факультете ВМиК МГУ. Несколько неожиданной оказалась профессионально-техническая готовность аудитории, и, как следствие, серьёзный потенциал для сообщества в лице студентов московского Университета.

Положение дел

Вот уже десятый год на факультете Вычислительной Математики и Кибернетики МГУ читается кафедральный спецкурс на UNIX-тематику. Каждый семестр курс новый (было всего два тематических повтора и один двухсеместровый курс). Последние несколько лет курсы базируются на ALT Linux / Sisyphus.

Помимо направленности на UNIX/Linux, курс имеет три специфические особенности, связанные с приёмом экзамена:

1. Экзамен проходит в виде беседы по всему материалу курса (обычно часа три). В беседе участвует один экзаменатор и трипять экзаменуемых.
2. При ответе на вопрос приветствуется аргументированное высказывание личных мнений, а также применение мыслительного аппарата. Применение любых информационных носителей, включая текст лекций, учебников, Google и мнение товарища, допускается (стоит иметь в виду, что при этом оценивается личный вклад экзаменуемого в собственный ответ, а в последнем случае — вклад товарища).
3. В число *экзаменаторов* часто попадают студенты-старшекурсники из *NIX-сообщества, квалификация которых делает идею принимать экзамен *у них* бессмысленной. Чем бездумно ставить им «автомат», лучше даать возможность получить дополнительный опыт и упорядочить понимание темы.

Кроме того, сам спецкурс совершенно факультативен: его можно прослушать, а потом не сдавать экзамена, или явиться, не сдать, и уйти, не испортив матрикула.

Умысел и деяние

В осеннем семестре 2009 года была — после аналитического обсуждения с аудиторией[1] выбрана на первый взгляд довольно специфичная тема Package Maintaining.

Причины такого выбора:

1. Желание автора;
2. Отсутствие (кажется, полное!) подобных курсов в академической среде;
3. Практическая ценность темы;
4. Широкий и довольно выразительный спектр затрагиваемых областей ИТ.

Ожидаемые проблемы:

- Высокий уровень требований к профессиональным качествам слушателя;
- Узкий круг заинтересованных;
- «Религиозные» войны.

Для придания курсу разносторонности была прочитана отдельная лекция про GNU Debian.

План курса (действительный):

- Дистрибутив ОС на основе свободного ПО: принципы формирования;
- Пакет как составная часть дистрибутива: требования и особенности; понятие сборки пакета;
- Сборка пакета из исходных текстов; upstream; спецификация;
- Сопровождающий (maintainer) пакета;
- Изолированная среда сборки (введение);
- Лекция Александра Герасёва о пакетах в GNU Debian;
- Работа с upstream, Составление спецификаций;
- Составление спецификаций — II, Исправление upstream (заплатки), Помещение в хранилище, Обратная связь с upstream и сообщения об ошибках;

[1] `http://uneex.ru/LecturesCMC/PackageMaintaining2009/Conspects/00`

- Git, Gear и git.alt;
- Git (лекция А. Герасёва);
- Практика сборки пакета (Live show).

Интервью с очевидцем

О том, как проходил экзамен, мы решили спросить –(прохожего)– Александру Панюкову, дважды участвовавшую в нём в качестве экзаменатора.

Первый экзамен

Как вы оцениваете аудиторию?

На первый экзамен пришли студенты, у которых процесс «линуксизации мозга» уже начался: даже если опыт использования был не очень богатым на самом деле, мышление уже можно было назвать Linux-ориентированным: просматривался соответствующий ход мысли, построение причинно-следственных связей, было заметно понимание идеологии. Есть ли какие-то замечания по уровню профессиональной подготовки экзаменуемых?:: Все студенты в той или иной степени знакомы хотя бы с одной VCS и BS из своего опыта, т.е. эта тема не была для них новой в курсе. Однако, именно с git и/или Bugzilla, на примере которых рассказывалась тема, имели дело далеко не все.

Если курс «инженерного плана», всегда есть опасность скатиться в перечисление инструментов и их возможностей. Как по-вашему, были ли в этом курсе основания для самостоятельного освоения предметной области?

На основании имеющегося материала, студенты довольно неплохо делают выводы о некоторых моментах, которые напрямую в лекциях не рассматривались (и при этом являются довольно специфическими, но могут быть додуманы с использованием здравого смысла).

Сейчас много говорят о растущей популярности Ubuntu. Отразилось ли это на знаниях пришедших на экзамен?

В основном собравшиеся студенты пользовались RPM-based дистрибутивами, как следствие — имели о них более полное представление. Несмотря на одну лекцию про Debian, представления о предмете были существенно более размытые, чем про ALT-специфические и RPM-специфические части. Был один арчевод — у него было представление об арче, но он скорее нерепрезентативен, так как один, у

остальных же представление было такое: rpm — довольно неплохо, deb — сильно в общих чертах, остальное — тёмный лес.

Было опасение, что информацию и инструментарий, специфичные для дистрибутивов ALT Linux, будет трудно донести до аудитории. Подтвердились ли они?

ALT-специфику (git.alt, gear) знают довольно размыто, однако, имея на руках описание и первого, и второго, удачно додумывают. Очевидно (в том числе некоторым студентам), что при попытке самостоятельного использования всё станет понятнее, встанет на свои места и т.п.

Общий вывод (одним предложением)

Студенты оказались подготовлены гораздо больше, чем ожидалось.

Что вы можете сказать о второй итерации экзамена? Известно, что его сдавал человек, за которым была возможность понаблюдать «до и после»?

Свои наблюдения я изложила в форме письменного отчёта.

Особенности второго экзамена

На втором экзамене присутствовало всего два экзаменуемых.

1. Меньше студентов закапывается глубже: чем больше студентов, тем более могуч коллективный разум.
2. Наличие подопытных кроликов в виде знакомых студентов: есть возможность более подробно изучить мотивацию и последствия курса или экзамена.

Образ студента:

- до подготовки к экзамену с Linux-системами был знаком достаточно поверхностно — когда-то видел, пробовал, но активно не пользовался;
- мотивация сдачи экзамена и чтения лекций — широкий кругозор, который он наблюдал у людей «в теме», которые находились вокруг него;
- подготовка к экзамену — только по конспектам лекций и рекомендованным ссылкам — курс не был прослушан«вживую»;
- руками описанные в курсе действия не делались.

После прочтения лекций студентом перед экзаменом была устроена небольшая консультация. Выводы из консультации — прочтение лекций породило страшную кашу в голове, которая, тем не менее, довольно быстро была разложена по полочкам в ходе консультации. После консультации появилось мышление, похожее на мышление студентов на первом экзамене. Перед подготовкой к экзамену его не было, перед консультацией — были намёки, которые погрязали в общей породившейся каше. Во время консультации студент сам отвечал на задаваемые наводящие вопросы, понимая что к чему, изредка получая комментарии. Синдром «линуксизации мозга» так и не появился.

Результат: несмотря на не идеально сданный экзамен, желание что-то делать и пробовать, разбираться дальше у студента не пропало (как минимум на словах).

Развязка

Не все лекции получились равно плотными: задача избегать излишней конкретики (в плане перечисления названий программных инструментов и их свойств) и вместе с тем описывать реальный процесс создания и сопровождения пакета, очевидно, противоречива по сути.

Последнее мероприятие (показательная сборка пакета) заняло — вместе со всеми заминками и повторами — 45 минут и оставило ощущение жульничества. В самом деле: перед глазами слушателей было «с нуля» настроено сборочное окружение, изрядно модифицирован пакет из Fedora Core, учтена ALT-специфика, исправлено несколько (почти) спонтанных ошибок, собран и отправлен на сборку в хранилище пакет (http://sisyphus.ru/ru/srpm/Sisyphus/nibbles). И при этом ничего недоступного пониманию не произошло!

Кроме того, несколькими экзаменаторами отмечался высокий уровень *изначальной* подготовки студентов (конечно, не всех, а тех, кто решился сдавать этот экзамен). Это отвечает одной важной побочной задачи всего проекта чтения UNIX-курсов вообще: вовлечение в сообщество квалифицированных людей с потенциалом.

Наконец, очевидно, что для успешного объяснения (и освоения) роли сопровождающего достаточно затронуть *азы* довольно *разнообразных видов профессиональной деятельности* — от программирования до team building. Дальнейшее освоение всего спектра может и не понадобиться, и уж точно вполне произвольно.

На этих основаниях можно сделать два вывода:

Оптимистический Сопровождение пакетов в Linux — благодарная для тема для университетов: она не требует от студентов ничего экстраординарного, даёт им в руки начала интересной профессии и может сослужить хорошую службу как сообществу, так и самому университету. Необходимым условием является достаточно широкий кругозор (или желание его достичь).

Провокационный Сопровождение пакетов в Linux — занятие **любительское**. Подобно радиолюбительству, любительскому спорту или любительскому искусству, оно не требует высокого уровня знаний и занятости, однако предоставляет все возможности этот уровень повысить. Мало того, уже на этапе обучения очевидны тенденции, которые могут сделать такого любителя специалистом более высокого класса, чем профессионал, знания которого ограничены сферой его непосредственной деятельности.

Д. А. Пынькин, И. И. Глецевич
Минск, Белорусский государственный университет информатики и радиоэлектроники
http://sisyphus.ru

Проектирование IT-инфраструктуры учебных заведений на базе ОС Linux

Аннотация

В докладе описывается способ организации многоместных систем на базе ОС Linux в режиме без сохранения состояния в учереждениях с публичным доступом к компьютерным системам.

Введение

При создании и настройке систем, предназначенных для эксплуатации в местах с публичным доступом, например в учебных заведениях, всегда возникает проблема поддержки рабочих мест в работоспособном состоянии, так как пользователи сменяют друг друга очень часто.

В докладе показано, как с использованием технологий с открытым исходным кодом и виртуализации обеспечить «независимость» пользовательских сеансов.

Задача проектирования

При проектировании и развертывании систем в общественных учреждениях возникает классическое противоречие между обеспечением удобства работы пользователя и обеспечением надежности и безопасности работы компьютерной системы.

При этом многие действия пользователю «удобно» производить с повышенными привилегиями, но слабо ограничиваемый доступ к системным ресурсам одного пользователя может негативно отразиться на стабильности и безопасности последующей работы всех других пользователей данного компьютера.

Кроме того в учебных заведениях, специализирующихся в области обучения программированию, полноценное выполнение некоторых заданий возможно только при условии полного доступа студента ко всем возможностям операционной системы.

С точки зрения администратора, свободный доступ пользователя к операционной системе приводит к увеличению усилий на поддержание компьютеров в работоспособном состоянии.

Таким образом, вырисовывается следующая техническая задача: предоставить пользователю возможность максимально полного управления компьютерной системой, при этом обеспечить полное восстановление этой системы после завершения сеанса.

Организация бездисковых виртуальных рабочих станций

Основой используемых решений являются бездисковая загрузка операционной системы Linux [1], многоместный режим работы графической подсистемы Linux [2] и возможность некоторых реализаций виртуальных машин работать в так называемом stateless-режиме — все вносимые пользователем во время работы изменения не записываются на жесткий диск, а накапливаются во временных файлах или в оперативной памяти компьютера. Соответственно, достаточно перезагрузить виртуальную машину для полного восстановления системы в исходное работоспособное состояние.

Для эффективного использования оборудования было использовано свойство видеокарт — возможность подключения нескольких мониторов, что, при правильной настройке графической подсистемы, позволило организавать несколько независимых рабочих мест на базе одного системного блока [2].

Фактически было реализовано объединение классической инфраструктуры VDI (Virtual Desktop Infrastructure) и методов терминального доступа к системе. В качестве сервера виртуальных рабочих столов используется тот же компьютер, что и для организации доступа к пользовательским виртуальным машинам! Причем, такой подход пригоден для организации как бездисковых, так и «полноценных» многоместных виртуальных рабочих столов.

В качестве сервера бездисковой загрузки может выступать компьютер под управлением любой операционной системы, при условии обеспечения сервисов:

- DHCP;
- TFTP;
- SMB или NFS (файловый сервер).

От администратора требуется лишь настроить загрузку системы и указать где лежат файлы, необходимые для работы Linux и виртуальных машин. Так, для среднестатистического класса, в котором используются одинаково настроенные системы, необходимы следующие файлы:

- ядро Linux;
- файл начальной инициализации (initrd);
- файл-образ файловой системы Linux;
- файл-образ жесткого диска виртуальной машины.

Чтобы предоставить пользователям возможность использовать различные операционные системы, администратору достаточно подготовить соответствующий виртуальный образ жесткого диска, но при этом необходимо учитывать тот факт, что нагрузка на дисковую подсистему сервера пропорциональна количеству одновременно используемых типов виртуальных машин.

Результат

Описанные принципы создания IT-инфраструктуры позволяют организовать рабочие места без сохранения состояния, которые позво-

ляют пользователю полностью контролировать свою работу, не влияя на работу других пользователей.

Работа администратора может быть сведена к подготовке базового образа виртуальной машины и корректной настройке сети.

Таким образом, вся предлагаемая IT-инфраструктура строится на основе Linux (серверы, сетевые экраны, гипервизоры на рабочих станциях). Пользователям доступны виртуальные машины с уже установленными и настроенными пользовательскими операционными системами.

Литература

[1] *Д.А. Пынькин, И.И. Глецевич, А.В. Отвагин* Использование бездисковых рабочих станций в образовательном процессе БГУИР // Материалы 4 международной конференции «Информационные системы и технологии» IST'2008. — Минск, 2008. — С. 310-315.

