НОУ «ИПС-Университет г. Переславля им. А. К. Айламазяна»
Институт Программных Систем РАН
Факультет вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова
Удмуртский государственный университет
Институт Логики
ALT Linux

Третья конференция

«Свободное программное обеспечение в высшей школе»

Переславль, 2–3 февраля 2008 года

Тезисы докладов

Москва,
ALT Linux,
2008

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

Программа конференции

2 февраля

11.00–11.30: Регистрация в холле гостиницы

11.30–12.00: Кофе-брейк

Утреннее заседание
12.00–13.30

12.00–12.10: В. Н. Юмагужина, С. М. Абрамов. Вступительное слово

12.10–12.20: А. Е. Новодворский. Информация оргкомитета

12.20–12.30: Н. А. Юренко. Взгляд на СПО со стороны Минобрнауки

12.30–13.00: Н. Н. Непейвода

13.00–13.30: А. К. Петренко, О. Л. Петренко, В. В. Рубанов

13.30–14.30: Обед

Дневное заседание
14.30–16.30

Вечернее заседание
17.00–19.30

3 февраля

9.30–10.00: Завтрак

Утреннее заседание
10.00–13.30

10.00–10.20: М. А. Ройтберг, В. В. Яковлев

Конвертор КУМИР – C++: поддержка перехода от учебных к профессиональным системам программирования 42

10.20–10.40: А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, Д. В. Хачко, В. В. Яковлев, А. В. Карпов, Н. М. Субоч

Система программирования КУМИР — интегрированная поддержка начальных курсов информатики и программирования 45

10.40–11.00: Д. А. Костюк, Д. А. Ильяшевич

Опыт внедрения свободного ПО в учебный процесс для специальностей информатики и радиоэлектроники 48

11.00–11.20: А. Н. Непейвода

Автоматизированное построение сетевого графика проекта. 51

11.20–11.40: Е. Д. Патаракин

Рост знаний в в сообществе Летописи.ру 53

11.40–12.10: Кофе-брейк

12.30–12.30: Б. Б. Ярмахов

OLPC как модель массового внедрения свободного ПО в образование 58

12.30–13.00: А. А. Панюкова

Создание обучающего курса для детей на базе Linux 61

13.00–13.30: М. В. Быков

Tagged Thesaurus древнегреческого языка 64

13.30–14.30: Обед

Дневное заседание
14.30–16.30

Вечернее заседание
17.00–19.30

Вне программы

Н. Н. Непейвода

Ижевск,
Удмуртский Государственный Университет

Логика как средство моделирования эффектов больших и средних систем в процессе обучения

Одним из камней преткновения при построении системы образования в тех областях, где требуется работа специалиста над задачами достаточно высокого уровня, является то, что хорошая работа над такими задачами невозможна без овладения хотя бы первоначальными навыками работы с эффектами, присущими большим и средним системам. А в реальности в преподавании рассматриваются лишь «учебные примеры» малых систем, на которых попытка заставить работать так, как нужно было бы хотя бы на средних, приводит лишь к «лишней писанине» и ничем (на взгляд студента) не оправданным неудобствам. Более того, практика решения пары таких задач вызывает у большинства студентов стойкую аллергию на изучаемые методы, поскольку они видят, что эти же задачи без наложенных на них «оков» они решили бы в несколько раз быстрее и проще.

Эта реальность связана с объективными факторами: ограниченность времени и интеллектуальных ресурсов студента как в целом, так и в рамках одной отдельно взятой дисциплины, и малоге количество занятий, отводимых на конкретную задачу. Конечно же, возможна попытка решения возникшей проблемной ситуации при помощи «грубой силы»: дать группе студентов проект на целый семестр. Но тут возникают другие сложности. Во-первых, проект получается всего один, и во-вторых, сложности задачи могут далеко отступить перед сложностью организации коллективной работы, тем более при практическом отсутствии (в реальности) у преподавателя инструментов влиять на мотивацию и работу студентов.

Возникает вопрос, нельзя ли здесь попытаться воспользоваться давно известным в теории творческого мышления и в стратегии войны принципом: не лезть в лоб на хорошо укрепленные позиции, а предпринять косвенные воздействия? Оказывается, в этом случае возможно промоделировать многие эффекты больших и средних систем, возникающие в практике информатиков, на системах совсем другого класса: логических задачах на стыке с лингвистикой.

Рассмотрим простейший пример. Как показать невозможность автоматизированной формулировки формальной постановки задачи по содержательной? Просто на задачах, связанных с переводом с естественного языка на формальный. Уже совокупность задач типа «Все A Ξ друг друга» даёт возможность увидеть, что совершенно подобные на первый взгляд высказывания переводятся по-разному в зависимости от входящих в них содержательных терминов. Пишется здесь «все», а часто переводится $\exists$, & часто переводится как $\vee$.

Доисторические ящеры пожирали друг друга
$\forall x(\text{Ящер}(x) \Longrightarrow \exists y(\text{Ящер}(y)\&(\text{Пожирает}(x,y) \vee \text{Пожирает}(y,x)))$
Члены правления ООО «Братва» ненавидели друг друга
$\forall x \forall y(\text{ЧП}(x)\&\text{ЧП}(y)\&x \neq y \Longrightarrow \text{Ненавидит}(x,y)\&\text{Ненавидит}(y,x)))$
и т. д.

Другие подобные задачи дают возможность выработать стойкое отвращение к пониманию текста по ключевым словам.

Обратные задачи — перевода формального текста на естественный язык — предоставляют возможность осознать, насколько нетривиальна задача объяснения результатов работы системы «обычным людям»: для понимания формулы необходимо восстановить модальности и другие особенности, опущенные в процессе формализации.

Задачи на анализ и восполнение рассуждений дают возможность увидеть, насколько часто в рассуждениях бывают скрытые посылки, насколько эти посылки бывают труднонаходимы как из-за нетривиальности, так и из-за полнейшей тривиальности, и насколько они необходимы для правильной формализации задачи. Например, в знаменитом «силлогизме Галена» типа

$$\frac{\text{Все студенты списывают} \qquad \text{Все студенты способные}}{\text{Некоторые способные люди списывают}}$$

опущены сверхтривиальные посылки, а в рассуждении

$$\frac{\text{Никто не любит нас, наркоманов} \qquad \text{Некоторые депутаты Думы нас любят}}{\text{Следовательно, } \ldots}$$

нетривиальное уточнение одной из посылок, но настолько легко находимое, что не составляет труда найти его одновременно с восстановлением следствия.

Рассмотрим т. н. «парадокс кучи куч». Если пытаться проверить на языке логики естественное рассуждение типа

«Для исследования космоса нужна куча денег.
Для армии нужна куча денег.
Для социальных программ нужна куча денег.
Для охраны среды нужна куча денег.
Денег на все не хватит.
Значит,. . . »

выявляется, что одинаковые на вид утверждения «нужна куча денег» должны переводиться разными высказываниями. Более того, здесь можно отметить такой эффект. Если эту же задачу формулировать на обычном математическом языке, то суть трудности маскируется использованием чисел и операции +. Если запретить пользоваться числами, то это будет воспринято студентами как искусственное условие, и задача будет интуитивно отторгаться. А в логике нет ни чисел, ни соблазна их использовать, и всё выглядит естественно.

Гибельность улучшений хороших систем также может быть показана еще на 1 курсе в курсе логики на базе опять-таки переводов достаточно сложных предложений. Тогда же может быть показана (например, на примере нестандартного анализа) роль абстрактных и принципиально нереализуемых концепций (*призраков*) при разработке программ, несравнимость выигрыша, достигаемого использованием идеальных высокоуровневых понятий с любой ползучей оптимизацией (теорема Оревкова), и одновременно необходимость высокой культуры при использовании таких понятий.

Теорема Лося показывает, что анализ моделей часто подсказывает язык, на котором целесообразно формулировать задачу. И даже модульность, которая, конечно же, показывает свои преимущества на достаточно средненьких программах, доступных в традиционном курсе программирования, в логике буквально навязывает себя, уже начиная с формул в одну строчку.

Возникает вопрос, почему же удалось достичь такого колоссального снижения сложности задач, на которых демонстрируются нетривиальные системные эффекты? За счёт четырёх факторов.

Первый фактор — естественный язык, входящий во многие такие задачи. Он на самом деле исключительно сложная система, но достаточно хорошо освоенная студентами. Простенькая формулировка задачи является лишь вершиной громадного айсберга привлекаемого контекста.

Второй взаимосвязанный с этим фактор — отказ от неявной аксиомы преподавания, что задача должна быть замкнута внутри читаемого курса или, в крайнем случае, внутри их системы. Открытые задачи для открытого образования!