[2] *Wikipedia* Multiseat Configuration `http://en.wikibooks.org/wiki/Multiterminal_with_Xephyr`

А. О. Маковецкий
Москва, компания «Asterisk-VoIP»
http://asterisk-voip.ru/

Автоматизация систем информирования и оповещения студентов ВУЗа на базе свободного программного обеспечения

Любое среднее специальное или высшее учебное заведение сталкивается с проблемой оперативного информирования. Обычно решают эту задачу «вручную» или полу автоматизированным способом, доставшимся по наследству от советского прошлого.

Технический прогресс позволяет полностью автоматизировать процесс информирования студентов, абитуриентов, преподавателей, используя решения на базе свободного программного обеспечения.

Основой решения для автоматизации может и должна стать связка СУБД MySQL и ip-pbx Asterisk, программных продуктов работающих под управлением операционных систем семейства Unix. Именно

Asterisk будет осуществлять голосовое информирование абонентов, а MySQL можно использовать для хранения уникальной информации для каждого из абонентов.

Стоит отметить, что многие ВУЗы в последнее время стали активно работать над собственными Web-порталами. Для таких ВУЗов можно использовать одну разделяемую Базу Данных (БД) с персональной информацией о студентах (например, результаты контрольных) как для публикации на портале, так и для голосового информирования.

К сожалению, для большинства ВУЗов невозможно предложить сколько-нибудь интересное, развитое решение, которое было интегрировано с внутренней системой управления студентами/сотрудниками в виду отсутствия оных систем. Очень часты случаи, когда необходимая информация хранится либо на бумажных носителях, либо, в лучшем случае, в excel-таблицах или doc-документах.

Поэтому в решении предполагается использование простой формы для загрузки excel-документа и использование samba-ресурса, с автоматическим его опросом.

Теперь, имея в своем распоряжении массив информации, полученный либо из БД, либо путем разбора загруженных документов, мы можем легко информировать студента. Вся информация может быть разделена на два основных типа — общая и персонализированная. Общая информация может быть любой, и представляться она должна в виде заранее записанного голосового сообщения. При этом, учитывая возраст и опыт работы с ПК работников, запись сообщения может быть реализована при помощи исходящего, либо входящего звонка на телефонный аппарат сотрудника. Если говорить о персональной информации студентов — то, учитывая специфику образовательных заведений, это, в основном, либо некие цифры (оценки), либо даты. Проговаривание, что числовых данных, что даты и времени, легко реализуется на основе «склеивания» отдельно записанных чисел и названия месяцев.

В описанном виде решение для автоматизированного информирования является простым для реализации, но требует высокой степени участия сотрудников в его использовании. При этом возможно несколько усложнив структуру решения, снизить степень вовлечения сотрудников в его эксплуатацию. Для этого возможно использовать свободные TTS-движки, такие как ru_tts и festival. Это позволит ис-

ключить процесс наговорки сообщений и достаточно предоставлять информацию в текстовом виде.

Использование автоматизированной системы информирование должно исключить большинство ошибок донесения информации до получателя за счет исключения посредников и упрощения самой процедуры. К тому же это хороший шаг в развитии инфраструктур ВУЗа, которые зачастую являются наследием советского союза.

В. Кулямин, В. Рубанов, А. Хорошилов
Москва, Институт системного программирования РАН

Проект: Комитет по образованию и высшей школе РАСПО
http://www.raspo.ru

О комитете по образованию и высшей школе Российской Ассоциации Свободного Программного Обеспечения

Аннотация

В докладе представлены цели и задачи недавно созданного комитета по образованию и высшей школе Российской Ассоциации Свободного Программного Обеспечения (РАСПО), а также потенциальные направления его деятельности. Среди этих направлений наболее подробно рассматриваются возможные мероприятия по повышению качества преподавание ИТ-дисциплин за счёт использования преимуществ СПО. В то же время, выступление нацелено не только на открытое обсуждение существующих предложений, но и на поиск новых идей и привлечение наиболее активных участников конференции к работе комитета.

О Российской Ассоциации Свободного Программного Обеспечения (РАСПО) было объявлено в начале 2009 года, когда представители 17 компаний подписали Меморандум о намерении создать эту организацию. 18 мая 2009 года прошло ее официальное учредительное собрание. Компании объединились в Ассоциацию вокруг миссии, заключающейся в содействии разработке, внедрению и популяризации свободного программного обеспечения в государственном, общественном и коммерческом секторах российской экономики, в развитии отечественной индустрии программного обеспечения, основанного на открытом коде и свободной лицензии, и её вхождении в мировой рынок разработки ПО.

В рамках Ассоциации было сформировано несколько профильных комитетов, в том числе и Комитет по образованию и высшей школе. Основными целями этого комитета являются:

- Содействие повышению компьютерной и правовой грамотности населения.
- Содействие повышению качества образования ИТ-специалистов.
- Расширение использования СПО в образовательных учреждениях.
- Привлечение молодёжи в сообщества СПО проектов.

Основными задачами комитета являются:

- Поддержка разработки и публикации методических материалов и учебной литературы по СПО.
- Поддержка разработки и внедрения учебных курсов по СПО.
- Помощь государству при формировании политики внедрения, использования и разработки СПО в учебных заведениях различного уровня.
- Предотвращение недобросовестной конкуренции при госзакупках программного обеспечения для целей образования.
- Пропаганда и содействие внедрению СПО в образовательных учреждениях.
- Содействие разработке СПО для образовательных учреждений.

Применительно к высшей школе в комитете предполагается развернуть работы по следующим направлениям:

- Расширение использования СПО в учебном процессе (применительно как к базовому и офисному ПО, так и к ПО, специфичному для предметной области).
- Повышение качества преподавания ИТ-дисциплин за счёт использования преимуществ СПО.
- Популяризация преподавания и исследования экосистемы СПО.
  - Преподавание принципов разработки СПО, особенностей лицензирования и т.д.
  - СПО-сообщества, СПО-разработка и само СПО как объект изучения социологических, экономических и других наук.
- Привлечение студентов к участию в СПО проектах.

- Расширение использования СПО в хозяйственной деятельности ВУЗов.

Наиболее близким для ИСП РАН является направление по обучению студентов по ИТ-специальностям, поэтому именно на проработке этого направления был сосредоточен первоначальный фокус наших усилий. Черновой вариант предложений по этой тематике был подготовлен в ИСП РАН и будет представлен на конференции.

В этом документе обозначены следующие преимущества использования СПО и проектов по его разработке, позволяющие повысить качество преподавания ИТ-дисциплин в университетах:

- Студенты на практике знакомятся с внутренним устройством и технологиями разработки сложного ПО, повседневно используемого в технических системах, бизнесе, науке и государственном управлении.
- Студенты и преподаватели получают доступ к последним разработкам в соответствующей области и могут без ограничений изучать инновационные технологии и архитектурные решения.
- Студенты и преподаватели приобретают возможность проводить любые полноценные исследования и вести любые разработки на основе последних достижений в интересующей их области ИТ-технологий на базе свободно изменяемого ПО.
- Студенты обретают дополнительную мотивацию для обучения, а преподаватели — для его проведения за счет востребованности получаемых знаний и выполняемой работы на мировом уровне, ощущения причастности к переднему краю ИТ-технологий, а также большей свободы в выборе направления и путей реализации (самообразования и сопутствующих исследований).
- Студенты и преподаватели могут достаточно свободно напрямую общаться с профессионалами высокого уровня в соответствующей области, принимающими участие в открытых проектах по разработке ПО. Это позволяет актуализировать содержание обучения и вырабатывать необходимые для полноценного развития ИТ-специалистов социальные и когнитивные навыки.

Таким образом, использование СПО и открытых проектов позволяет существенно повысить качество ИТ-образования за счет следующих факторов.

- **Приведения его содержания в соответствие с требованиями дальнейшего экономического и социального развития**.
- **Значительного увеличения исследовательской составляющей в работе преподавателей,** без которой полноценное современное ИТ-образование невозможно.
- **Актуализации проводимых исследований** с помощью постоянного соотнесения их с важнейшими текущими проблемами ИТ, выделяемыми мировым исследовательским сообществом, активно участвующим в разработке СПО и открытых проектах.
- **Социальной адаптации** студентов в среду единомышленников и профессионалов, не ограниченную рамками учебной группы и ВУЗа.

Для реализации этих возможностей предлагается инициировать следующие работы (безусловно с публично доступными результатами):

- Доработка существующих и разработка новых *образовательных материалов* различных типов (общетеоретических и специализированных курсов, образовательных модулей, практикумов, лабораторных работ и наборов упражнений, учебников и практических пособий, методических материалов, материалов для тренингов, дополнительного образования, дистанционного и заочного обучения), использующих доступные материалы открытых проектов и СПО.
- Доработка существующих и разработка новых *специализированных курсов* по архитектурным решениям и технологиям, использованным в рамках достаточно зрелого и широкого используемого СПО высокой сложности (операционные системы, компиляторы, системы управления базами данных, телекоммуникационные системы, ПО промежуточного уровня и пр.).
- Стимулирование выполнения *дипломных работ*, результаты которых интегрируются в существующие проекты по разработке СПО или оформляются как полноценные независимые СПО проекты. Например, одним из видов стимулирования может быть организация публичных конкурсов таких дипломных проектов.

Мы осветили лишь первоначальные идеи для организации деятельности комитета. Однако, спектр потенциальных активностей гораздо

шире. Поэтому, в заключение, мы хотели бы пригласить все заинтересованные стороны к обсуждению рассмотренных предложений, возможной тактики и стратегии их реализации, а также к поиску новых идей и к активному участию в работе комитета.

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг,
Я. Н. Зайдельман, А. В. Карпов, Е. В. Святушенко,
Н. М. Субоч, Д. В. Хачко, В. В. Яковлев
Москва, Пущино, Переславль, НИИСИ РАН, ИМПБ РАН, Университет г. Переславля

Проект: КуМир http://www.niisi.ru/kumir, http://lpm.org.ru/kumir

Система КуМир — новые возможности

Аннотация

КуМир (Комплект Учебных Миров) — система, ориентированная на преподавание информатики в средней школе и вводных курсов программирования в высшей школе. Система была разработана в 80-х — 90-х годах XX века коллективом студентов, аспирантов и сотрудников мехмата МГУ и Академии наук СССР, работала на всех мыслимых и немыслимых компьютерах, установленных в школах СССР, и получила широкое распространение. С течением времени интерфейс КуМира (под операционнной системой МС ДОС) устарел. Современная новая реализация системы КуМир является свободно распространяемым программным продуктом и функционирует под управлением операционных систем Linux, Mac OS, Windows.

Система Кумир разработана и распространяется свободно на условиях лицензии GNU 2.0. Данная лицензия разрешает бессрочно использовать КуМир на любом количестве компьютеров в любых целях без оформления каких либо дополнительных документов.

В системе КуМир используется школьный алгоритмический язык с русской лексикой и встроенными исполнителями Робот и Чертежник, что соответствует учебнику [1]. Система КуМир отличается от производственных систем программирования на порядок более высоким уровнем дружественности при отладке небольших учебных программ. Возможности системы КуМир были представлены в докладе [2].

В версии конца 2009 г. в КуМире появились три новых возможности.

Наиболее важной является разработка и реализация стандарта, обеспечивающего удобное подключение новых исполнителей. В соответствии с этим стандартом реализованы исполнители, соответствующие учебникам [3, 4]. Таким образом, система КуМир стала открытой для поддержки вновь разрабатываемых учебников. Важным методически элементом стандарта является возможность непосредственного, «пультового» управления любым подключенным исполнителем. При этом команды, отданные с пульта, запоминаются и могут быть преобразованы в программу на языке КуМир. Подробнее о разработке новых исполнителей и работе с ними см. [5].

Другой возможностью является система поддержки гипертекстовых учебников, которая позволяет вставлять в стандартный гипертекст кнопки, позволяющие читателю выполнять различные действия по управлению КуМиром (загрузка программы, ее выполнение и пр.). В настоящее время в этой среде реализованы отдельные главы из учебника [1] и краткое руководство по использованию системы КуМир.

Наконец, разнообразные справочные материалу по языку, системе КуМир и исполнителям разделены на две части. Наиболее важная информация (синтаксис управляющих конструкций, перечень используемых типов и операций, системы команд встроенных исполнителей и др.) генерируется по программному коду, и тем самым всегда соответствует используемой версии системы. Долговременная информация, наоборот, отделена от системы, и может редактироваться и просматриваться любыми стандартными средствами (на сегодня используется формат pdf).

В 2009 г. система КуМир использовалась как во внешкольном преподавании (Зимняя Пущинская школа, Калужская летняя школа), так и в стандартном курсе информатики в ряде средних школ, а также в ВУЗах (механико-математический факультет МГУ, Университет г. Переславля). Этот опыт показал, что надежность системы КуМир достаточна высока и система может использоваться в регулярном учебном процессе. Авторы благодарят Д. Кириенко и В.Тарасову за использование системы КуМир при преподавании курса информатики в средней школе и многочисленные полезные советы и замечания.

Литература

[1] Информатика : 7-9 кл.: Учеб. для общеобразоват. учр. / А.Г. Кушниренко, Г.В. Лебедев, Я.Н. Зайдельман. — 4-е изд., стер. — М.: Дрофа, 2003. — 335 с.

[2] КуМир вернулся: обучение основам программирования спомощью системы КуМир / А. Г. Кушниренко, А. Г. Леонов, А. В. Карпов, М. А. Ройтберг, Н. М. Субоч, Д. В. Хачко, В. В. Яковлев. — «Свободное программное обеспечение в высшей школе», сборник тезисов, М.: AltLinux, 2009, с. 15

[3] Информатика : алгоритмика : Учеб. для 6 кл. общеобразоват. учр. / А.К. Звонкин, С.К. Ландо, А.Л. Семенов. — М.: Просвещение, 2006. — 237 с.

[4] Информатика: алгоритмика : Учеб. для 7 кл. общеобразоват. учр. / С.К. Ландо, А.Л. Семенов, М.Н. Вялый. — М.: Просвещение, 2008. — 207 с.

[5] Новые Миры в системе КуМир / Кушниренко А.Г., Леонов А.Г., Ройтберг М.А., Хачко Д.В., Тарасова В.В., Яковлев В.В. — «Свободное программное обеспечение в высшей школе», настоящее издание

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, В. В. Яковлев

Проект: ПиктоМир http://lpm.org.ru/kumir/

ПиктоМир — программирование для дошкольников

Аннотация

Программирование — одна из грамотностей, которой должен будет обучиться любой землянин, родившийся в 21 веке. По мнению авторов, это обучение можно и полезно начинать в дошкольном возрасте, до устойчивого овладения навыками чтения и письма. ПиктоМир — техническое средство, полезное при обучении началам программирования дошкольников, не умеющих или не любящих читать и писать. Методика обучения программированию с использованием этого средства находится пока в процессе разработки.

ПиктоМир — среда для подготовки и выполнения программ, которые управляют некоторым *Исполнителем*, имеющим определенную систему команд и действующим в определенной обстановке. Программы

составляются из пиктограмм, подобно тому, как составляются слова и короткие фразы в «кассе букв и слогов» при обучении чтению и письму. Каждая пиктограмма отображает либо команду исполнителя, либо управляющую конструкцию, либо вызов подпрограммы. Первые опыты показывают, что работа в ПиктоМире доступна детям 4-6 лет и вызывает у них интерес. Чтобы удержать внимание и интерес обучаемого в начальном курсе программирования, желательно построить обучение так, чтобы составление и отладка первых программ заняли бы возможно меньшее время. В курсе программирования на механико-математическом факультете МГУ, который один из авторов данного доклада начал читать в 1979 году, эта задача решалась так: первые программы составлялись студентами из заранее подготовленных перфокарт. На перфокартах были подготовлены команды исполнителя Путник, команды вызова и завершения подпрограмм (с заранее выбранными фиксированными именами), цифры от 0 до 9 и «открывающие» и «закрывающие» скобки управляющих конструкций «цикл N раз», «если то» и «цикл пока». Такая методика позволяла обучаемому, впервые столкнувшемуся с программированием, отладить первую элементарную программу в начале 45 минутного занятия и вполне содержательную программу с управляющими конструкциями к концу занятия. Единственным недостатком «сборочной» методики оказались достаточно часто допускаемые обучаемыми «синтаксические» ошибки, состоящие в «разбалансировке» открывающих и закрывающих скобок управляющих конструкций. ПиктоМир построен по аналогичному принципу – программа собирается из готовых пиктограмм, однако ПиктоМир спроектирован так, что синтаксические ошибки в нем принципиально невозможны. Толчком к разработке ПиктоМира послужил успех шестилетней внучки одного из авторов в освоении замечательной Flash-игре LightBot (`http://armorgames.com/play/2205/light-bot`). В данной игре ребенок должен составлять небольшие линейные последовательности пиктограмм, которые предписывают Роботу обойти поле и зажечь свет в отмеченных клетках. Система ПиктоМир является развитием игры LightBot: кроме подпрограмм в ПиктоМире можно использовать повторители, условия и циклы. ПиктоМир — система, позволяющая освоить некоторый замкнутый набор фундаментальных понятий. Вместе с тем, ПиктоМир это система ограниченного назначения, предполагается, что ребенок будет работать в ПиктоМире не более нескольких часов. Курс обучения началам программирования

в системе ПиктоМир будет состоять из нескольких модулей-игр, в каждом из которых нужно будет выполнить последовательность заданий возрастающей сложности. Разные модули могут использовать разные «Роботы-исполнители», однако управляющие конструкции во всех модулях задаются одинаково. В начальной версии системы реализован только один исполнитель. ПиктоМир разрабатывался, как «младший брат КуМира», в ПиктоМире имеются средства превращения разработанной Пикто-программы в КуМир-программу. Начальная версия Пиктомира может загружена в составе ежевечерней сборки КуМира с сайта `http://lpm.org.ru/kumir`.

Краткое описание системы

Интерфейс ПиктоМира состоит из трёх основных компонент:

1. Область наблюдения за исполнителем;
2. Область составления программы;
3. Полка с командами.

Составление программ

«Написание» программ сводится в выбору соответствующих команд-пиктограмм на полке и переносу их в область составления программы, которая представляет собой упорядоченную последовательность пиктограмм.

В программе выделяются отдельные блоки, которые можно трактовать как тела функций, циклов или условий. Мы сознательно отказались от идеи «вложенных конструкций», считая их слишком громоздкими и сложными для детей дошкольного возраста. Пиктограммы на полке разделены на три категории: условия (в форме ромба), повторения (круглой формы) и простые команды. Рядом с блоками программ находятся две специальные клетки: одна ромбовидной формы, в которую можно ставить условие, при котором будет выполняться блок, а другая — круглой формы, в которую можно вставить одну из пиктограмм повторения. Вызов блока вызывается одной из соответствующих пиктограмм К1. . .Кn. Таким образом, всего лишь помещая в нужные клетки соответствующие пиктограммы, можно составлять программы любой сложности, не объясняя детям про вложенность конструкций.

При разработке системы ПиктоМир мы также учитывали эргономические особенности интерфейса, предназначенного для дошкольников. Перетаскивать пиктограммы в стиле *Drag'n'Drop* дошкольникам неудобно и может даже оказаться вредным, так как требует долговременного статического усилия мышц кисти. Поэтому мы реализовали перетаскивание в стиле старых компьютерных игр: первый клик — *выбор объекта*, второй клик – *выбор конечной точки* его перемещения.

Исполнитель

В начальной версии ПиктоМира реализован только один исполнитель. Мы нарисовали робота, похожего на R2D2 из фильма «Звёздные воины», а саму обстановку Робота сделали псевдотрёхмерной (изометрической). Этот Робот умеет двигаться вперед, поворачивать направо и налево и закрашивать клетку, на которой он находится.

Текущее состояние разработки

На данный момент ПиктоМир проходит апробацию на некоторых детях-добровольцах (и их родителях!). По результатам этой апробации будут составлены модули-игры, о которых говорилось выше. В процессе аппробации предусмотрено создание различных миров, создаваемых и подключаемых по стандарту, разработанному для системы КуМир.

Среднесрочная задумка

Первые эксперименты с ПиктоМиром показали очевидную вещь: любой деятельностью дети охотнее занимаются *в компании*, а не *в одиночку*. На настоящий момент единственная опробованная совместная деятельность, это обучение одного (неопытного младшего) ребенка другим (старшим и более опытным). К лету 2010 мы планируем включить в ПиктоМир средства организации соревнований двух или более детей.

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, В. И. Хачко, Д. В. Хачко, В. В. Тарасова, В. В. Яковлев
Москва, Пущино, Переславль, НИИСИ РАН, ИМПБ РАН, СШ №1 г.Переславля

Проект: КуМир http://www.niisi.ru/kumir, http://lpm.org.ru/kumir

Новые Миры в системе КуМир

Аннотация

КуМир (Комплект Учебных Миров) — система, ориентированная на преподавание информатики в средней и высшей школе (вводный курс). «Мир» — это открытое множество «обстановок» определенного вида и исполнитель, который действует в этих обстановках. Описаны средства создания новых миров в системе КуМир и три мира, поддерживающих преподавание информатики по учебникам «Информатика:алгоритмика» для 6-го и 7-го классов средней школы.

Общие сведения

«Мир» — это открытое множество «обстановок» определенного вида и исполнитель, который действует в этих обстановках. Два мира (Робот и Чертежник) встроены в систему КуМир [1, 2]. В КуМире был предусмотрен механизм создания новых исполнителей. Однако практика использования КуМира в учебном процессе потребовала создания нового механизма, который делает работу с внешними исполнителями практически неотличимой от работы с встроенными исполнителями Роботом и Чертежником. Это связано с необходимостью длительного использования некоторых внешних исполнителей в учебном процессе.

Как правило, подобно Роботу, мир допускает непосредственное управление исполнителем с помощью пульта. С помощью стандартных средств поддерживается возможность преобразования последовательности команд пульта в неветвящуюся программу и передача этой программы в систему КуМир.

Разработка такого нового мира ведется независимо от системы КуМир. Исполнитель должен быть написан на C++ с использованием библиотеки Qt[3] и собран в виде динамически подгружаемой библиотеки. Исполнители подгружаются в момент старта системы КуМир. Взаимодействие Кумира и Миров организовано через класс QPlugin.

Мир может использоваться (без изменения исходного кода) и автономно от системы КуМир. Для этого в комплекте поставки есть специальная программа, она подгружает библиотеку мира. Управление миром при автономном использовании осуществляется при помощи пульта управления. При этом сохраняется полная функциональность мира (кроме, естественно, работы под программным управлением из системы КуМир и передачи последовательности выполненных команд в КуМир).

Примеры Миров

В поставку КуМир 1.7 входят три новых Мира, предназначенных для поддержки учебников [4, 5]: Черепаха, Водолей и Кузнечик. Миры прошли апробацию в учебном процессе в школе №1 г. Переславль-Залесский. Каждый мир имеет пульт управления. Помимо этих трех Миров существует демонстрационная версия управления роботом LEGO NXT.

Черепаха — реализация знаменитого исполнителя, предложенного С.Пейпертом . Мир Черепахи — квадратный песчаный остров, окруженный водой. Черепаха передвигается по нему (прямо и делая повороты). При этом хвост черепахи может оставлять на песке след. Наша реализация детально следует исходному описанию. В частности, черепаха рисует именно хвостом (а не серединой живота, как это часто делают).

Кузнечик — исполнитель, описанный в учебнике [4]. Мир Кузнечика — числовая прямая, на которой отмечены целочисленные точки. Пользователь может менять масштаб и протягивать прямую через окно наблюдения с помощью стандартных средств. Кузнечик может «прыгать» по числовой оси вперед и назад (длины этих прыжков могут быть различны). В отличие от Черепахи, Кузнечику может быть выдано определенное задание. В нем указываются длины прыжков, точки, куда нужно попасть (они отмечаются флажками) и, возможно, описание фрагмента оси, который доступен Кузнечику. Пользователь может создать новое задание, сохранить его, загрузить и отредактировать.

Водолей — это мир известных задач-игр про переливание. В этих задачах даны три стакана, объем каждого стакана — целое число (если объем равен нулю, стакан не показывается). У игрока есть следующие возможности:

1. долить нужный стакан доверху («из крана»);
2. вылить всю воду из указанного стакана (при этом стакан становится пустым);
3. перелить воду из одного стакана в другой (если удается перелить всю воду, то первый стакан становится пустым; в противном случае второй стакан становится полным, а остаток остается в первом стакане).

Цель игры — получить в каком-либо из стаканов указанное количество воды (начальное количество воды в каждом стакане также задано). Как и в случае Кузнечика, пользователь может работать с различными заданиями (в задании Водолея описываются объемы стаканов, начальное количество воды в них и требуемое количество воды).

LEGO NXT — Реализация управления роботом Lego NXT из системы Кумир. Подключение к роботу осуществляется через интерфейсы Bluetooth или USB.

Доступны следующие команды:

1. повернуться на требуемый угол;
2. сместиться вперед;
3. сместиться назад;
4. проверить показания датчика касания.

Литература

[1] Информатика : 7-9 кл.: Учеб. для общеобразоват. учр. / А.Г. Кушниренко, Г.В. Лебедев, Я.Н. Зайдельман. — 4-е изд., стер. — М.: Дрофа, 2003. — 335 с.

[2] КуМир вернулся: обучение основам программирования спомощью системы КуМир / А. Г. Кушниренко, А. Г. Леонов, А. В. Карпов, М. А. Ройтберг, Н. М. Субоч, Д. В. Хачко, В. В. Яковлев. — «Свободное программное обеспечение в высшей школе», сборник тезисов, М.: AltLinux, 2009, с. 15

[3] Qt 4: Программирование GUI на C++ / Бланшет К., Саммерфилд М. – изд. Кудиц-Пресс, 2007. — 641 с.

[4] Информатика : алгоритмика : Учеб. для 6 кл. общеобразоват. учр. / А. К. Звонкин, С. К. Ландо, А. Л. Семенов. — М.: Просвещение, 2006. — 237 с.

[5] Информатика: алгоритмика : Учеб. для 7 кл. общеобразоват. учр. / С. К. Ландо, А. Л. Семенов, М. Н. Вялый. — М.: Просвещение, 2008. — 207 с.