Третий фактор — явное нарушение явной аксиомы, что задача должна быть поставлена точно. В жизни все задачи будут поставлены неточно, либо неточно и неправильно, либо точно, но неправильно. Именно из-за этой аксиомы образование слишком часто противоречит жизни.

Четвёртый фактор — то, что через логику можно быстро достичь высот абстракций, к которым приводит математика.

И, наконец, последний вопрос. Является ли логика единственным путём достичь такого эффекта? На проверенной практике это пока что единственный путь, но только что проделанный анализ показывает, что *некоторые другие отрасли (по крайней мере) математики вполне могли бы быть перестроены так, чтобы достичь подобных же эффектов*. Так что логика является эталоном, а не каноном. Вопросы, какие именно науки могут быть центральными здесь, как именно преподавать их для этого и как нужно в связи с этим перестроить весь цикл дисциплин, остаются тем, кто осмелится предпринять серьёзные и комплексные педагогические эксперименты. Единственное, в чем уверен автор, — что любой, начинающий такой эксперимент, должен знать логику по крайней мере настолько же хорошо, насколько мы, логики, в нем участвующие, знаем математический анализ и лингвистику, и прекрасно понимать реальные задачи информатики.

Литература

[1] *Н. Н. Непейвода* Прикладная логика, Новисибирск: НГУПресс, 2000.

[2] *Н. Н. Непейвода, И. Н. Скопин* Основания программирования, Москва-Ижевск: Институт компьютерных исследований, 2004.

А. К. Петренко, О. Л. Петренко, В. В. Рубанов Москва,
Институт системного программирования РАН

Проект: Центр верификации ОС Linux
http://ispras.linux-foundation.org, http://linuxtesting.ru

Создание открытой образовательной среды на основе открытых проектов

Аннотация

В докладе обсуждаются принципы открытого образования как основного прогрессивного вида образовательного процесса в современном обществе. Показывается, как использование проектов по разработке программного обеспечения с открытым кодом позволяет естественным образом реализовать эти принципы в обучении студентов программной инженерии. Приводятся примеры организации открытой образовательной среды с вовлечением студентов в такие проекты на кафедрах системного программирования МФТИ и МГУ на базе ИСП РАН.

Открытая образовательная среда

Система образования должна соответствовать потребностям и тенденциям развития общества. Традиционная система профессионального образования, в основном, строилась на передаче студентам конкретных знаний с тем, чтобы они могли их непосредственно использовать в дальнейшей практической работе. Однако опыт показывает, что в условиях интенсивного технологического развития при таком подходе к образованию возникают трудности, так как конкретные технические знания быстро устаревают и важнейшими навыками для эффективной деятельности становятся умение адаптировать и пополнять имеющиеся знания для нужд конкретных проектов, умение принимать решения и планировать свою деятельность и т. п.

Схематично общие умения для ИКТ-специалистов можно представить следующим образом:

Таким образом, только узкопрофессиональных умений недостаточно для успешной работы специалиста. Меморандум Европейской Комиссии о непрерывном обучении подчеркивает важность таких социальных умений, как умение действовать уверенно, умение направ-

лять собственные действия на достижение нужных результатов и умение принимать рискованные решения, а также таких когнитивных навыков, как способность учиться, приспосабливаться к изменяющемуся окружению, быстро воспринимать необходимые новые навыки и находить нужную информацию в широком информационном потоке, обрушивающемся на каждого включенного в общественную жизнь человека.

Признание этих принципов непрерывного образования побуждает к активному переходу к так называемой *открытой образовательной системе*, ориентированной не столько на передачу обучающимся некоторого объёма знаний, сколько на воспитание самостоятельно целеполагающего, самообучающегося индивида, способного к эффективному взаимодействию с постоянно меняющимся окружением — индивида, который *умеет учиться*.

В такой системе образовательное пространство открыто для студента, ему видны возможные траектории собственного становления и потому становится возможным обсуждение средств и методов достижения индивидуальных целей. Одной из важнейших характеристик открытого образования является самостоятельное принятие студентом решений и ответственности за них.

Основными принципами открытого образования можно считать:

- индивидуализация обучения;
- ответственность за собственные успехи;
- сотрудничество;
- направленность на непрерывное образование.

Переход к открытому образованию может осуществляться следующим образом:

- Необходимо научить студента быть самостоятельным в обучении. Для этого обучение непременно должно быть активным.
- Студенты должны быть вовлечены в процесс живого межиндивидуального общения.
- Начало и конец каждого участка образовательного пути должны быть максимально индивидуализированы.
- Необходимо ввести в обучение этап рефлексии.

Важным шагом в сторону отрытого образования является создание специальной *образовательной среды*, которая подталкивает студента к включению в социальные институты изучаемой области и к активному взаимодействию внутри них, обеспечивает наличие множества различных возможностей в реализации учебного процесса и свободу их выбора. На важность такой среды для качественного обучения студентов указывал еще академик П. Л. Капица в своем письме Сталину в 1946 году о создании Физтеха: «Ведение воспитания с первых же шагов в атмосфере технических исследований и конструктивного творчества с использованием для этого лучших лабораторий страны».

Открытое ПО в открытом образовании

Использование программного обеспечения (ПО) с открытым исходным кодом может служить отличной основой для построения открытой образовательной среды в области программной инженерии. В частности, это позволяет разработчикам учебного курса получить практически для любой темы в данной дисциплине следующие инструменты:

- Готовый учебный материал: требования к ПО, само ПО, спецификации его архитектуры, исходный код, уже решённые и все

ещё актуальные проблемы и дефекты, возникшие в ходе его разработки.
- Готовую инфраструктуру для ведения практических занятий: реальные проекты по разработке ПО, доступное общение с реальными командами разработчиков.
- Возможность создания сильной мотивации для студентов в их практической деятельности, связанной с их участием в реальных проектах, имеющих большую общественную значимость.

Разнообразие программного обеспечения с открытым исходным кодом позволяет предоставить студентам большую свободу выбора при построении индивидуальной траектории освоения материала в рамках широкого учебного плана по изучению программной инженерии. Материалы открытых проектов служат отличной доступной «базой знаний» в этой области, своего рода аналогом классической публичной библиотеки, благодаря которой можно повысить активность и самостоятельность студентов в учебном процессе, а также получить больше возможности для индивидуализации образовательного процесса. Кроме того, предоставляя студентам свободу выбора предметной области, языков программирования и применяемых технологий в рамках изучения отдельной темы, можно повысить личную заинтересованность и мотивированность, а также уровень ответственности за самостоятельно сделанный выбор.

Открытое образование в ИСП РАН

В последние годы проекты по разработке свободного ПО стали важной частью учебного процесса на кафедрах системного программирования МФТИ и МГУ на базе Института системного программирования РАН. В рамках обязательной научно-исследовательской работы каждый студент обязан участвовать в каком-нибудь отрытом проекте на свой выбор (обычно студенты вовлекаются в собственные проекты Института, но могут выбрать и любой другой публичный проект). В рамках такого участия студентом обычно выполняется разработка отдельных улучшений в реальном ПО от предложений новых требований, новых архитектурных решений до их реализации, тестирования и написания технической документации.

Однако одного участия в открытом проекте для достижения целей открытого образования недостаточно. Для того чтобы использовать

такие проекты как педагогический инструмент, мы используем их в качестве примеров при изучении курса программной инженерии. Такой симбиоз позволяет эффективно реализовать принципы открытого образования. Периодически для студентов проводятся коллоквиумы, на которых в форме свободной беседы обсуждается понимание студентом материалов как собственно самого теоретического курса, так и его связи с практической деятельностью в конкретном проекте. На коллоквиуме студент представляет не только свою конкретную работу, но и проект в целом, в этой роли он обязан отвечать за все аспекты проекта: цели, сроки, качество, бизнес-перспективы и т. д. На первых коллоквиумах большинство студентов находятся в полной растерянности, так как в традиционной системе обучения они привыкают к тому, что их учат и им ставят задачи, здесь же они начинают понимать многообразие проблем в реальном проекте и важность проактивной позиции участников, что позволяет усваивать принципы программной инженерии на существенно более высоком уровне.

А. В. Гришин, Ю. Э. Ионан, О. Г. Крючкова Брянск,
Образовательный консорциум «Среднерусский университет»,
Брянский открытый институт управления и бизнеса

О проблемах перехода ВУЗа на свободное ПО при условии сохранения штатного расписания

В связи с необходимостью перехода на лицензионное программное обеспечение (ПО), инициированного руководством образовательного консорциума «Среднерусский университет», перед информационно аналитическим центром (ИАЦ) встал вопрос выбора пути решения данной задачи — покупка Microsoft Windows (легализация уже существующей организации труда) или переход на свободное ПО.