Д. А. Кузнецов
Нижний Новгород, Нижегородский Радиотехнический Колледж
http://fireforge.net/projects/cunewebform,
http://fireforge.net/projects/elkartoteka

Разработка программного обеспечения для работы учебного заведения на базе СПО

Аннотация

Многие задачи в учебном заведении решаются с помощью автоматизированных систем. Но за частую, сложно найти программный продукт для выполнение специфических задач подразделений. В этих случаях хорошим решением может стать разработка собственного приложения для решения именно наших задач. В этом случае свободное программное обеспечение предоставляет незаменимые инструменты и возможности для быстрой разработки и дальнейшего функционирования созданной системы.

В чем преимущества использования Свободного Программного Обеспечения при разработке приложений для внутреннего использования?

- Благодаря отсутствию ограничений на количество устанавливаемых копий и доступности программного обеспечения всегда актуальной новой версии разрабатываемые системы при необходимости легко масштабировать, переносить на обновленные сервера, дорабатывать, исправляя возникающие ошибки и внедряя новый требуемый функционал.
- В легальном свободном доступе находятся как средства разработки, так и средства, на которых в конечном счете функционирует программный продукт.
- Нет никаких препятствий в одновременном функционировании программного продукта на коммерческий платформах, если свойства кросплатформенности или кросбраузерности заложены в техническом задании на проектирования того или иного продукта, который, как правило, является курсовой иди дипломной работой студентов.

Особенности лицензирования программ по лицензии GNU/GPL при использовании кода под другими свободными лицензиями

При создании свободного программного обеспечения, лицензируемого под GNU/GPL очень часто используются готовые библиотеки, которые имеют свободную лицензию, отличную от GNU/GPL, например MIIT-лицензию, или BSD-образную лицензию, или LGPL лицензию, некоторые пункты которых противоречат принципам GNU/GPL или наоборот. Как поступить в таких случаях, когда все-таки хочется залицензировать программу под GNU/GPL? Анализируя опыт использования программистами ФГОУ СПО «НРТК» программных компонентов под различными лицензиями и правомерность этого использования можно сделать вывод, что совмещать подобные лицензии можно совершенно свободно, соблюдая лишь некоторые правила...

Опыт разработки приложений для внутренней работы Нижегородского Радиотехнического Колледжа

Еще до начала дискуссий о свободном программном обеспечении преподавателями информационных технологий и специалистами колледжа была сделана ставка на обучение студентов сознанию распределенных информационных систем на базе клиент-серверных web-технологий. Это дало и дает по сей день очень много преимуществ:

- централизованное управление данными посредством использования сервера приложений с возможностью резервного копирования данных;
- отсутствие необходимости установки каких-либо приложений на компьютеры пользователей, за исключением web-браузера, без которого немыслима ни одна современная операционная система;
- единое информационное пространство для всех подразделений колледжа.

Все эти преимущества очевидны, т. к. такие приложения являются по умолчанию кросплатформенными.

После внедрения свободного программного обеспечения в профильные предметы и после изучения средств разработки графических «тяжелых» приложений у преподавателей и студентов стал стремительно расти интерес к кросплатформенной разработке графических приложений с применением библиотек QT и RAD/IDE

QTCreator. Это новый этап для совершенствования знаний в области применения новейших технологий, которые в большинстве своем базируются на свободном программном обеспечении, потенциал которого становится очень быстро виден студенту, если он учится в Нижегородском радиотехническом колледже.

И. Чубин
Киев, Учебный центр Сетевые Технологии

Виртуальные сети как часть живой документации

Аннотация

В работе рассматривается новый способ подготовки документации, который может помочь облегчить процедуру её написания и сделать её использование более эффективным. Особое внимание уделяется виртуальным компьютерным системам и сетям, как важному вспомогательному элементу, использующемуся как при подготовке документации, так и при её использовании.

Введение

В мире UNIX/Linux-систем и систем с открытым исходным кодом получили широкое распространение руководства, представляющие собой пошаговое описание процедуры решения какой-либо задачи, то есть рассказывающие о том, как "сделать что-то", что и отражено в названии документов такого класса — HOWTO.

В общем виде документ HOWTO содержит:

1. Описание системы, в которой выполняется работа;
2. Описание её исходного состояния;
3. Описание её конечного состояния;
4. Описание процедуры перехода из начального состояния в конечное.

Как правило, все перечисленные выше пункты представлены как простое словесное описание, иногда сопровождающимся прилагаемыми конфигурационными файлами и текстом программ и скриптов.

Обычная документация является линеаризованным представлением некоторой процедуры настройки. И ключевые пункты настройки,

и несущественные детали находятся на одной и той же прямой, и выделены только изобразительным талантом автора. Автор всегда должен находить компромисс между степенью детализации руководства и перегруженностью деталями. Принимая решение о том, что должно быть упомянуто в руководстве, а что является несущественным, автор основывается на своём представлении об идеальном читателе документа, который не всегда соответствует реальному. В результате, при прочтении руководства конкретным читателем в конкретном случае оказывается, что некоторой информации ему не хватает, при том, что есть много лишнего.

Описание исходной системы тоже не всегда идеально. Как правило, для UNIX/Linux-систем и сетей оно выглядит как словесное описание того, какие операционные системы используются, и какое программное обеспечение установлено.

Описание системы, рассматриваемой в документации, в идеальном случае должно включать в себя:

- Множество узлов и взаимосвязь между ними;
- Программное обеспечение узлов, версии операционных систем и дополнительного программного обеспечения;
- Процедуру начальной настройки операционной системы и дополнительного программного обеспечения.

Полное документирование всех этих параметров вручную является сложной и не всегда выполнимой задачей, в особенности, если задача решается не в свежеустановленной системе.

Для того, чтобы исключить влияние неучтённых факторов, проведение экспериментов при написании документации часто выполняется в свежеустановленных системах внутри виртуальных машин.

Система или сеть, описываемая в руководстве, формируется, а затем внутри неё производятся действия, которые должны быть описаны в документации. После того как документация сделана, сеть или уничтожается, или оставляется для продолжения работы над следующими версиями документа. В последнем случае, сохраняются, как правило, образы виртуальных машин, участвовавших в написании документации.

У читателя, работающего с HOWTO, есть две альтернативы:

1. Он может просто читать текст;

2. Он может создать описываемую в документации модель с помощью виртуальных машин, и выполнять действия, описанные в документации в этой модели.

Живая документация

Как видно из описанного выше, при написании документации значительная часть работы при составлении документации является вспомогательной, а большая часть информации доступной при написании остаётся недоступной читателю, даже если она ему нужна.

Для получения интересующих его, но отсутствующих в документе деталей, читателю остаётся только один вариант: развернуть систему самостоятельно, и проделать всю процедуру по настройке самому.

Документ можно сделать намного более информативным, если дополнить его всей промежуточной информацией, доступной в ходе настройки, а также формальным описанием модели сети, в которой она выполняется.

При использовании документации, подготовленной таким образом, у его читателя появляются дополнительные возможности:

1. Получение дополнительных деталей описываемых процедур;
2. Запуск и исследование описываемой модели системы;
3. Перенос настроек из модели в настоящую систему.

Для этого необходимо несколько изменить процедуру подготовки документации и использовать при её написании дополнительные инструменты.

Процедура создания документации:

1. Описание модели системы, подлежащей настройке. Создание формального описания, достаточного для развёртывания виртуальной системы, описываемой в документации.
2. Создание модели системы, подлежащей настройке. То есть собственно создание виртуальных машин и организация связей между ними.
3. Настройка и запись.

   Собственно выполнение всех действий, которым посвящён документ. Все действия и результаты их выполнения автоматически фиксируются и впоследствии являются доступными читателю.

Текстовый документ HOWTO составляется на основе этих записей. Причём, существует формальная связь между текстовым документом, и выполненными операциями по настройке. Например, в документе встречается некая команда. Мы можем посмотреть, когда она выполнялась; какие действия предшествовали ей; каковы были результаты её выполнения.

4. Редактирование процедуры настройки; добавление описания.

   На основе сделанных записей составляется документация, которую будет читать пользователь.

5. Публикация документации.

Быстрое построение виртуальных сетей

Одним из важнейших элементов системы, описываемой выше, является легковесная процедура развертывания виртуальной сети, то есть разработать способы и средства для построения виртуальной сети по описывающему ее конфигурационному файлу.

Исследуемые сети могут быть не только чисто виртуальными, но и смешанными: наряду с виртуальными узлами в них могут работать и обычные компьютеры и сетевые устройства. Виртуальная часть сети может работать поверх хоста, множества распределенных хостов или в облаке виртуализации, таком как EC2.

Для построения виртуальных сетей может использоваться существующее уже в настоящий момент программное обеспечение, например:

- Xen или KVM как системы виртуализации;
- QEMU/PEMU/Dynamips как эмуляторы сетевых устройств;
- openvpn или vde как средство для объединения распределенных систем виртуализации;
- элементы ядра Linux (модули 8021q, bridge и ряд других), которые обеспечивают объединение виртуальных устройств между собой требуемым способом.

Программное обеспечение, позволяющее управлять всей виртуальной сетью в целом (создавать узлы, изменять топологию сети, вносить изменения в показатели качества работы сетевых соединений, такие как задержка, пропускная способность, вероятность потери трафика),

и отдельными ее виртуальными подсистемами, разработано специально для решения описанной задачи.

Оно же может применяться для автоматизации процесса начальной настройки систем, то есть в ходе применения процедур настройки, о которых шла речь выше. Как правило, эти процессы содержат много рутинных операций, которые всегда хочется переложить на компьютер – например, такие как настройка IP-адресов, маршрутизации, начальных учетных записей, паролей для доступа к ним, различных интерфейсов для сетевого доступа и множество других.

Основные задачи:

- быстрое разворачивание и начальная настройка произвольных виртуальных и смешанных сетей;
- предоставление инструментария для моделирования разнообразных условий работы сети (задержка, пропускная способность, вероятность потери трафика);
- фиксация результатов экспериментов.

Ключевые возможности системы:

- унификация управления;
- визуализация сети;
- интегрированные инструменты для исследования работы сети;
- поддержка множества операционных систем и виртуальных сетевых устройств;
- интеграция виртуальных и реальных устройств;
- распределенность;
- поддержка бездисковых систем;
- отказоустойчивость;
- возможность исполнения внутри облака виртуализации.

Сети могут предназначаться для решения как учебных и исследовательских, так и производственных задач.

Дополнительная информация

- http://xgu.ru/wiki/LiLaLo – запись операций в консоли UNIX-системы
- http://xgu.ru/wiki/Xentaur – программное обеспечение для организации виртуальных сетей

И. П. Русинов, И. А. Нечаев
Кострома, КГУ им. Н. А. Некрасова
Проект: KSU Cluster

Свободное программное обеспечение для научных исследований по квантовой теории конденсированных сред

Современные исследования в области квантовой теории конденсированных сред не могут обходиться без привлечения большого количества вычислительных ресурсов, а применение последовательных алгоритмов для решения задач, стоящих перед теоретиками, становится бесперспективным. Поэтому актуальной является разработка программных комплексов для высокопроизводительных вычислительных систем, позволяющих использовать преимущества, обусловленные параллельностью процесса вычисления. Сегодня силами различных групп теоретиков и программистов из разных стран Европы и Америки развивается целый спектр программных комплексов, разрабатываемых как свободное программное обеспечение (ПО) и позволяющих на высоком уровне проводить научные исследования в рамках таких направлений, как, например, нанофотоника, органическая электроника, спинтроника. Лидерами среди такого рода программных продуктов, на наш взгляд, являются следующие проекты: Abinit [1], SIESTA [2], Quantum-Espresso [3] и OpenMX [4].

Каждый из указанных проектов по-своему реализует подход теории функционала электронной плотности к квантовомеханическому описанию периодических твердых тел и наноразмерных материалов. Наряду со свойствами основного состояния, они позволяют изучать различные свойства отклика системы (Abinit, Quantum-Espresso), исследовать возбужденные состояния (Abinit), транспорт электронов (OpenMX, SIESTA), а также проводить расчеты по структурной релаксации и молекулярной динамике. Также эти продукты являются полем не только для научных расчетов, но и средством для обучения специалистов.

В отличие от привычного пользователям программного обеспечения, подобное специфическое ПО приходится поддерживать администраторам кластера самостоятельно, так как невозможно установить

ранее скомпилированный мэнтейнером RPM пакет, даже в случае существования их в репозитории. Для функционирования ПО необходимо заниматься его сборкой на том же самом кластере, где программные пакеты будут загружены в память для выполнения. Причиной этого является крайне широкий простор для скомпилированных расчетных библиотек на каждом отдельном кластере.

Указанное выше ПО было установлено и в настоящее время используется на вычислительном кластере КГУ им. Н. А. Некрасова. Кластер имеет 5 узлов по 8 ядер в каждом. Таким образом, максимальное число параллельных процессов для задачи равняется 40. Пиковая производительность всего кластера составляет 480 ГФлопс. На кластере используется дистрибутив Suse Linux Enterprise Server 10.0 sp1 (SLES 10). Этот дистрибутив установлен на каждом из узлов кластера. В качестве стандартных математических библиотек установлена Intel Math Kernel Library [5], в которую включен их исходный код.

Для визуализации полученных результатов нами широко используется такое свободное ПО, как XCrySDen [6] и GnuPot [7]. Первая программа (XCrySDen) позволяет визуализировать кристаллические и молекулярные структуры, отображать изоповерхности и контуры, которые могут быть наложены на эти структуры. Вторая программа (GnuPot) позволяет строить любые виды двумерных и трехмерных графиков. Возможна визуализация как при помощи *eps* файлов, так и терминала. Особенно интересен язык сценариев для создания различного вида графиков. Поддерживают различного рода аппроксимации для точечных графиков.