Учитывая экономические, юридические и технические аспекты, было принято решение о миграции на GNU/Linux. Из-за казусов российской юриспруденции нельзя использовать свободное ПО без формального правообладания лицензионной копией. В связи с этим было принято решение о сотрудничестве с компанией ALT Linux, предоставляющей сам дистрибутив операционной системы (ОС) и берущей на себя решение вопросов, связанных с формальным юридическим

требованием государства о легальности использования дистрибутива. Дополнительной мотивацией перехода на свободное ПО была надежность платформы Linux (отсутствие вирусов, постоянный контроль линукс-сообществом уязвимостей системы и работа над их устранением). Также учитывалось, что управление системой зависит от квалификации администратора больше, чем при администрировании систем семейства MS Windows.

Следующим шагом нашей деятельности стала разработка и планирование этапов перехода на свободное ПО.

Первый этап — подготовительный (информационный), на котором происходила постановка цели и задач, а также изучение предмета внедрения техническим персоналом ИАЦ, владеющим общими знаниями администрирования операционных систем.

Исходя из того, что потребности персонала ограничиваются использованием офисных программ, выбор был сделан с учетом требований к системным ресурсам компьютеров, поэтому в качестве среды рабочего стола была выбрана XFCE; остальное ПО было также подобрано под нересурсоемкий набор элементов интерфейса для X Window System — GTK.

На втором этапе (внедрения) организуется установка системы в нерабочее время, совместное использование GNU/Linux и Microsoft Windows и проведение тренингов для персонала. Совместное использование двух систем необходимо для выявления степени функциональности будущей системы и постановки новых задач для ее совершенствования.

На третьем этапе происходит отказ от нелегального ПО. Как только персонал становится способным к выполнению своих задачи в ОС Linux и обходиться только этой системой, происходит удаление раздела с ОС Windows.

Данный план не касается перевода учебного процесса на свободное ПО, но и не противоречит такому переходу впоследствии. Это задача будущего.

Хотя сейчас мы находимся на втором этапе внедрения ОС Linux, мы уже пытаемся аккумулировать и обобщать опыт работы. Достаточная квалификация сотрудников ИАЦ позволила не заострять внимание на технической реализации миграции. Основные проблемы связаны с методической стороной перехода на свободное ПО. Недостаточно установить ПО и обеспечить его техническую поддержку, важно научить людей профессионально пользоваться новой системой. В

связи с этим был подготовлен преподаватель и разработан 10-часовой тренинг для персонала.

Были выявлены следующие группы проблем при проведении тренингов:

1. Отбор содержания осваиваемого материала.
2. Психологический барьер освоения новой системы, вызванный изменениями условий труда.
3. Недостаток информационной грамотности персонала.

Таким образом, в статье были рассмотрены этапы миграции на GNU/Linux в рамках ВУЗа при условии сохранения штатного расписания работы персонала. Перечислены основные методические проблемы возникающие при таком переходе.

Л. А. Татарникова Томск,
ОЦ «Школьный университет ТУСУР»

Проект: http://itdrom.com, http://spo.tomsk.ru

Использование свободного ПО в учебном процессе: разработка, внедрение, методическое сопровождение

Особенности разработки и перехода на СПО учебных заведений Томской области. Совместная работа ОЦ «Школьный университет ТУСУР», Регионального центра развития образования, ООО «Интрайс» над проектами по внедрению и сопровождению СПО.

История внедрения свободного программного обеспечения в Томской области совсем ещё молода, но, как хорошая детективная история, уже радует динамичностью своего развития и в какой-то мере непредсказуемостью.

А начиналось всё немногим более года назад, когда осенью 2006 года образовательный центр «Школьный университет ТУСУР» принял решение о разработке учебных курсов, ориентированых на СПО.

Школьный университет — это образовательный центр, получивший статус Федеральной экспериментальной площадки. Школьный университет помогает организовать профильную IT-подготовку школьников, предоставляя в учебные заведения УМК (Учебно-методические комплекты, представляющие собой набор: программа, учеб-

ное пособие, электронный практикум, контрольные работы, методические рекомендации для учителя, дополнительные материалы) и осуществляя методическое сопровождение учебного процесса.

До названного времени в программы Школьного университета входили курсы, направленные на использование привычного всем нам «небесплатного» ПО. Одним из новых курсов, создание которых было запланировано на 2006–2007 г., является курс Компьютерной графики, который решено было ориентировать на использование приложений Gimp и Inkscape. Летом 2007 года этот курс успешно запущен в действие (`http://itdrom.com/files/demo/komp_dezing/design/`).

Кроме этого, в текущем году ведутся разработки нескольких «основополагающих» курсов для профильной информатики. Прежде всего, это джентльменский набор, без которого ни один пользователь не будет чувствовать себя уютно — пакет OpenOffice.org: Writer, Calc, Impress и Base. Также ведётся переработка курса программирования на языке Pascal, ориентированного на использование среды разработки Free Pascal.

С сентября 2007 года работа Школьного университета по внедрению СПО приобрела дополнительное направление. Но рассказ об этом следует начать с другой стороны.

В феврале 2007 года на базе РЦРО (Региональный центр развития образования) приказом Департамента образования Томской области была создана Рабочая группа, одной из задач которой является внедрение и поддержка СПО в учебных заведениях Томской области.

В принципе, для увлечённого учителя информатики вопрос перехода на другую систему, например Linux, не является таким уж большим камнем преткновения — приобрёл дистрибутив, установил на паре компьютеров, разобрался, потом приобрёл другой дистрибутив... И в результате лет эдак через 5 он будет готов дать исчерпывающую справку по любому сколько-нибудь доступному дистрибутиву. Долго?

По этой-то причине около года назад участники Рабочей группы «разобрали» дистрибутивы Linux для изучения их особенностей и возможности применения в качестве «школьной» системы. Стоит отметить, что участники группы достаточно далеко разнесены географически, общение в основном происходит по Интернету (на одном из IRC-каналов), но, наверное, это являлось огромной положительной стороной, поскольку позволяло оперативно советоваться, сравнивать, помогать друг другу. В результате жесточайшей борьбы буквально за пару месяцев было отобрано 5 дистрибутивов.

После этого Рабочая группа провела ряд выездных семинаров, посвящённых работе Районных ресурсных центров и вопросам внедрения СПО в школы.

Примерно в это же время в ТГПУ (Томском государственном педагогическом университете) заговорили о переходе на СПО, и не просто заговорили, а начали этот самый переход, постепенно заменяя на учебных компьютерах Windows на Linux.

Дальше назрела необходимость обобщить опыт и получить уже более систематизированные знания от профессионалов, благо появился свой выстраданный опыт и накопилась масса вопросов. И в апреле 2007 года Томске на базе Регионального центра развития образования прошли уникальные по тематике, организации, содержанию и реализации курсы повышения квалификации «Основы операционной системы GNU/Linux» (72 уч. ч.).

Анализ сложившейся в томском образовании ситуации, информация о которой поступала из первых рук — от школьных учителей, привёл к мысли, что Сибирским Афинам негоже отставать, более того, у нас есть потенциал, позволяющий серьёзно опередить проявление назревающих проблем. И летом 2007 года Администрация области с подачи ТГПУ объявила конкурс. В рамках контракта необходимо собрать дистрибутив (по принципу «запустил — и работай»), обеспечить подготовку и проведение курсов повышения квалификации, разработать и осуществить широкую поддержку заинтересованных слушателей.

Неожиданно для всех в конкурсе победила ООО Интрайс, практически неизвестная в IT-кругах области. К разработке дистрибутива компания подошла тоже с уникальных позиций: уже первые, сырые, варианты дистрибутива были выложены на «всеобщее тестирование» на сайте проекта (`http://spo.tomsk.ru`), одновременно проводились курсы повышения квалификации для школьных учителей. Кроме того, компания Интрайс, будучи довольно старым «соратником» Школьного университета, поручила ему разработать открытый обучающий курс — электронный практикум, который должен помочь учителям даже самых дальних школ, зачастую не имеющих возможности «вживую» получить поддержку, освоить за кратчайшие сроки операционную систему и необходимые для учебного процесса приложения.

В настоящее время активно работает сайт проекта, в январе в разделе поддержки началась публикация уроков электронного практи-

кума (`http://kubuntu.intrice.ru/training/`). Можно скачать как CD-, так и DVD-версию дистрибутива. DVD-версия рассчитана на то, чтобы можно было получить начальный, достаточно обширный пакет приложений даже без выхода в Интернет. В целом можно сказать, что проект активно развивается и уже имеет реальные результаты.

Главным, на мой взгляд, самым значительным, результатом всей этой работы является свобода выбора, которая появляется у учителя. Теперь он может не бояться «страшной» для обычного неискушённого пользователя системы Linux, может сэкономить время, получив готовый «начальный» комплект для работы.

Э. Б. Хайруллов, А. В. Коноплянов
Димитровград,
ООО «Открытые решения»

Проект управления информационными ресурсами школ г. Димитровграда

Требования, выдвигаемые к учебному процессу, в настоящее время просто невозможны без доступа в Интернет. Решение должно быть универсальным для учебных заведений и требовать минимальных материальных затрат, инфраструктура школы должна обеспечивать в полной мере учебный процесс. Управлением Образования была поставлена задача — решить данную проблему. В настоящее время, когда компьютерные технологии развиваются быстрыми темпами, в рамках программы «Интернет для школьников» во все школы города был проведен высокоскоростной интернет по технологии ADSL, но до рабочих мест школьников он так и не был доведен.

После внедрения в нашей области программы информатизации школ, состояние компьютерного парка в образовательных учереждениях улучшилось, на данный момент все школы имеют хотя бы по одному компьютерному классу. Поэтому на первом этапе была поставлена задача «довести интернет до школьников».

При разработке проекта были выдвинуты следующие требования к системе:

1. Все компоненты системы должны быть полностью лицензионными.
2. Система должна обеспечивать безопасность школьной сети и ограничивать контент потребляемого трафика.

3. Генерировать отчеты по программному наполнению всего парка школьных машин.
4. Генерировать отчеты по аппаратной части.
5. Генерировать отчеты о количестве потребляемого трафика в реальном времени.

Предлагается опыт по внедрению программного обеспечения с открытым кодом в учебный процесс получения общего образования (школы). Интерес данного опыта заключается в том, что решения на базе открытого программного обеспечения внедряются в условиях интеграции в информационную систему, построенную до внедрения целиком на решениях от Microsoft, после интеграции пропретарные продукты по-прежнему составляют подавляющее большинство программ, эксплуатирующихся в организации.

ООО «Открытые Решения» в начале прошлого учебного года начало внедрение в учебный процесс в школах города собственной программы обучения школьников, построенной на свободном программном обеспечении. Причём цель «поголовного» перехода на свободное программное обеспечение не ставилась, т. к. авторы осознавали проблемы, которые возникнут при попытке изменить привычную среду работы для учителей. Вследствие этого программа составлена на базе программных продуктов, функционирующих как в MS Windows, так и в Linux.

На сегодняшний день по согласованию с МО учителей информатики г. Димитровграда было принято решение установить дистрибутив Alt Linux Junior во всех компьютерных классах, чтобы учителя могли высказать свои пожелания и замечания по работе с ним. Из 20 школ на данный момент 17 находятся под обслуживанием ООО «Открытые Решения». В каждой школе установлен Интернет-сервер (шлюз), который покрывает все требования выдвинутые к системе. В перспективе развития планируется создать единый метакаталог на основе OpenLDAP.

На основании опыта работы справедливыми считаем следующие выводы и предложения:

1. Вести учебный процесс полностью на открытом программном обеспечении можно, что с успехом подтвердили несколько учителей информатики.
2. Необходимо централизовано заняться проблемой поставки в школы свободного программного обеспечения, тогда разрабо-

танная официальная программа преподавания дала бы толчок к внедрению Linux.
3. Необходимо постоянно поднимать вопрос у руководства Управления Образования к решению проблем лицензирования школьного программного обеспечения.
4. Необходимо уделять больше внимания провинциальным школам, где остро стоят проблемы не только финансирования, но и кадров.
5. Ввести в программу специализированных учебных заведений изучение свободных программных продуктов.

И. Чубин

Киев, Украина,
Учебный центр «Сетевые технологии»

Система ведения журналов лабораторных работ LiLaLo

LiLaLo позволяет автоматически фиксировать полный ход работы с терминалом Unix-системы, включая текст команд и результат их выполнения, текущий каталог вызова, время вызова и множество других. Кроме записи работы с терминалом, автоматически фиксируются изменения, сделанные с помощью текстового редактора. Полученные в ходе записи данные могут быть представлены в формате XML, пригодном для дальнейшего анализа хода работы, создания заготовок документации и сценариев командного интерпретатора.

В процессе работы в консоли Unix/Linux-системы, будь то непосредственное выполнение задач администрирования, экспериментирование с целью найти и описать решение какой-то задачи или самообучения, демонстрация приёмов работы на живых примерах или что-то другое, часто возникает необходимость зафиксировать происходящий в консоли процесс.

Записи нужны для того, чтобы или просто использовать при попытке повторить те же действия, но в другой раз, или потом, доработав и снабдив необходимыми комментариями и ссылками, превратить их в полноценную документацию.

Запись обычно выполняется одним из нескольких способов:

- запись вручную на бумаге;
- запись вручную в электронном виде;
- запись путём копирования мышью в текстовый редактор;
- с применением программы script.

Каждый из этих способов имеет собственные достоинства и недостатки. Преимущества *Записи вручную на бумаге* в том, что она может использоваться для записи команд, которые выполняются на другом компьютере или демонстрируются с помощью проектора. Недостатки:

- долго;
- неудобно;
- может содержать ошибки;
- непригодна к дальнейшей электронной обработке.

Запись вручную в электронном виде обладает теми же достоинствами и недостатками, что и в случае ручной записи, но результат записи поддаётся дальнейшей электронной обработке.

Запись путём копирования в текстовый редактор имеет то преимущество, что копирование выполняется быстро и без ошибок. Недостатком является то, что копирование в текстовый редактор требует дополнительных действий и, что особенно важно при обучении, — оно невозможно во время демонстрации команд.

В Unix существует *программа script*, которая автоматически выполняет запись всего происходящего на терминале, где она запущена.

Преимущества использования программы script для записи:

- запись производится прозрачно;
- может выполняться во время демонстрации;
- запись не содержит ошибок.

Недостатки:

- необходимость в обработке после завершения записи;
- запись может производиться только для действий, выполняемых непосредственно в командной строке.

Предлагаемое решение — система ведения журналов работы с терминалом Unix-системы LiLaLo[1] — свободно от всех перечисленных выше недостатков и обладает рядом преимуществ.

LiLaLo использует для записи программу script. Однако, в отличие от программы script в чистом виде, во время записи фиксируются не только команды и результат их работы, но и множество дополнительной информации о командах. Это позволяет в дальнейшем более полно реконструировать ход работы. Кроме того, информация, которую LiLaLo автоматически записывает при ведении журнала, позволяет выполнять анализ хода работы и автоматически создавать заготовки для сценариев командного интерпретатора.

Автоматическая запись дополнительной информации о командных строках возможна за счёт модификации приглашения командного интерпретатора. Хотя визуально это (практически) никак не заметно, приглашение командного интерпретатора модифицируется, и в него, в скрытом виде, добавляется несколько параметров, характеризующих команду, которая набирается в этом приглашении и будет выполнена. В их числе:

- текущий каталог, из которого производится вызов команды;
- время;
- код завершения предыдущей команды и ряд других.

Помимо того, что производится запись всего хода работы в командной строке, автоматически фиксируются все изменения в файлах, сделанные с помощью текстового редактора. Есть возможность делать скриншоты и показывать в журнале окна, которые, возможно, имеют непосредственное отношение к производимым в консоли действиям.

Записанные данные хранятся в формате программы script, то есть, фактически, непосредственно в виде набора команд терминалу. Они могут быть обработаны и представлены в структурированной форме, в виде XML-файла, который в дальнейшем может быть либо преобразован в HTML-файл и визуализирован, либо может просто попасть в хранилище.

Анализ терминального скрипта, преобразование его в XML и визуализация при помощи веб-интерфейса выполняется в реальном времени и без всякого дополнительного участия пользователя.

Литература

[1] `http://xgu.ru/wiki/LiLaLo` — страница проекта LiLaLo

В. Р. Роганов, И. Н. Трушкин, А. А. Трушкина Пенза,
Пензенский Государственный Университет

Проект: Программная среда «Арифметик»

Разработка единой информационной платформы

В настоящее время в большинстве компаний существует тенденция использования труда программистов для подготовки документов, помощи неквалифицированным пользователям и простейшей работе по ремонту и обслуживанию компьютеров и офисной техники. Это дискредитирует само понятие «программист».

В то же время существует множество задач, которые должны решаться и решаются программистами, например разработка нового программного обеспечения и модернизация существующего. При решении поставленных задач имеет место объективная проблема — отсутствие единой информационной платформы, которая позволила бы решать следующие задачи:

- Визуальное проектирование, позволяющее абстрагироваться от возможностей того или иного языка программирования и в конкретный момент времени видеть наглядное схематичное изображение как всей системы в целом, так и отдельных алгоритмов.
- Управление базами данных, допускающее также визуальное проектирование.
- Автоматическая проверка корректности и работоспособности создаваемых систем, используя разработанные критерии качества.