В заключении отметим также, что совместно с сотрудниками *Donostia International Physics Center (San Sebastian, SPAIN)* нами ведется разработка собственного ПО для высокопроизводительных систем, позволяющего проводить исследования квазичастичных свойств в двух- и трехмерных электронных системах как в рамках модели желе, так и в рамках более реалистичных моделей, учитывающих особенности зонной структуры рассматриваемых систем. В качестве последних при описании поверхностных эффектов выступает метод повторяющихся пленок, характеризуемых одномерным псевдопотенциалом, а при изучении динамики квазичастиц в объемных периодических структурах за основу берется первопринципный метод линеаризованных muffin-tin орбиталей [8].

Литература

[1] www.abinit.org
[2] www.icmab.es/siesta/
[3] www.quantum-espresso.org/
[4] www.openmx-square.org
[5] software.intel.com/en-us/intel-mkl/
[6] www.xcrysden.org/XCrySDen.html
[7] www.gnuplot.info
[8] www.fkf.mpg.de/andersen/docs/manual.html

М. Ш. Исламов
Ижевск, Удмуртский Государственный Университет
http://refal51.googlecode.com

Динамический Рефал — инструмент для обучения сентенциальному программированию

Аннотация

Сентенциальное программирование в последние годы занимает всё большую популярность на рынке информационных технологий. MDX, Linq, JQuery — одни из самых наглядных примеров, подтверждающих это. Динамический Рефал — предлагаемый автором открытый инструмент для обучения сентенциальному программированию в высшей школе. Динамический Рефал — это расширение языка Рефал средствами, позволяющими более выразительно и эффективно писать программы и обучаться данному стилю.

Сентенциальный стиль программирования [1] является подвидом декларативного стиля. Другими словами, программы, написанные на сентенциальном языке, представляют собой описания фактов предметной области и постановку задачи, а ее решение исполнитель (ЭВМ) находит самостоятельно.

Актуальность данного направления в последние годы только растет. Всего несколько лет назад появился MDX — язык запросов для простого и эффективного доступа к многомерным структурам данных (используется в технологиях OLAP), AJAX-библиотека JQuery — для гибкой работы с web-интерфейсом (и не только), Linq– в .NET для быстрой работы с коллекциями, базами данных, xml-документами и

другими источниками. Все эти и многие другие сентенциальные инструменты ускоряют разработку программ за счет того, что для решения задачи от программиста требуется лишь описание своих требований на простом и понятном языке.

С ростом популярности данной парадигмы на практике, растет необходимость ее более углубленного изучения в высшей школе. Уже сейчас в некоторых высших заведениях студенты знакомятся с сентенциальным стилем, изучая регулярные выражения и SQL, но более традиционным способом знакомства с этой парадигмой является изучение языков написания полноценных программ, а не отдельных запросов.

Такие языки существуют с начала времён программирования для ЭВМ (Prolog, Alain Colmerauer, 1972 г.; Рефал, Валентин Турчин, 1966 г. [2]), однако внимание мирового сообщества до последнего времени было приковано преимущественно к императивному стилю и ООП, в результате чего эти языки значительно отстали от представителей других парадигм. Так, например, изучение Рефала позволяет студентам быстрее осваивать регулярные выражения (RegExp), однако в задачах обработки строк текста гибкость и выразительность регулярных выражений значительно превосходит шаблоны Рефала. Возможно, пришло время уделить больше внимания этой проблеме.

В результате проделанной работы, автором был спроектирован и описан диалект Рефала (Динамический Рефал [3]) для обучения сентенциальному стилю программирования, среди прочего включивший в себя новые типы данных и переменных, а так же некоторые конструкции регулярных выражений для представления образцов. Экспериментальная реализация данного диалекта в виде интерпретатора доступна по адресу `http://refal51.googlecode.com` и распространяется по лицензии LGPL.

В ходе работы были определены следующие задачи:

- расширить возможности управления сопоставлением в Рефале с помощью декларативных, а не императивных конструкций;
- предоставить пользователю механизм абстракции данных с помощью пользовательских типов переменных;
- все расширения должны быть концептуально непротиворечивы Базисному Рефалу;

Полученный в результате проделанной работы диалект не ограничивается применением только в образовательных целях. По мощности

и выразительности Динамический Рефал не уступает многим сентенциальным инструментам, включая Рефал-5, а значит может применяться на практике.

Литература

[1] Непейвода Н. Н., Скопин И. Н. Основания программирования. М.; Ижевск, 2003.

[2] Турчин В. Ф. РЕФАЛ-5. Руководство по программированию и справочник. http://www.refal.net/rf5_frm.htm

[3] Исламов М. Ш. Некоторое расширение языка Рефал. 2009. http://ulm.udsu.ru/~soft/wiki/lib/exe/fetch.php?media=ubs2009_direfal.pdf

А. Н. Пустыгин, О. И. Суворов, И. С. Ермолаев, Д. С. Ботов, А. С. Новиков, В. П. Поляков, Б. Тарелкин, Д. Егоров
Челябинск, Челябинский государственный университет

Проект утилиты для представления знаний, полученных по открытому исходному тексту программ

Аннотация

Для упрощения знакомства программиста с кодом программы известны инструменты, визуализирующие связи и отношения в программе. Целью разрабатываемого проекта является создание такого универсального инструмента.

Программное обеспечения с открытым исходным текстом может использоваться в методических целях как источник знаний о построении алгоритмов и записи их на современных языках программирования, посколько в исходном тексте содержится вся информация о закодированном алгоритме. [1]. Между тем применение специализированных программных инструментов обещает получение серьезных преимуществ обладателю открытого програм-много кода, так как позволяет сократить требуемые время и усилия для получения знаний об алгоритмах ПО.

На настоящий момент известны программные инструменты для изучения открытого кода, наиболее мощным из которых является

утилита DoxyGen [2], однако её функциональные возможности ограничиваются построением диаграм-мы классов изучаемого проекта. Применение структурных моделей известно для целей разработки программного обеспечения [3], предложено применить этот способ для анализа исходного кода.

Иерархию интерпретаций исходного текста можно в первом приближении представить следующим образом:

- структура файлов исходного текста,
- уровень синтаксиса — идентификаторы и служебные слова,
- структура данных исходного текста — иерархия классов,
- структура данных исходного текста — дерево объектов,
- операционная структура исходного текста — граф передачи управления,
- операционная структура исходного текста — граф потока данных.

Данная иерархия позволяет анализировать проект в разных «плоскостях» или «срезах», однако для достижения такой цели следует выработать способ представления структуры программного кода, позволяющий генерировать любое из перечисленных представлений по одному и тому же представлению исходного текста. В качестве внутреннего формата представления исходного текста применяем дерево объектов, описанное по xml -шаблону.

Для программной обработки XML в языке Java существует несколько моделей:

1. Текстовая модель, в которой обработка документа производится как текстового файла с использованием регулярных выражений и разделения на токены (tokens).
2. DOM (Document Object Model) — Объектная модель. Представление документа в виде дерева, узлы которого соответствуют компонентам XML-документа: элементам, атрибутам, текстовому содержимому, инструкциям обработки, объявлениям пространств имен и пр.

В проекте используется модель DOM, как наиболее близкая к модели внутреннего формата данных. DOM API, описанная W3C DOM Working Group, представляет собой набор интерфейсов для построения объектного представления в форме дерева анализируемого XML документа.

DOM API предоставляет функции для работы с деревом на простейшем уровне, чего недостаточно для работы XML-процессора. Поэтому был введен внутренний формат данных — TreeNode. После того как DOM-представление получено, его нужно преобразовать во внутренний формат. Преобразование производится путем обхода дерева DOM-элементов в глубину и построения соответствующего ему дерева из элементов TreeNode.

Полученное в результате обработки дерево структуры проекта предназначено для визуализации в интерфейсных модулях, для этого необходимо сериализовать структуру в форматах XML-документа и Dot-файла.

dot — формат данных одноименной утилиты из состава GraphViz, оптимально подходящий для текстового описания графов, отображающих различные аспекты структуры проекта. Кроме того, использование этого формата позволяет применять уже существующую утилиту, распространяемую в рамках GNU GPL, для построения изображений и их разметки, избавляя от необходимости реализовывать указанный функционал в модуле визуализации.

Созданный функционал построения структуры исходного текста позволяет представлять знания об алгоритме проекта с помощью представленного выше набора «срезов» по единому внутреннему набору данных древовидной структуры.

Литература

[1] Спинеллис, Диомидис Анализ программного кода на примере проектов Open Source. : Пер. с англ. — М. : Издательский дом «Вильямс», 2004. — 528 с. : ил.

[2] Инструментальные средства для изучения программного обеспечения с открытым исходным кодом / А.Н. Пустыгин, Д.С. Ботов, М.Ю. Ежов и др; под ред А.Н. Пустыгина — Челябинск: Изд-во ЮУрГУ, 2008 — 117 с.

[3] А.М. Вендров, Jet Info Online, 4/2004 Методические основы технологий разработки программного обеспечения. Визуальное моделирование Цитируется по
`http://www.redbrick.ru/develop/softwaredevelop_metodics/`

А. Н. Пустыгин, О. И. Суворов, И. С. Ермолаев, Д. С. Ботов,
А. С. Новиков, В. П. Поляков, Б. Тарелкин, Д. Егоров
Челябинск, Челябинский государственный университет

Опыт разработки инструментов исследования программного обеспечения с открытым исходным кодом

Аннотация

Отладка Java-приложений с помощью отладчиков, входящих в состав пакетов разработки или интегрированных сред, требует использования исходных текстов проекта. Если продукт выпускается под лицензией Open Source, то исходные тексты доступны. Если исходные тексты программного продукта не доступны, то требуются дополнительные сервисы и инструменты.

В данном докладе описана разработка прототипа отладчика для приложений на Java без исходных текстов, которое содержит минимальный, но в тоже время достаточный функционал, позволяющий управлять ходом выполнения Java- программы на уровне инструкций виртуальной машины Java.

Полученное приложение (Byte Code Wizard) будет взаимодействовать с виртуальной машиной Java посредством реализованных в прототипе отладчика JDM (Java Debug Manager) интерфейсов, библиотекой ASM для выполнения операций над байт-кодом. Взаимодействие JDM с виртуальной машиной Java осуществляется через агенты, работа с которыми ведется через сокет по протоколу JDWP.

Виртуальная машина, с загруженным приложением и подключенными агентами — отлаживаемая часть

Ее можно разделить на 5 важных составляющих:

- Виртуальная машина (JVM);
- Встроенный в виртуальную машину интерфейс управления (JVM TI);
- JDWP-agent (агент отладки) — программа, использующая JVM TI для управления работой виртуальной машины;

- Java-agent — агент, используемый для получения байт-кода классов. Этот агент может и не использоваться в отладке программы, но без него нет возможности работы с байт-кодом, так что для работы Byte Code Wizard он обязателен;
- Приложение, запускаемое на виртуальной машине.

Оба агента реализуют в себе протокол JDWP для взаимодействия с внешним отладчиком (JDM).

Управляющее процессом отладки приложение Java Debug Manager — отлаживающая часть

Отлаживающая часть состоит из следующих частей:

- Ядро JDM, в котором реализован протокол отладки JDWP, система взаимодействия отладчика с агентами отладки, все внутренние объекты JDM;
- Основной поток выполнения пользовательского интерфейса, который управляет ядром JDM, указывая когда отправлять запросы и, собственно, какие это будут запросы (установка точки останова, получение байт-кода, и другие);
- Два дополнительных потока приема и обработки ответов от агентов отладки. Создаются основным потоком. Они также используют ядро JDM для получения формируемых в объекты ответов, а также для запуска обработчиков этих ответов;
- Модуль для работы с байт-кодом (Byte Code Wizard). Он получает управление, когда пользователю необходимо произвести какие-либо действия с байт-кодом. Вызывается из основного потока выполнения вводом соответствующей команды («processbc»);
- Библиотека ASM используется модулем для работы с байт-кодом. При помощи неё вносятся изменения в байт-код классов отлаживаемого приложения.

В случае недоступности Java-агента второй дополнительный поток не создается, но тогда не доступны функции модификации байт-кода.

Процесс обмена сообщениями между этими двумя частями производится с помощью Socket -соединения на основе разработанного компанией Sun протокола отладки — Java Debug Wire Protocol (JDWP).

Заключение

В результате проделанной работы разработан и реализован прототип отладчика, выполняющий отладку программ, представляющих собой байт-код Java. Отлаживаемую программу можно запускать на любой программной или аппаратнойплатформе, на которой установлена виртуальная машина Java и присутствует агент отладки (библиотека) JDWP.

Данная система позволяет широко использовать доступные функции интерфейса JVM TI, а пользователь этого продукта получает достаточно мощный инструмент для изучения поведения Java программ и механизмов виртуальной машины, однако существует существенно ограничение.

Литература

[1] Java ByteCode Debugger (JBCD), 2003, `http://jbcd.sourceforge.net/`.

[2] Bytecode Visualizer, Dr. Garbage Ltd., 2008, `http://www.drgarbage.com/bytecode-visualizer-3-5.html`

[3] ASM, ObjectWeb, 2009, `http://download.forge.objectweb.org/asm/asm-guide.pdf`

[4] Eclipse IDE for Java Developers — Help, Eclipse corporation, 2009, `http://help.eclipse.org/galileo/index.jsp?topic=/org.eclipse.jdt.doc.user/reference/views/debug/ref-debug_view.htm`

[5] IntelliJ IDEA, JetBrains, 2009, `http://www.jetbrains.com/idea/features/compiler.html#link8`

[6] Java Development Kit (JDK) documentation, Sun corporation, 2009, `http://java.sun.com/javase/6/docs/`

[7] BCEL, Apache Software Foundation, 2006, `http://jakarta.apache.org/bcel/`.