В результате фирмы поставлены перед необходимостью покупать новые версии широко известных программных продуктов из-за того, что в них новой и нужной является одна утилита, которой не было в старом варианте, или создавать и поддерживать собственную разработку.

В Специальном проектно-конструкторском и технологическом бюро Института систем управления и информационной безопасности Пензенского Государственного Университета в целях создания новой программной среды (которая могла бы первоначально смягчить ряд

проблем при доработке лицензионного ПО, а на втором этапе создать необходимую среду для свободно распространяемого ПО) была поставлена и успешно решена задача использования существующих программных средств вместе с новой средой, обеспечивающей возможность модернизации существующего программного обеспечения методом поиска точек входа и обеспечения визуального сопровождения взаимодействия разрабатываемого программного модуля с уже имеющимся программным обеспечением. В итоге мы получаем, возможно, не оптимальный код решения задачи, но максимально облегчаем труд программистов.

Такой средой визуального проектирования является «Арифметик» (свидетельство об отраслевой регистрации разработки ОФАП № 6497 от 29 июня 2006 г). Система построена таким образом, что существует возможность группировки модулей и подсистем для решения конкретных прикладных задач, а также их объединения в рамках единого интерфейса, являющегося основной частью среды визуальной разработки.

Также среда имеет в своём составе инструменты для интеграции с уже существующими программными продуктами и внедрения пользовательских модулей в готовые приложения. Предусмотрена возможность расширения инструментария программ без внесения изменений в их код.

Разработанный язык программирования данной среды отличается от других языков программирования тем, что его интерпретатор разделён на 3 независимых блока: язык действий, язык формул и подсистему работы с графикой. Каждая из этих частей может взаимодействовать со своим классом объектов и/или функций, которые по-своему обрабатывают текст, находящийся в их области действия — контейнерами. Контейнеры могут быть как полностью автономными, так и зависимыми, т. е. переводящими исходный текст программы в вид, понятный другим контейнерам ключевых слов. Контейнеры могут представлять собой отдельные модули, являющиеся разработкой сторонних производителей. Благодаря этому их свойству возможно создание таких контейнеров, которые позволяли бы использовать практически любые языки разработки в рамках одного проекта и листинга. При этом допускается даже дифференцированный подход к обработке интерпретатором исходных текстов. Ввиду широты спектра задач, решаемых на основе создаваемой системы, среда включает в себя несколько классов контейнеров. Это класс компилирующих кон-

тейнеров реального времени и класс интерпретирующих контейнеров, допускающих написание на их основе самомодифицирующихся программ.

Предложенная среда имеет средства для организации взаимодействия автономных модулей обработки данных с программами, средства управления работой интерфейсов межпрограммного взаимодействия и позволяет осуществлять универсальную и управляемую интеграцию различных программных средств.

М. М. Якшин

Москва,
МГТУ им. Н. Э. Баумана

Проект: Электронный Университет

http://www.bmstu.ru

Интеграция различных компонент автоматизации университета в МГТУ им. Н. Э. Баумана с помощью веб-сервисов

Аннотация

Система автоматизации МГТУ им. Н. Э. Баумана представляет собой сложную многокомпонентную сеть сервисов, общающихся между собой. Доклад посвящён проектированию и организации такой сети, указаны выработанные стандарты написания веб-сервисов, программные интерфейсы взаимодействия. Освещены типовые проблемы, встречающиеся при такой разработке, и найденные варианты решений.

С 2003 года в МГТУ им. Баумана проводится программа автоматизации деятельности ВУЗа. В частности, эта программа предусматривает автоматизацию работы таких отделов, как отдел кадров, деканаты, отдел по работе с иностранными студентами и аспирантами, бухгалтерия и т. п. Было принято решение сегментировать создаваемую систему по отдельным, сравнительно небольшим классам задач и создавать не одну монолитную, а много систем, связанных между собой. Тем самым используется модульный подход, что позволяет вводить системы в строй по мере написания их сравнительно небольших частей.

Для связи создаваемых систем между собой была выбрана технология веб-сервисов SOAP[2]. Эта технология обладает как многочисленными плюсами, так и минусами, что будет подробнее показано

ниже. Тем не менее, в целом результаты внедрения веб-сервисов можно считать положительными.

Так как первой в рамках программы была реализована система Контингент[1], обрабатывающая данные о студентах, обучающихся в МГТУ им. Баумана, логично было начать проектирование структуры веб-сервисов с неё

В результате проектирования были выработаны следующие соглашения, которые позволяют описать модель данных, представленную в Контингент, а также без сильных ограничений представить данные других систем МГТУ.

Синтаксические соглашения

Все идентификаторы — имена типов (которые в дальнейшем становятся классами) называются с большой буквы в `UpperCamelCase`; идентификаторы — имена атрибутов (в дальнейшем становятся атрибутами классов) и методов называются с маленькой буквы в `lowerCamelCase`. Подобные соглашения приняты в Java[5], Qt[6] и многих других средах.

Все идентификаторы представляют собой грамотный общеупотребительный перевод русского термина на английский язык. Исключение: middleName — отчество у человека (предложенные аналоги: fatherName — не достаточно распространённый перевод, patronymical — редко применяется на практике английского языка, сложно для написания и запоминания).

Иерархия и структура типов данных

Все типы данных (классов), которые соответствуют некоей сущности в реальном мире, имеют общего предка (возможно, непрямого) — класс BaseObject. Это класс, имеющий идентификатор сущности (`id`) и её имя (`name`). Семантика поля «имя» меняется в зависимости от того объекта, который представляет эта сущность и, как правило, формируется для целей показа пользователю на экране, не разбираясь в сложной структуре объекта.

Идентификация сущностей

Каждая сущность, с которой работает система, имеет идентификатор, — стандартный 16-байтовый UUID. Механизм формирования

UUID позволяет определять тип сущности по её UUID, по первым байтам.

Информацию о любой одной сущности (с некоторыми ограничениями реализации) можно получить с помощью метода get, который вернёт объект, наиболее полно представляющий её.

Типы данных

В SOAP можно разделить типы данных по предназначению.

Типы данных для запросов-ответов. Для большинства методов их запросы и ответы оформлены в виде отдельных типов данных; эти типы имеют названия, сформированные как имяМетодаRequest и имяМетодаResponse. Это соглашение сильно упрощает написание клиентов для веб-сервиса в таких средах, как .NET и Java, которые сами следуют подобным неформальным соглашениям.

Общие типы. Типы данных, действительно представляющие некие классы. Введены следующие типы данных в соответствии с иерархией наследования:

```
* BaseObject
  * Classifier
    * ProfessionClassifier
  * Person
    * Student
      * StudentInOrder
  * Order
    * ContingentOrder
  * Group
    * GroupInOrder
  * Department
  * Faculty
```

Классификаторы. Некий достаточно общий тип данных в веб-сервисе, используемый во многих местах. Они представляют собой справочники, где в соответствие UUID (например, `2c4fdb64-0c14-4dd1-81c1-000000000001`) ставится строка-название

(мужской) и, опционально, код по общероссийскому классификатору (`01 1`).

Методы

При проектировании архитектуры веб-сервисов был принят подход CRUD[3][4], с оговоркой, что всё взаимодействие сервисов будет осуществляться по pull-схеме и все предоставляемые сервисы работают в режиме read-only. Таким образом, всё множество методов сводится к всего двум подходам: либо по UUID объекта получить исчерпывающую информацию об объекте (метод `get`), либо искать объект в базе путем указания некоторых признаков (методы `list*`).

Особенности реализации на разных платформах

Вследствие различных особенностей реализации SOAP на разных платформах и неких отклонений от стандарта в WSDL и самом вебсервисе сделаны некие дополнительные доработки. В качестве клиентов для веб-севриса были протестированы платформы .NET, Java Axis/WSIF, PHP5 SOAP, Soap4R, Perl SOAP/SOAP::Lite.

PHP5 SOAP, во-первых, имеет проблемы с обработкой и сериализацией namespaces. Разные версии ведут себя по-разному, закономерностей выявить не удалось. *Workaround в сервере*: у всех элементов в запросе сбрасывается namespace, и таким образом принимаются любые namespaces. Во-вторых, имеются проблемы с отправкой опциональных полей. В случае передачи в конструктор Request-объекта пустых строк — отправляет XML вида `<name></name>`, в случае передачи в конструктор NULL'ов — отправляет `<name/>` (что эквивалентно). *Workaround в сервере*: во всех list-методах введена проверка на пустоту принимаемых критериев — если критерий пустой, то считается, что он не задан.

.NET требует присутствия в WSDL в схеме данных атрибута `elementFormDefault="qualified"`. В случае отсутствия некорректно работает с namespaces и в результате не может ни сформулировать запрос, ни десериализовать ответ.

Ruby SOAP4R имеет тенденцию выносить все поля, собранные со всех предков класса, в самый крайний класс. В случае WSDL «Контингент» этими классами оказываются классы запросов и ответов. Не принципиально для программирования клиента, но перегружает

генерируемые заглушки массой лишних полей и mappings в схеме десериализации.

Результаты

Спроектированный веб-сервис был реализован в рамках системы Контингент, эффективно предоставляя доступ к имеющимся данным о студентах для других систем. Была разработана концепция проектирования веб-сервисов для данных внутри МГТУ, которая может быть применена для всех взаимодействий автоматизированных систем МГТУ, которые оперируют объектами, схожими с объектами системы Контингент. Применение таких правил дает достаточно гибкую схему, однозначную (по сравнению с достаточно широкими стандартами W3C) генерацию WSDL и ускоренное внедрение веб-сервисов в гетерогенной среде.

Литература

[1] *Якшин М. М.* Построение системы автоматизации университета в МГТУ им. Н.Э.Баумана // Вторая конференция «Свободное програмное обеспечение в высшей школе» (27-28 января 2007, Переславль-Залесский). — М. 2007. `http://heap.altlinux.ru/pereslavl2007/yakshin/abstract.html`

[2] XML Protocol Working Group, Web Services `http://www.w3.org/2000/xp/Group/`

[3] *Kilov, H.* From semantic to object-oriented data modeling // First International Conference on System Integration, 1990. — С. 385–393.

[4] *Kilov, H.* Business Specifications: The Key to Successful Software Engineering. — New Jersey, Prentice Hall, 1998. — ISBN 978-0130798442.

[5] Code Conventions for the Java Programming Language. — Sun Microsystems, 1999. — `http://java.sun.com/docs/codeconv/html/CodeConventions.doc8.html#367`

[6] Qt Developer Notes: Naming Guidelines. `http://trolltech.com/developer/notes/naming`

Л. В. Дмитриев
Москва,
ВМиК МГУ им. М. В. Ломоносова

Об электронном обучении на базе свободного ПО

В лаборатории открытых информационных технологий факультета ВМиК МГУ им. М. В. Ломоносова проводятся разработки в области электронного обучения (e-learning). Эти работы положили начало созданию Виртуального национального университета ИТ-образования [1].

Развитие электронного обучения привело к необходимости в стандартах. Много передовых разработок различных организаций стандартизации вобрал в себя стандарт SCORM (Sharable Content Object Reference Model) группы ADL (Advanced Distributed Learning) [2], созданной правительством США. В нем описаны требования к формату представления учебных материалов и обучающей системе (среде выполнения). Стандартизация в перспективе должна привести к интероперабельности и возможности повторного использования учебных материалов, но на практике при использовании различных средств подготовки и LMS (Learning Management System — система управления обучением/система дистанционного обучения) имеются проблемы совместимости.

Процесс электронного обучения можно разделить на два основных этапа: подготовку учебных курсов и собственно обучение. Для подготовки курсов мы используем Reload Editor, а обучение проводится в среде LMS Sakai. Эти продукты и минимально необходимое для их функционирования ПО распространяются свободно.

Reload Editor был создан в рамках проекта RELOAD (Reusable eLearning Object Authoring & Delivery) [3]. Он написан на языке Java и позволяет создавать и редактировать учебные курсы в формате SCORM различных версий. Данное ПО распространяется свободно, исходный код проекта находится на `sourceforge.net`.

LMS Sakai разрабатывается Sakai Foundation [4], основанным группой ведущих университетов США. LMS Sakai написана на языке Java и имеет модульную архитектуру; распространяется под ECL (Educational Community License). Данная система активно развивается (в основном, для реализации потребностей учебного процесса американских университетов) и обладает большой функциональностью

помимо базовой поддержки SCORM. В Sakai реализована поддержка смешанного обучения (blended learning). Для работы с LMS в качестве обучаемого достаточно стандартных навыков использования Интернета.

Среди практических результатов следует отметить работу LMS Sakai под ОС FreeBSD, что не декларируется разработчиками. LMS с набором учебных курсов готовы к широкому использованию. Таким образом, приведен пример построения соответствующей международным стандартам системы электронного обучения на основе свободного ПО.

Литература

[1] *Сухомлин В. А.* Создание Виртуального национального университета ИТ-образования. М.: МАКС Пресс, 2007. 60 с.

[2] `http://www.adlnet.gov/`

[3] `http://www.reload.ac.uk/`

[4] `http://www.sakaiproject.org/`

А. С. Сивер

Москва,
ООО «Марафон»

Проект: MAXIMA
`http://community.livejournal.com/maxima_platform/`

Компьютерная алгебра MAXIMA как free Mathematica

Аннотация

Используя компьютерную алгебру MAXIMA и парсер синтаксиса Mathematica из старого пакета MockMma, оказалось возможным написать lisp-функцию, которая принимает программу на Mathematica и выполняет ее в среде MAXIMA (для очень небольшого подмножества функций).

В настоящее время современные компьютерные алгебры воспринимаются и используются как полноценные языки программирования.

MAXIMA уступает в проработанности своей архитектуры, но имеет те преимущества, что интегрирована с lisp-машиной и содержит богатую библиотеку вычислительных функций. Заменив в MAXIMA входной язык и схему вычислений на более проработанные, которые используются в Mathematica, можно надеятся, что на базе MAXIMA удасться разработать современную платформу для символьных научно-технических вычислений.

В настоящее время современные компьютерные алгебры воспринимаются и используются как полноценные языки программирования. Язык Mathematica является достаточно привлекательным для использования программистами-математиками для решения научно-технических задач, при решении которых требуется производить сложные математические преобразования или не простые преобразования над сложными структурами данных нечисловой природы. Однако для выполнения разработанных Mathematica-программ на конечном компьютере должна быть установлена система Mathematica. Это ограничивает варианты использования программ на Mathematica и число потенциальных пользователей этих программ.

Возникла идея, как сделать возможным выполнение Mathematica-программ (использующих хотя бы небольшой набор функций) без установленной системы Mathematica. Для этого можно использовать пакет MockMma `http://www.cs.berkeley.edu/~fateman/mma.mailer`, написанный более 10 лет назад программистом Richard Fateman, совместно с системой компьютерной алгебры MAXIMA.

В пакете MockMma (который сам написан, как и MAXIMA на common lisp) реализован парсер синтаксиса для Mathematica-выражений, который переводит выражения в lisp-выражение. Такие выражения можно достаточно легко перевести в lisp-выражения, которые являются внутренним представлением выражений в MAXIMA, после чего все вычисления можно выполнить с помощью MAXIMA (если все используемые функции реализованы и доступны).

Таким образом, программы на Mathematica, которые используют только функции, аналоги для которых есть в MAXIMA, можно вычислять с помощью MAXIMA. Конечно, при этом будет существовать некоторое количество исключительных случаев, когда в MAXIMA программа будет выдавать другой ответ, чем в Mathematica. Например, очевидно, программы не должны полагаться и использовать знания о конкретном виде промежуточных выражения, которые возникают в ходе вычислений в Mathematica? и проводить дальнейшие вы-

числения в зависимости от видов этих выражений: например, при расскытии скобок программа может взять некоторую часть выражения, которая при конкретном вычислении может оказаться разной при вычислении в Mathematica и MAXIMA.

Существует большой класс задач на программирование, для которых целесообразно использовать системы компьютерной алгебры. При этом сами системы компьютерной алгебры уже стали или должны стать частью индустрии программирования. Это накладывает некоторые достаточно серьезные требования к MAXIMA в том, что касается её языка и архитектуры. Однако, как мы увидели, входной язык MAXIMA можно достаточно просто заменить на более современный язык Mathematica. Возможно, это повысит интерес к MAXIMA как основы для разработки платформы для для символьных научно-технических вычислений.

В. О. Корепанов

Ижевск,
Удмуртский Государственный Университет

Свободное программное обеспечение и лабораторные работы

Аннотация

Основная тема доклада — использование принципов свободного программного обеспечения к оформлению лабораторных работ в учебных заведениях. Автор хочет показать, что поможет как студентам так и преподавателям и уровню подготовки специалистов.