[8] Abe White, SERP, 2007, `http://serp.sourceforge.net/`

[9] Java Development Kit (JDK) documentation, Sun corporation, 2009, `http://java.sun.com/javase/6/docs/`

[10] Java Debug Wire Protocol, Sun Microsystems, Inc., 2004, http://java.sun.com/j2se/1.5.0/docs/guide/jpda/jdwp-spec.html.

М. М. Дронов, Д. М. Ахметов
Ульяновск, Ульяновский государственный университет
Проект: Ульяновская группа пользователей Linux `http://ul-lug.ru/`

Интеграция взаимодействия проекта внедрения СПО в образовании и деятельности групп пользователей Linux

Аннотация

Доклад посвящен решению актуальной проблемы при внедрении СПО в учебном процессе как в средней, так и высшей школе, связанной с обучением школьников и студентов. В качестве примера выступает Ульяновская группа пользователей Linux

Группы пользователей Linux

Группа пользователей Linux, LUG (от англ. Linux User Group) — некоммерческое, неформальное объединение пользователей операционной системы GNU/Linux. Основными целями группы пользователей чаще всего являются взаимная поддержка и обмен программным обеспечением. Обычно группы создаются по территориальному признаку, например, Московская группа пользователей Linux, и др. Собственно говоря, группа пользователей Linux — некоммерческое сообщество людей, тем или иным способом использующих GNU/Linux в своей деятельности. Организаторами проводятся различные мероприятия и встречи пользователей.

Чаще всего группы пользователей создаются на базе университетов, многие мероприятия проводятся на базе университетов. Это неудивительно, так как *nix (в частности, GNU/Linux) является «университетской» системой. Ядро системы было создано Линусом Торвальдсом будучи студентом.

- *Сайт.* Основное общение пользователей происходит именно на сайте — здесь можно получить on-line консультацию у специалистов, задать любой интересующий вопрос, найти ресурсы для поиска информации и многое другое. Другими словами, сайт — основной инструмент в деятельности любой группы пользователей. С появлением Web 2.0 очень стали популярны сайты-блогосферы (к примеру, сайты на движке LiveStreet), где каж-

дый пользователь может вести свой собственный блог, где может делиться опытом с окружающим.

- *Linux Install Fest.* Стоит упомянуть об уникальности этого мероприятия. Linux Install Fest (фестиваль установки Linux) — событие, на котором люди собираются вместе для установки операционной системы либо программного обеспечения на своем компьютере. Обычно в качестве устанавливаемой операционной системы выступает Linux и набор свободного программного обеспечения. В целом, это информационное событие направленное на формирование сообщества. Новички приходят со своими компьютерами и копиями операционных систем, которые бы они хотели установить, а более опытные пользователи помогают им решая возникшие сложности. Иногда на мероприятии всем желающим раздаются флаеры и диски с дистрибутивами Linux. Некоторые организаторы просят принести с собой сетевые удлинители и свитчи. На этом мероприятии можно не только посмотреть «что такое Linux», но и ощутить это, поработав на компьютере с Linux. На фестиваль приглашаются пользователи всех уровней, от самых новичков до экспертов. Фестивали разнятся от неформальной встречи для установки до больших праздников с едой и напитками. Тон мероприятия, место проведения и набор событий во время фестиваля зависят от организаторов и спонсоров.
- *Линуксовки.* Рядовые встречи пользователей, обычно для того, чтобы пообщаться и отдохнуть в приятном кругу. Проводятся довольно часто, где каждый может осознать, что он — частичка большого сообщества пользователей Linux.

Ульяновская группа пользователей Linux

Проект «Ульяновская группа пользователей Linux» функционирует с июня 2009 года. Именно тогда был собран основной состав команды организаторов. В июле был создан сайт, `http://ul-lug.ru/`

Несмотря на столь небольшой возраст проекта, он уже успел достичь успеха.

В конце июля проект был представлен на всероссийском молодежном инновационном форуме «Селигер 2009», по итогам которого был признан одним из десяти лучших по итогам всей образовательной

смены. Было подписано пятистороннее соглашение с правительством Ульяновской области по совместным действиям в развитии проекта на 2010 год.

19 декабря на базе Ульяновского государственного университета был проведён первый Linux Install Fest. На мероприятии присутствовало порядка 40 человек, учитывая примерные прогнозы в 20-25. Интерес к СПО был налицо — участники увлеченно смотрели презентации и участвовали в дискуссиях.

На 2010 год также запланировано проведение встреч, семинаров, фестивалей установки. Также в марте будет организована подготовка волонтеров для проведения цикла ознакомительных семинаров «Что такое Linux».

Перспективы

Из вышесказанного ясно, что группы пользователей Linux — проекты информационные. Они создают идеальное поле для обучения и обмена опытом как в работе Linux, так и в системном администрировании и программировании на платформе *nix в целом.

Что могут обеспечить группы пользователей Linux:

- *Информационное взаимодействие.* Как указывалось выше, основным инструментом станет портал группы. У каждой группы своя посещаемость портала, зависящая от активности организаторов. Посещаемость портала специалистами в данной области поможет более быстрому обучению пользователей, ведь не секрет, что основным инструментом в обучении Linux является Интернет (т. к. и сам Linux разрабатывается тысячами программистов со всего света, контактирующими друг с другом во всемирной сети Интернет).
- *Коммуникативное взаимодействие.*Различные фестивали, семинары и прочие мероприятия, связанные с СПО собирают большое количество аудитории. Основная польза от таких мероприятий — непосредственное живое общение как опытных пользователей, так и начинающих в режиме реального времени.

Активное взаимодействие ведущих Российских компаний, связанных с внедрением СПО в образовании (Alt Linux, АйТи), общественных организаций (ЦеСТ www.centercest.ru, Российская группа пользователей Linux www.lug.ru) несет большие плюсы внедрению, так это бу-

дет не просто портал проекта внедрения, а портал, который соберёт различных специалистов, чья деятельность так или иначе связана с СПО. Активное общение, обмен опытом — несомненно большая выгода. Поддержка местных правительств только улучшит этот процесс.

Ю. П. Немчанинова, Е. Г. Пьяных
Томск, Томский Государственный педагогический университет
http://www.tspu.edu.ru

Проблема развития ИКТ-компентности педагогических кадров в условиях перехода на свободное программное обеспечение и организационно-педагогические условия ее решения

Аннотация

В докладе рассмотрен опыт обучения работе в среде Linux, накопленный в ТГПУ в рамках работы по реализации областного и федерального проектов по внедрению СПО в образование, а так же опыт обучения сотрудников ВУЗа в рамках внутривузовской системы повышения квалификации.

К настоящему времени в Томском государственном педагогическом университете накоплен достаточно большой опыт по внедрению СПО. Работы по переводу учебного и административного процессов на свободное программное обеспечение ведутся в ТГПУ с 2006 года. На первоначальном этапе проблемы перехода на СПО можно разделить на две группы: проблемы технические (выбор или создание дистрибутива, удовлетворяющего потребностям образовательного и административного процессов) и проблемы методические (обучение технического персонала и педагогических кадров). Вторая проблема оказалась наиболее острой.

В рамках решения методических проблем в ТГПУ были разработаны и апробированы программы повышения квалификации для профессорско-преподавательского состава вуза и административных кадров вуза. Обучение этих двух категорий ведется в рамках внутривузовской системы повышения квалификации. Первоначально была разработана программа «Использование программного обеспечения на платформе ОС Linux в профессиональной деятельности», включающая в себя следующие модули: информационно-коммуникационные

технологии в деятельности преподавателя вуза, программное обеспечение в образовании, использование текстового редактора в профессиональной деятельности (Open Office Org.Writer), использование электронных таблиц в профессиональной деятельности (Open Office Org.Calc), компьютерные презентации в системе высшего профессионального образования (на примере Open Office Org.Impress), обработка графической информации на компьютере, интернет-технологии в образовании, современные технические средства обучения. Программа рассчитана на 72 часа очного обучения. В дальнейшем была разработана программа «Использование информационных технологий в образовании», ориентированная на слушателей, уже имеющих базовые навыки по работе с СПО. Целью данной программы является дальнейшее формирование ИКТ-компетентности преподавателей ВУЗа. Программа включает в себя следующие модули: Проблемы информатизации образования в ВУЗе, Роль учебно-методического комплекса (УМК) в учебном процессе ВУЗа, Подготовка графических объектов для учебных материалов УМК, Создание учебных материалов для УМК с помощью редактора мультимедийных презентаций, Web-сайт как платформа для учебно-методического комплекса. Программа рассчитана на 72 часа очного обучения. В качестве учебного обеспечения к процессу повышения квалификации было разработано пособие «Практическая работа на компьютере в среде Linux», включающее в себя наиболее важные темы, необходимые слушателям на этапе перехода на СПО. Пособие включает базовые сведения по работе с операционной системой Linux, основы работы с пакетом OpenOffice.org, описание работы в сети Интернет и принципов работы некоторых наиболее употребляемых пользователями программ. За период с 2007 по 2009 год в ТГПУ было обучено в общей сложности более 140 сотрудников из числа профессорско-преподавательского состава и учебно-вспомогательного персонала.

Опыт, накопленный в ходе реализации внутривузовского повышения квалификации был успешно применен для обучения работников средних общеобразовательных учреждений на этапе перехода на свободное программное обеспечение. ТГПУ участвовал в проектах по внедрению СПО в образование — областном (`http://spo.tomsk.ru/site/`) и федеральном (`http://linux.armd.ru/`) проектах по внедрению СПО в образование. В обоих проектах ТГПУ отвечал за методическую поддержку процесса внедрения. В рамках областного проекта по внедрению СПО разработаны программы повы-

шения квалификации работников образования, ориентированные на разные категории слушателей: учителей–предметников, учителей информатики, системных администраторов, административные кадры школы. Программы повышения квалификации различных категорий пользователей разрабатывались с учётом их профессиональных потребностей. Содержание программ ориентировано на развитие знаний, умений, навыков использования СПО в деятельности учителя и, в конечном итоге, направлено на формирование информационно-коммуникационной компетентности. При разработке программ были учтены требования Закона РФ «Об образовании», Государственных образовательные стандартов, ведомственных нормативов, регламентирующих дополнительное профессиональное образование и повышение квалификации работников образования. Учитывая специфику взрослой аудитории, форма изложения материала предполагает предоставление возможности слушателям в ходе обучения делать логические выводы, адаптировать содержание к собственной практике и апробировать полученные умения в условиях тренингов и при выполнении специальных упражнений. Обучение ведется на основе сравнения, так как многие слушатели уже имеют опыт работы с коммерческим программным обеспечением. В качестве итоговой работы слушатели предоставляют тезисы или презентации к выступлениям или докладам по проблематике включения информационных технологий в современный урок, иллюстрированные таблицами, схемами, диаграммами, выполненные средствами офисных технологий или дидактические и раздаточные материалы, разработки уроков (внеклассных занятий) с использованием средств ИКТ. Для реализации программ предлагается использовать все многообразие форм и методов учебной работы: лекции, семинары, практические, в том числе, индивидуальные занятия, ознакомление с опытом коллег, обсуждение и анализ ситуаций, работу в малых группах, консультации, элементы дистанционного обучения. Содержание программ построено по модульному принципу. Модули характеристика ОС Linux, решение типичных пользовательских задач в ОС Linux, работа с текстовыми документами в редакторе Open Office Org.Writer, работа с электронными таблицами Open Office Org.Calc, создание мультимедийных презентаций в редакторе Open Office Org.Impress, Работа с графическим редактором Open Office Org.Draw, работа с векторным графическим редактором Inkscape, работа с растровым графическим редактором Gimp, компьютерные сети, Интернет включены в программы для учителей

информатики и для учителей предметников (с разным количеством часов). Помимо этого, программа для учителей предметников включает модули применение ИКТ в преподавании школьных предметов и работа с образовательными программами. Программа для учителей информатики включает модули установка и настройка ОС Linux и программирование. Принимая во внимание большую занятость слушателей, необходимость обеспечивать учебный процесс в школах обучение проводится преимущественно в вечернее время и в каникулы.

В рамках областного проекта в общей сложности было обучено более 400 слушателей, из которых 40 непосредственно обучались очно в ТГПУ. Для обучения остальных слушателей использована сетевая модель обучения с тьюторской поддержкой.

Учебные программы для учителей направлены не только на изучение СПО (офисные технологии, программирование), но и на методику применения СПО в деятельности учителя. Программа повышения квалификации для административных кадров образовательных учреждений в большей степени ориентирована на изучение офисных технологий и подготовку различной документации с использованием СПО. Базовой целью всех программ повышения квалификации является общее знакомство с возможностями СПО. Следует отметить, что программы повышения квалификации ориентированы не только на разные профессиональные категории пользователей, но и позволяют учитывать различный уровень владения информационными технологиями.