В настоящее время для оформления лабораторных работ (и программ из курсовых проектов) нет чётких требований, такие требования появляются от совмещения требований факультета, требований кафедры и самих преподавателей курсов, связанных с программированием. Такой подход мало того, что скорее всего не полон, но часто и противоречив. С другой стороны, такие требования часто зависят не от потребностей сферы применения, а от личных симпатий преподавателей (хотя такие «симпатии» у опытных преподавателей обусловлены именно профессиональным опытом).

Но всё же, применение принципов свободного программного обеспечения для создания лабораторных работ было бы отличным шагом

как к умеренной унификации, так и к приближению программистских качеств студентов к качествам, требуемым на рынке.

Как и в реальной ситуации программирования, при создании лабораторных работ студентами не учитывается тот факт, что их программа может быть использована в будущем, а не предназначена только для одноразового использования: «работает — зачёт». Даже если не планируется дальнейшее использование другими студентами, сам студент может в будущем продолжить работу над этой программой. Но скорее всего здесь не стоит применять принципы свободного ПО повсюду, так как они рассчитаны на достаточно серьёзные проекты. Например, не нужно заставлять делать лабораторные типа «Сумма 10 чисел» в стиле свободного ПО, здесь больше времени уйдёт на оформление, да и повредит процессу обучения, так как такие задачи рассчитаны на азы программирования. А вот программы уровня калькулятора с деревьями решений — вполне уместный пример, а в курсовых проектах старших курсов это будет только плюсом.

Но в связи применением принципов свободного ПО появится проблема плагиата. Хотя для получения поддержки студенту будет выгоднее явно заявить об использовании чужих разработок. И главное, нужен высокий уровень преподавателя, чтобы выявлять подобные случаи, а также для проверки соответствия программ студента нормам свободного ПО. Для студентов нужно делать упор не на улучшении готовых программ-модулей, а на использовании их в оригинальных разработках. Здесь важен процесс написания, в котором ошибки студента играют не последнюю роль.

Облегчится процесс интеграции проектов, ведь часто применение готовых наработок продвигает решение гораздо дальше. Студентам можно будет явно использовать чужие наработки и получать помощь от разработчиков других проектов. Сама идея открытого ПО подразумевает продвинутые средства поддержки разработки программных систем и, следовательно, требует от разработчика не только знания языка программирования, но и целого класса разнородных инструментов, что в результате приводит к расширению кругозора и увеличению опыта. Опять же, не обязательно заставлять изучать такие инструменты на специальностях, не связанных с прямой разработкой ПО, где знание языка программирования нужно лишь изредка или только для узких программных решений, например, для разработки низкоуровневого ПО.

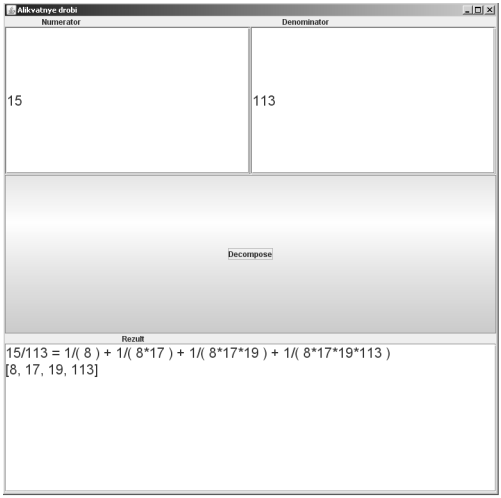


Рис. 1: Окно программы

Для демонстрации будет представлена лабораторная, сделанная для курса «История информатики» на языке программирования Java. Программа раскладывает обыкновенные дроби на египетские: сумму дробей, каждая из которых представляет собой дробь со знаменателем «1». Вид окна программы представлен на рис. 1.

Т. А. Воронина
Ижевск,
Удмуртский Государственный Университет

Проект: Облако Оорта

Облако Оорта

Аннотация

В работе создан учебно-познавательный анимационный фильм, иллюстрирующий гипотетический процесс образования комет Солнечной системы в Облаке Оорта. 3D-анимация выполнена с использованием свободного пакета Persistence of Vision Raytracer (POV-Ray).

Информационные технологии в XXI веке — необходимая составляющая всех областей человеческой деятельности. В любой сфере мы можем наблюдать повсеместную интеграцию вычислительных устройств, программ, электронных носителей информации.

Не является исключением и образовательная деятельность. Сейчас образование — это сложный технологический процесс, объединяющий в себе необходимость управления, планирования, применения психологических и педагогических способностей, внедрения инновационных методик. И сегодня как никогда остро представляется нам вопрос об использовании открытого программного обеспечения в образовании. Многие учебные заведения находятся перед выбором: так ли необходим переход на открытое ПО?

Такой переход ставит вопросы переориентирования квалификации всех специалистов в области информационных технологий. Впервые за последние 15 лет снова возникла необходимость в подготовке специалистов, которые владели бы самими технологиями, а не навыками работы в отдельных программных продуктах.

С другой стороны, существует и проблема совместимости форматов данных. Исходя из того, что цели образования мало созвучны с идеологией коммерческого распространения ПО, возникает серьезный вопрос выбора программных средств и методик использования их в сфере образования.

В разделе мультимедиа-технологий большую роль играет 3D-моделирование (направление машинной графики по созданию трёхмерных моделей). Студенты стремятся к изучению этой области. Но

успешно работать в анимационных 3D-средах не так легко. Причиной является хотя бы то, что интерактивный способ описания 3D-сцены не подходит в задачах анимации: для описания движения необходимо уравнение такого движения, тут не поможет интерфейс для создания объёмных геометрических тел при помощи указывающих устройств.

Имитационное моделирование — одна из наиболее интересных, но и одна из наиболее сложных задач информационных технологий. Компьютерные имитационные модели позволяют представлять физические процессы с любой степенью детализации и наглядности: для этого нужно лишь математическое описание процесса и средство его визуализации.

Одной из программ, позволяющей создавать такие наглядные модели, является пакет Persistent of Vision Raytracer (POV-Ray) [2]. Эта программа позволяет создавать статичные и анимационные графические модели при помощи языка, в который легко интегрируются математические модели, и обладает большими возможностями в области машинной 3D-графики.

В представляемой работе при помощи программной системы POV-Ray автором был создан учебно-познавательный анимационный фильм на астрономическую тематику. Тема выбрана не случайно, она позволяет изучить механизмы интеграции уже известных математических моделей движения небесных тел в код трёхмерной сцены.

В качестве тематики анимационного фильма был выбран гипотетический процесс образования комет Солнечной системы в Облаке Оорта. Что же такое Облако Оорта? Обратимся для этого к курсу астрономии[1].

Долгое время оставалось загадкой, откуда и почему появляются кометы. Приближаясь к Солнцу, ядра комет начинают испаряться, формируют хвосты, но дальнейшую судьбу этих небесных тел проследить очень сложно. Большинство астрономов предполагали, что комета приходит к Солнцу лишь один раз и затем навсегда покидает его окрестности. Однако позже ученые заметили, что некоторая часть комет имеет циклическую природу появления.

За последние годы появилась гипотеза о существовании так называемого Облака Оорта (облако названо именем ученого Яна Оорта, высказавшего данную гипотезу). В настоящее время многие факторы указывают на то, что на окраинах Солнечной системы существует скопление кометных и метеоритных тел. Оно и было названо Облаком Оорта.

Выполняя подготовку методических материалов для студентов и школьников, можно в тексте учебного пособия описать весь процесс зарождения, движения и распада комет. Однако намного нагляднее и понятнее показать, как это происходит, при помощи анимационного фильма. Применение 3D-анимаций при проведении занятий позволяет повысить качество усвоения предмета по сравнению с использованием традиционного учебника (хотя анимация, разумеется, не заменяет ни сам учебник, ни преподавателя).

Задача создания модели, иллюстрирующей движение кометы, не такая простая, как может показаться, для этого требуется специализированное ПО. Это касается таких аспектов, как создание множества кадров, обеспечивающих движение, возможность визуального представления небесных тел, создание оптических эффектов свечения, тени.

Сценарий фильма предполагает смену нескольких сцен, которые отражают процесс появления комет в Солнечной системе, и рассказывают об истории исследования данного вопроса. Применение визуальных моделей помогает в закреплении полученных знаний.

Представляемый анимационный фильм может быть использован на уроках астрономии для демонстрации движения комет, их происхождения и наглядного представления относительного расположения составляющих Солнечной системы.

Литература

[1] Malte Borges, Jürg Schumacher, and Torsten Redeker. *StarOffice 7*. Markt und Technik Verlag, 2003.