В качестве учебного обеспечения разработано более 20 пособий по работе с СПО, среди них пособия по пакету Open Office, по основам работы с OS Linux, по программному обеспечению для вёрстки и подготовки публикаций, по объектно-ориентированному программированию, по работе с образовательными программами и др. 18 пособий по свободному программному обеспечению были разработаны в рамках федерального проекта по внедрению СПО в образование и размещены на сайте проекта (`http://linux.armd.ru/ru/documentation/metod/`). В рамках апробации пособий было обучено 48 сотрудников общеобразовательных учреждений, среди которых учителя информатики, учителя предметники, системные администраторы и административные кадры ОУ.

Важно не только повышать квалификацию уже работающих учителей, но и готовить будущих педагогов, готовых работать со свободным программным обеспечением. С 2007 года учебный процесс

в ТГПУ практически полностью переведен на базу свободного программного обеспечения. Очень помогает в подготовке молодых специалистов программа Intel «Teach to the Future». Программа Intel «Teach to the Future» направлена на расширение применения передовых технологий в учебном процессе. Она призвана помочь учителям средней школы глубже освоить новейшие информационные технологии, расширить их использование в повседневной работе с учащимися и при подготовке учебных материалов. Обучение предусматривает работу с интернет–ресурсами, создание web–страниц и освоение офисных приложений. По программе проходят обучения студенты старших курсов педагогического университета. Будущие преподаватели учатся использовать базовые информационные технологии в их повседневном труде.С переходом образовательного процесса на свободное программное обеспечение возникла необходимость искать пути реализации данной программы. В результате была разработана программа, сочетающая в себе методические принципы интеловского курса и использование СПО.

В перспективе планируется организация дистанционного обучения по направлению «Применение СПО в профессиональной деятельности». В ТГПУ разработан ряд курсов для самостоятельного изучения, рассчитанных на разные категории обучающихся.

Е. С. Васильева
Ижевск, УдГУ

Самостоятельное изучение языка РЕФАЛ в высшей школе: проблемы и их преодоление

Аннотация

Язык РЕФАЛ — это одна из самых удачных разработок отечественной науки в области информационных технологий. Однако для самостоятельного изучения языка РЕФАЛ есть серьёзные препятствия, и, пожалуй, основной проблемой является отсутствие в открытом доступе простой для понимания школьником или студентом литературы и учебных программ. В данной работе предлагается решение этой проблемы, составленное с точки зрения студента, который столкнулся со всеми трудностями самостоятельного изучения РЕФАЛа.

Удивляет то, как мало в настоящее время литературы по языку РЕФАЛ. Язык создан российским ученым ещё в 1966 году, а возможностей для его самостоятельного изучения до сих пор мало.

Ведь изучение РЕФАЛА параллельно или сразу после изучения того же Паскаля весьма повлияло бы на развитие мышления студентов и школьников. Как правило, изучая лишь примеры императивных языков, учащиеся принимают подобную парадигму как единственную. В дальнейшем, сталкиваясь в процессе обучения или работы с нестандартными задачами, они впадают в ступор. Изначальное изучение хотя бы одной дополнительной парадигмы позволит использовать для решения различных задач наиболее подходящие приёмы и методы программирования. Однако даже при самостоятельном изучении РЕФАЛа возникает ряд проблем.

Проблемы

Последовательное возникновение проблем:

- Если современному студенту надо что-либо узнать, он первый делом забивает это в поисковик. Google, Яндекс, Nigma, кому что нравится. По запросу «Учебник по рефалу» только одна подходящая ссылка `http://www.botik.ru/pub/local/scp/refal5/`, но учебник на английском языке и является авторским переводом руководства В.Ф. Турчина `http://refal.net/rf5_frm.htm`

  В итоге, единственная информация — описания языка, составленные самим создателем В.Ф. Турчиным и его последователями. Но они созданы скорее не для тех, кто хочет изучить язык, а для людей уже им пользующихся.

- В процессе штудирования имеющихся описаний языка РЕФАЛ, неизбежно возникают вопросы, особенно по принципу работы РЕФАЛ-машины и рекурсивности вызовов функций. А без понимания подобных ключевых моментов писать программы на РЕФАЛе сложновато.

- Даже если уяснить основные принципы выполнения РЕФАЛ-программ, то заставить мышление выдать алгоритм решения какой-либо задачи, подходящий для реализации на РЕФАЛе сложно. Особенно тем, кто начинал изучение языков програм-

мирования с Паскаля. А примеров в описаниях, перекликающихся с императивными языками, практически нет.

Решение

Для решения вышеуказанных проблем предлагается:

- Составление подробного описания языка программирования РЕФАЛ-5. По сути составление учебника с разделами «Для начинающих» и «Для продолжающих», который будет интересен как тем кто только узнал о существовании языка рекурсивных функций, так и тем кто уже более-менее активно его использует.
- Использование в описании примеров, перекликающихся с императивными языками, в частности с Паскалем. Это сложная задача, но для упрощения понимания языка её надо решить.
- Важным этапом в изучении языка является выполнение заданий. Однако, как правило задания составляются и помещаются в учебники. Однако ответов к ним нет. Какой смысл давать в учебнике для самообучения задания без возможности самопроверки? Поэтому в описании будут теоретические задания с ответами, а практические (программы) — хотя бы с тестами и результатами их выполнения.

Автором доклада была проделана работа в данном направлении. Ее результатом стало следующее:

- Составлена статья — описание языка РЕФАЛ-5.

  В разделе «Для начинающих». Составлено описание использования интерпретатора для РЕФАЛа-5. В том числе инструкция по установке, запуску первой программы (Hello World!) и способам её редактирования. Подробное описание базиса языка: структура программы на РЕФАЛЕ, осуществление вызовов функций, типы переменных. К каждому разделу прилагаются простые, но ёмкие примеры с пояснениями. Для успешного применения описанного материала использованы теоретические и практические задания. Составлено подробное описание принципов работы РЕФАЛ-машины. Приведены подробные пошаговые примеры работы РЕФАЛ-машины.

  Раздел «Для продолжающих» содержит описание расширений языка РЕФАЛ и основных приёмов программирования.

- Создана программа обучения.

  В программе обучения находится ряд задач по РЕФАЛу-5, составленный по возрастанию сложности. Существует возможность проверки правильности решения.

  Основные разделы программы обучения: использование инструмента (интерпретатора), основы языка, расширения языка, приёмы программирования.

В дальнейшем планируется развитие описания языка и программы обучения по мере их практического применения. Так же в будущем, на основе уже существующих статьи и программы , будет составлена программа обучения для других диалектов РЕФАЛа, в частности Динамического Рефала.

Литература

[1] Турчин В.Ф., РЕФАЛ-5. Руководство по программированию и справочник., http://www.refal.net/rf5_frm.htm

В. А. Бондаренко
Нижний Новгород, Нижегородский радиотехнический колледж
nntc.nnov.ru, blog.nntc.nnov.ru, cunewebform.nntc.nnov.ru

Информационная система учебного заведения на базе свободного программного обеспечения на примере Нижегородского радиотехнического колледжа

Аннотация

Информатизация учебных заведений начиналась с обеспечения работоспособности учебных классов для проведения практических занятий. Но за последние 5-7 лет задачи информационной системы учебного заведения значительно расширились. Компьютеры появились не только в учебных классах, но и у администрации колледжа, в преподавательских комнатах, библиотеках. Так же не стоит забывать и о персональных компьютерах студентов, которые они приносят с собой

из дома. Задачей современной информационной системы стало объединить все это в общую структуру и обеспечить сервисы для взаимодействия преподавателей, студентов и сотрудников колледжа. Использование коммерческого программного обеспечения для решения данной задачи ведет к значительным финансовым затратам и не всегда позволяет добиться необходимой интеграции. В Нижегородском Радиотехническом Колледже была построена система удовлетворяющая всем самым современным требованиям учебного процесса и требованиям работы подразделений на базе Свободного Программного Обеспечения. Предлагаем вам рассмотреть технологии и методы использовавшиеся для создания интегрированной информационной системы ФГОУ СПО «НРТК».

Задачи информационной системы

1. Организация занятий в учебных классах.

На этапе внедрения наша задача потребовала собрать данные по используемому программному обеспечению в учебном процессе. В дальнейшем, был произведен анализ и подбор аналогов программ среди Свободного Программного Обеспечения. В ходе использования информационной системы разрабатывались новые программы по предметам «Компьютерные сети и технологии», «Интернет технологии», «Разработка программного обеспечения», практические занятия и методические материалы к которым включили в себя свободные инструменты, программы, протоколы и технологии.

2. Централизованное хранение учетных записей и данных пользователей.

Задача первостепенной важности для учебных заведений с числом студентов больше 200 человек. В Нижегородском Радиотехническом Колледже 7 компьютерных классов использующихся в «поточном» режиме. Практические занятия по предмету в разные дни могу проходить в разных классах. Это требует возможности доступа студента к своей домашней директории с любого компьютера в колледже в любое время учебного процесса из любой используемой операционной системы. Для этого на центральном сервере колледжа была развернута система централизованной аутентификации с хранением домашних каталогов пользователей на сервере.

3. Централизованное управление доступом в интернет для администрации, студентов и преподавателей.

Задача обеспечения доступа к интернет для сотрудников и студентов колледжа делиться на 2 части: распределение пропускной способности интернет канала и контроль доступа к интернет-ресурсам (ограничение доступа к сайтам с нежелательным содержанием). При этом, система контроля доступа так же должна интегрироваться в общую структуру информационной системы.

4. Обеспечение работы внутреннего информационного портала (для размещения информации, методического материала и практических заданий).

Активно продвигаемые коммерческие решения зачастую перегружены ненужным функционалом и требуют значительных ресурсов от сервера. В нашем случае задача обеспечения интерактивным инструментом преподавателей для размещения методических материалов и заданий для практических занятий была решена с помощью системы MediaWiki.

5. Обеспечение функционирования внутренних сервисов.

- Электронная библиотека;
- Online система распознавания текстов;
- Электронный журнал куратора групп;
- Система тестирования студентов;
- Система заполнения бланков дипломов;
- Электронный журнал неисправностей компьютерной техники.

6. Сервис централизованного сбора и обработки электронной почты.

колледж использует единый адрес электронной поты. Но для работы требуется одновременный доступ к электронной корреспонденции нескольких сотрудников из различных подразделений. Данная задача прекрасно решается с помощью свободной почтовой системы Zimbra.

7. Интернет-представительство колледжа (сайт и блог-портал).

web-сайт уже не новость. Каждое учебное заведение считает обязательным иметь представительство в интернете с информацией и данными о специальностях и условиях приема. Для работы сайта nntc.nnov.ru мы выбрали свободную систему управления контентом Jumla. Для более динамичного представления жизни колледжа было принято решение об открытии блог-раздела, где появляются все самые последние новости о событиях, мероприятиях и достижениях колледжа. В качестве системы управления контентом сайт blog.nntc.nnov.ru использует свободную систему WordPress.

Н. Живчикова, Е. Иванов, А. Котомин, Е. Титова
Переславль-Залесский, НОУ ИПС – «УГП имени А.К. Айламазяна»

Проект: Информационная система для университета `http://edu.botik.ru`

Информационная поддержка трекинга учебных практик и научных конференций

Введение

Длительное существование научно–образовательного комплекса, созданного на базе Института программных систем Российской академии наук в Переславле-Залесском, невозможно без подготовки высококвалифицированных кадров.

Мы настойчиво ищем высокоэффективные технологии качественного образования[2].

Известные технологии информационного обеспечения совместной деятельности не дают достаточной основы для перспективных образовательных методик.

Нужна информационная система, способная так объединить усилия студентов разного уровня и аспирантов в реальной практической деятельности, чтобы освоение нового материала, взаимодействие с передачей опыта от старших младшим, самооценка, взаимооценка и сдача–прием заданий и продуктов происходили с опорой на множество подсистем разносторонней оценки вклада каждого в общий процесс и абсолютно прозрачно для присматривающих специалистов высочайшей квалификации.

Целью создания информационной системы «Ботик»[1] является информационная поддержка не только традиционных форм обучения, но и групповой проектной деятельности, студенческих научных конференций и других (в том числе экспериментальных, пока неизвестных) видов образовательной деятельности.

Подсистема трекинга учебных и производственных практик

Модуль трекинга разрабатывался для контроля прохождения студентами учебных, производственнных и преддипломных практик.

Подсистема позволяет

студенту:

– заносить отчеты и оценивать свою деятельность,
– получить дневник прохождения практики в формате pdf;

научному руководителю:

– следить за выполнением студентом плана практики,
– оценить работу студента по окончании практики;

учебной части:

– утвердить или отклонить научного руковдителя и место прохождения практики, заданные студентом,
– просмотреть текущее состояние практики в виде диаграмм.

Подсистема поддержки конференций

Проведение студенческих конференций стимулирует творческую исследовательскую активность, развитие технических и психологических навыков деловых коммуникаций в интеллектуальной сфере, обмен идеями, опытом и результатами практической работы. Будучи составной частью учебного процесса, научно-практические конференции УГП заметно отличаются от большинства научных и научно-практических конференций ярко выраженной образовательной направленностью.

Подсистема автоматизирует следующие этапы проведения конференции [3]:

- прием заявок на участие в виде тезисов, оформленных в системе LaTeX;
- добровольное участие преподавателей и студентов в рецензировании;
- обязательное участие авторов статей в рецензировании не менее 4 чужих работ;
- учет научных интересов рецензентов при рандомизированном распределении докладов на рецензирование;
- ролевые дискуссии по каждой статье с отбором замечаний для доработки и контролем их выполнения;

- автоматическое постороение линейного порядка работ, наиболее точно согласующегося с линейными порядками рецензентов;
- отбор лучших работ в сборник тезисов и программу конференции на основании линейного порядка;
- работа с авторами по устранению замечаний рецензентов перед изданием сборника;
- выступление авторов, не попавших в программу, в качестве оппонентов;
- подведение итогов конференции на объективной основе учета мнений всех слушателей, пожелавших принять участие в ранжировании докладов и оппонентов;
- количественная оценка каждого участника.

Заключение

Информационная система реализуется отрыто на свободном программном обеспечении. В настоящий момент студентами 1-3 курсов ведется разработка подсистемы учебной части.

Литература

[1] *Коряка Ф.А.* Автоматизированная система управления вузом — UPIS. XI научно-практическая конференция «Университета г. Переславля». Переславль-Залесский, апрель 2007, изд.-во «Университет города Переславля», Т. 1, с. 59—63. `http://wiki.botik.ru/up/pub/IS4UGP/StudConf/1-2/03-koryaka-p-59.pdf`

[2] *Амелькин С.А., Знаменский С.В.* Информационная поддержка организации сложной совместной деятельности. // Труды международной конференции «Программные системы: теория и приложения», ИПС РАН имени А. К. Айламазяна, г. Переславль-Залесский, май 2009. Переславль-Залесский: Изд-во «Университет города Переславля», Т.1, 2009. — с. 123—132

[3] *Живчикова, Н.С., Знаменский, С.В., Титова, Е.В.* Информационная поддержка научных конференций. // Тезисы докладов Международной научно-образовательной конференции «Наука в вузах: математика, физика, информатика. Проблема высшего и среднего профессионального образования». — М.: РУДН, 2009. — с. 883–885

Е. Л. Сыромятников
Москва, ALT Linux
Проект: семинар UNИХ http://uneex.ru/

Курс «Сопровождение пакетов в Linux» — заметки на полях

Об одном курсе из многих

Проблемный семинар UNИХ [1], основанный и возглавляемый Георгием Курячим, являет собой уникальную в рамках факультета ВМК среду, где слушатели курса могут выразить своё желание прослушать (и, иногда, рассказать) то, что интересно им, а не только лектору.

Идея курса по сопровождению дистрибутива и сборке пакетов зрела давно, и в этот раз она, наконец, приняла вполне осязаемую форму при практически полностью единодушном согласии заинтересованных в семинаре людей.

Сама задача по прочтению подобного курса, несмотря на его необходимость и актуальность, достаточно нетривиальна, так как курс имеет явно практическую направленность. Это вылилось в такой экспериментальный для данного семинара способ передачи знаний, как realtime-демонстрация сборки пакета, проведённая практически без подготовки.

По окончании курса и прошествии ряда итераций экзамена, показавших усвоенные слушателями знания, можно сделать следующие выводы:

– Лекционные курсы по темам с практической направленностью вполне успешно могут использовать демонстрации при отсутствии возможности проводить семинарские занятия, более того, возможно, их следует признать обязательными (создалось впечатление, что по ряду моментов именно демонстрация позволила развеять часть иллюзий относительно такой недостаточно документированной области знаний, как сопровождение пакетов)

– Относительно высокая средняя подготовленность (по сравнению с некоторыми предыдущими семестрами) людей, сдававших экзамен, при практически аналогичном их количестве, наводит на мысли, что проведение исключительно лекционного курса было сделано не напрасно и что данная тема действительно актуальна.

О наследии

Задача сохранения и передачи «тайного знания» (здесь: знания, не являющегося общеизвестным и легкодоступным) всегда являлась одной из основных. Данная задача становится тем более сложной и тем более актуальной, чем более этого знания появляется и чем меньшую его долю способен отдельный индивид усвоить. Одним из способов сохранения знаний является их запись в том или ином виде.

Основная проблема заключается в том, что для сохранения сиих знаний в индексируемой для последующего поиска и удобной для быстрого усвоения форме необходимо затратить усилий не меньше (а гораздо больше), чем его передать, пусть даже и неоднократно.

Данный случай вполне конкретен — сохранить информацию, изложенную на лекциях, учитывая тот факт, что читаются они единожды.

Как показывает практика автора, после конспектирования лекции в реальном времени результат оного конспектирования как таковой оставляет желать лучшего и требует значительной переработки. По приблизительным оценкам, на один час лекции уходит от 6 до 12 человеко-часов для получения полноценных конспектов, снабжённых ссылками, примечаниями и связями с другими частями лекционного курса, в результате чего от исходного конспекта мало что остаётся (пример подобной работы над конспектами, которая проводилась в рамках проекта «Документирование ПСПО» [2], [3] — `http://uneex.ru/PspoClasses/080720/03PackageUtils?action=diff&rev2=19&rev1=1`). Как следствие, данный вид материалов, несмотря на большую его полезность (при наличии достаточно информативного и актуального исходного материала), очень трудоёмок в своём изготовлении, и для го получения требуется значительная мотивация (например, финансовая, как это было в упомянутом проекте документирования), которая возможна далеко не всегда.

Типичным решением после осознания данного факта и наличии желания тайное знание таки сохранять является запечатление как

можно большего объёма информации (не только текстовую, но и аудиальную и визуальную составляющую) для возможности дальнейшего восприятия хотя бы в таком виде, что являет собой некий компромисс — с одной стороны, объём требуемых работ снижается, с другой, время на восприятие и поиск требуется больше, что приводит к несколько меньшей ценность подобного формата материалов по сравнению текстовым конспектом.

В рамках курса по сопровождению пакетов была сделана попытка делать помимо аудиозаписи (которая производилась уже второй год и не пользовалась большим спросом, особенно по сравнению с конспектами лекций) запись видео.

Несмотря на достаточно длительную историю, предшествовавшую началу видеозаписи лекций (в том числе, видеозаписи смежных событий — собраний этого же проблемного семинара), не всё шло гладко, ряд первоначальных замыслов (в частности, касающихся потоковой трансляции) не был осуществлён.

Хотя две из четырёх записанных видео на данный момент ещё находятся в состоянии пост-обработки, уже сейчас можно сделать ряд выводов (довольно банальных, но тем не менее):

– Не любая видеозапись лекции пригодна для её просмотра. Полноценный монтаж, делающий видеозапись пригодной к использованию в качестве учебного материала, занимает время, сравнимое с подготовкой полноценных конспектов на основании записанного во время лекции (тем не менее, время это примерно в 2—3 раза меньше)

– В виду отсутствия индексируемого материала (текста) видеозаписи лекции для их успешного многократного просмотра (а, как следствие, востребованности и придания осмысленности телодвижениям, осуществляемым для их создания) необходимы факторы, делающие просмотр ценным для смотрящих. К сиим факторам, по мимо объективных (например, качества исходного материала, монтажа, информативность материала) могут относиться такие слабо зависящие от собственно видеозаписи вещи, как артистизм лектора, например.

– Лектор должен учитывать то, что его снимают, и накладываемые этим фактом технические ограничения (например, угол обзора и размер кадра, не позволяющий захватывать всю доску целиком и/или разобрать недостаточно крупные надписи).

– Разворачивание видеозаписывающей аппаратуры занимает время большее (примерно 10—15 минут), чем средняя задержка начала чтения лекции относительно начала пары.

– Как показала практика смежных мероприятий (собраний) семинара, не смотря на то, что во время «живых» лекций использование слайдов не всегда актуально и/или оправдано, они весьма способствуют восприятию информации при просмотре видеозаписи. Вероятно, для последующих курсов они будут готовиться специально.

Весной 2010 года планируется продолжить практику видеозаписи лекций с учётом полученного опыта, дополнив её поточным вещанием лекций, что, возможно, привлечёт дополнительную аудиторию к планируемому курсу «Программирование в Linux» LinuxProgramming.

Данный текст (вместе со всеми ссылками) доступен на сайте семинара UNИХ по адресу http://uneex.ru/eSyr/Pereslavl2010/Thesis

Литература

[1] Проблемный семинар UNИХ, `http://uneex.ru/`.

[2] *Курячий Г. В.* Проблемы и методы командной разработки свободных, учебных материалов, *«Четвёртая конференция «Свободное программное обеспечение в высшей школе» (Переславль). Тезисы докладов.»*, 2009, стр. 60—63, `http://www.altlinux.ru/media/book-thesis-Pereslavl-2009-4.pdf`.

[3] *Курячий Г. В., Сыромятников Е. Л.* Командная разработка свободных учебных материалов по проекту «Документирование пакета свободного программного обеспечения», *«Свободное программное беспечение в образовании. Сборник трудов Всероссийской конференции (г. Челябинск)»*, под редакцией А. В. Панюкова, Издательство ЮУрГУ, 2009, стр. 14—18, `http://freeschool.altlinux.ru/wp-content/uploads/2009/03/chelyabinsk_25-26_march_2009.pdf`.

[4] Страница курса «Программирование в Linux» на сайте семинара UNИХ, `http://uneex.ru/LecturesCMC/Programming2010`.

П. Е. Еньков
Ульяновск, Ульяновский Государственный Университет
petere@mail.ru

Вычислительный кластер ЭВМ на основе операционной системы ALTLinux с использованием свободного программного обеспечения (СПО)

Аннотация

В докладе рассматривается построение кластера в ОС ALTLinux.Тема увеличения скорости вычислений весьма актуальна для всех тех, чья деятельность связана с большим объемом вычислительных работ. Так, например, рассчеты гравитирующих газовых дисков. Даже сейчас у российских университетов чаще всего нет средств для закупки мощных компьютеров типа nCube, Cray или подобных. Однако с развитием программного обеспечения и появлением свободно распространяемой операционной системы Linux стало возможным создать вычислительный комплекс с эффективным быстродействием, сравнимым с быстродействием суперкомпьютеров, но со стоимостью в десятки раз меньшей.

В наше время круг задач, требующих для своего решения применения мощных вычислительных ресурсов, постоянно расширяется. Это связано с тем, что произошли фундаментальные изменения в самой организации научных исследований. Вследствие широкого внедрения вычислительной техники, значительно усилилось направление численного моделирования и численного эксперимента. Численное моделирование, заполняя промежуток между физическими экспериментами и аналитическими подходами, позволило изучать явления, которые являются либо слишком сложными для исследования аналитическими методами, либо слишком дорогостоящими или опасными для экспериментального изучения. При этом численный эксперимент позволил значительно удешевить процесс научного и технологического поиска. Стало возможным моделировать в реальном времени процессы интенсивных физико-химических и ядерных реакций, глобальные атмосферные процессы, процессы экономического и промышленного развития регионов и т.д. Очевидно, что решение таких масштабных задач требует значительных вычислительных ресурсов.

Мощности современных процессоров вполне достаточно для решения элементарных шагов большинства задач, а объединение нескольких десятков таких процессоров позволяет быстро и эффективно решать многие поставленные задачи, не прибегая к помощи мэйнфреймов и супер компьютеров.

Сейчас в наших научных организациях и университетах, как правило, имеются энтузиасты бесплатного распространяемого ПО и специалисты по ОС Linux. В то же время парк более-менее современных персональных компьютеров в этих организациях так же имеется. Закономерно появилась идея создания параллельных вычислительных систем из общедоступных компьютеров на базе процессоров Intel и недорогих Ethernet-сетей, устанавив на эти компьютеры Linux и, объединив с помощью одной из бесплатно распространяемых коммуникационных библиотек (PVM или MPI) эти компьютеры в кластер. Оказалось, что на многих классах задач и при достаточном числе узлов такие системы дают производительность, сравнимую с той, что можно получить, используя дорогие суперкомпьютеры.

В самом деле, кластеры Beowulf обеспечивают университеты с ограниченными ресурсами хорошей платформой для изучения параллельного программирования и недорогой производительной вычислительной системой для ученых. Затраты на установку в университетах минмиальны: многие студенты заинтересованы в таких проектах и используют Linux на собственных компьютерах, установка кластера и написание параллельных программ является частью процесса обучения.

Beowulf — это мультикомпьютерная архитектура, которая может использоваться для параллельных вычислений. Это система, обычно состоящая из одного серверного узла и одного или более клиентских узлов, соединенных при помощи Ethernet или некоторой другой сети. Это система, построенная из готовых промышленных компонент, например ПЭВМ, на которых может работать ОС Linux, стандартных адаптеров Ethernet и коммутаторов. Она не содержит специфических аппаратных компонентов и легко воспроизводима. Beowulf также использует программные продукты, такие как ОС Linux, среды программирования Parallel Virtual Machine (PVM) и Message Passing Interface (MPI). Серверный узел управляет всем кластером и является файл-сервером для клиентских узлов. Он также является консолью кластера и шлюзом во внешнюю сеть. Большие системы Beowulf могут иметь более одного серверного узла, а также возможно специ-

ализированные узлы, например, консоли или станции мониторинга. В большинстве случаев клиентские узлы в Beowulf пассивны. Они конфигурируются и управляются серверными узлами и выполняют только то, что предписано серверным узлом. В бездисковой конфигурации клиентов, клиентские узлы даже не имеют IP-адресов или имен, пока их не назначит сервер. Одно из основных отличий Beowulf от кластера рабочих станций состоит также в том, что Beowulf работает как одна машина. В большинстве случаев клиентские узлы не имеют клавиатур и мониторов, и могут быть доступны только через удаленное подключение. Узлы Beowulf могут рассматриваться как элементы процессор+память, которые вставляются в кластер так же как процессор или модуль памяти вставляются в материнскую плату.