М. А. Ройтберг, В. В. Яковлев Пущино,
Институт математических проблем биологии РАН,
Пущинский Государственный Университет

Проект: Кумир http://www.infomir.ru

Конвертор КУМИР – C++: поддержка перехода от учебных к профессиональным системам программирования

Система программирования КуМир-2 (Комплект Учебных МИРов) предназначена для поддержки начальных курсов информатики и программирования в средней и высшей школе, основанных на методике, разработанной во второй половине 1980-х годов под руководством академика А. П. Ершова. В настоящее время КуМир-2 используется в начальном практикуме по программированию на механико-математическом факультете МГУ.

В работе представлен конвертор КСИ, преобразующий программы, созданные в КуМир-2, в программы на языке C++. Педагогическое назначение конвертора — демонстрация сходств и различий двух достаточно близких языков программирования, а также поддержка перехода от программирования в учебной среде к программированию в профессиональной среде.

Основные свойства конвертора

Максимальное сходство кода

Генерируемый конвертором код максимально сохраняет вид исходной программы. Каждой строке полученного кода на языке C++ ставится в соответствие строка на языке КуМир, оформленная в виде комментария. Это позволяет учителю объяснять особенности языка C++ на тех примерах программ, которые ученики, знакомые с КУМИРом хорошо знают. Строки служебного характера, которые обязаны присутствовать в программе на C++, но отсутствуют в КуМире, например инициализация массивов, помечены комментарием «ЭТА СТРОКА СОЗДАНА КОНВЕРТОРОМ».

Все используемые имена переменных и функций транслителируются в допустимые имена на латинице, при необходимости к ним до-

бавляются суффиксы вида «$_n$», где n — номер первого неиспользованного имени переменной или функции.

Для сохранения компактности кода операторы «ввод» и «вывод» заменяются на соответствующие функции «INPUT» и «PRINT».

Идентичность выполнения

Для большинства встроенных функций КуМир созданы (за исключением простейших, например *sin*) соответствующие функции на C++, которые по формату вызова полностью идентичны КуМир-функциям. Это обеспечивает идентичность выполнения программ как интепретатором КуМир, так и скомпилированными из C++ кода программами.

Скрытие особенностей работы с памятью в C++

В сгенерированных программах все действия по выделению памяти максимально возможно скрываются от пользователей. Инициализация массивов и их очистка (необходимость которой проверяется на этапе конвертации) заменяется на функции, которые генерируются конвертором и помещаются в отдельных файл.

Тип «лит» языка КуМир переводится в тип *std::string*, что освобождает код от операций выделения и перераспределения памяти под строки.

Данные особенности сгенерированного кода обеспечивают последовательность учебного процесса от объяснения простых программ до более сложных.

Ограничения конвертора

На данном этапе конвертор работает только с КуМир-программами, состоящими из одного программного модуля. Это ограничение будет снято в следующей версии конвертора.

Другим важным ограничением является работа со строками. Поскольку система КуМир-2 работает с Unicode-строками, а язык C++ работает с 8-битными строками, то не гарантируется идентичность работы при использовании русских букв в качестве значений литеральных переменных. Конвертор может сохранять вывод в одной из кодировок: «IBM866», «КОИ8-Р», «CP1251» или «UTF-8», но поскольку у

каждой из них есть свои особенности, то и поведение программ будет разным.

Особенности реализации

Конвертор является консольным приложением, которое может быть либо вызвано из системы КуМир-2, либо использовано как самостоятельное приложение. Конвертор реализован на Java и требует для своей работы *Java Runtime Environment Standart Edition* версии не ниже 5.0. Как и система КуМир-2, конвертор является кроссплатформенным (Windows/Linux).

Конвертор использует для своей работы байткод для интерпретатора КуМир-2, созданный этой средой, который передается ему в виде XML-потока. Этот поток также может быть сохранен из КуМир'а в текстовый файл, а затем обработан конвертором, используя соответствующие аргументы командной строки.

Благодарности

Авторы благодарят А. Г. Кушниеренко и А. Г. Леонова за постановку задачи и внимание к работе, а также Д. В. Хачко за полезные обсуждения. Работа выполнялась в рамках сотрудничества с отделом учебной информатики НИИ Системных исследований РАН по теме «Разработка методических основ учебных курсов программирования и программного обеспечения учебного процесса».

Литература

[1] *Кушниренко, А. Г.; Лебедев, Г. В.; Сворень, Р. А.* Основы информатики и вычислительной техники. Изд-во: М.: Просвещение, 1993 г.

[2] *Кушниренко А. Г., Лебедев Г. В., Зайдельман Я. Н.* Информатика. 7-9 класс: Учебник для общеобразовательных учреждений. — 3-е изд., стереотип. - М.: Дрофа, 2002.

[3] *Кушниренко А. Г., Леонов А. Г., Ройтберг М. А., Хачко Д. В., Яковлев В. В., Карпов А. В., Субоч Н. М.* Система программирования КУМИР — интегрированная поддержка начальных курсов информатики и программирования. Настоящее издание.

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг, Д. В. Хачко,
В. В. Яковлев, А. В. Карпов, Н. М. Субоч

Москва, Пущино,
НИИ Системных исследований РАН,
Московский Государственный Университет,
Институт математических проблем биологии РАН,
Пущинский Государственный Университет

Проект: Кумир http://www.infomir.ru

Система программирования КУМИР: интегрированная поддержка начальных курсов информатики и программирования

Система программирования КуМир (Комплект Учебных МИРов) предназначена для поддержки начальных курсов информатики и программирования в средней и высшей школе, основанных на методике, разработанной во второй половине 1980-х годов под руководством академика А. П. Ершова. Эта методика, поддержанная массово изданными учебниками, широко использовалась в средних школах СССР, а затем и России, была поддержана программным обеспечением для всех типов школьных ЭВМ, а также нашла применение на механико-математическом факультете МГУ и ряде других вузов на мини-ЭВМ и персональных ЭВМ под управлением MS DOS. В массово изданных школьных учебниках и в системе КуМир-ДОС использовался придуманный А. П. Ершовым школьный алгоритмический язык — простой алголоподобный язык с русской лексикой и со встроенными командами управления программными исполнителями (Робот, Чертежник и др.).

Свободно распространяемая система программирования КуМир-2 — это новая реализация системы КуМир, совместимая с КуМир-ДОС по языку и методике, но адаптированная к реалиям работы на современных компьютерах и коммуникационных устройствах. КуМир-2 может использоваться под управлением Linux или MS Windows, что обеспечивается реализацией над библиотекой Qt. В настоящее время оба исполнения КуМир-2 используется в начальном прак-

тикуме по программированию на механико-математическом факультете МГУ.

Во время ввода или исправления программы компилятор КуМира постоянно обрабатывает вносимые человеком изменения и постоянно выдает на полях программы предупреждения о замеченных ошибках или несоответствиях. КуМир отслеживает все синтаксические ошибки, которые в принципе обнаружимы при редактировании: ошибки в записи выражений, попытки изменения значений аргументов процедуры, несоответствие параметров при вызове по числу и типу и т. д. В любой момент редактирования программы готов откомпилированный код, который может быть запущен на выполнение без малейшей задержки. Аналогично, при выполнении программы КуМир привязывает к исходному тексту и показывает Человеку все ошибки процесса исполнения: попытки использования переменных с неопределенным значением, выход индекса за границу массива, переполнения и т. д.

Отладчик КуМира в пошаговом режиме показывает на полях результаты присваиваний и проверок условий. Это позволяет новичку составлять и отлаживать простейшие программы, не пользуясь командами ввода–вывода или какими-либо возможностями отладчика.

Разработка системы КуМир-2 оказалась труднее разработки традиционных «пакетных» компиляторов и синтаксически ориентированных редакторов. Дело в том, что, согласно идеологии системы Кумир,

- С одной стороны, пользователь имеет право путем редактирования исходного текста создавать любые программы, в том числе и глубоко неправильные.

- С другой стороны, «непрерывно действующий компилятор» в каждый момент времени должен анализировать программу в полном объеме и сообщать обо всех допущенных в программе синтаксических непорядках.

  Это приводит к тому, что компилятор должен иметь дело не только с таким относительно простым объектом, как «правильная программа на языке Кумир», но и с гораздо более сложным объектом «неправильная программа на языке Кумир», в частности, система тестов компилятора должна проверять, что он «правильно обрабатывает» любые неправильные программы.

- Наконец, стоит заметить, что для каждого из этих объектов компилятор должен изготовить исполняемый код, так как по приказу пользователя Кумир пытается выполнять, пока это возможно, любую программу.

Постановка задачи на разработку системы КуМир-2 была выполнена А. Г. Кушниренко и А. Г. Леоновым. Разработка велась группой сотрудников ИМПБ РАН под руководством М. А. Ройтберга. Разработка архитектуры системы и реализация первой версии были выполнено Д. В. Хачко. Окончательная версия системы была создана Д. В. Хачко и В. В. Яковлевым. Отдельные модули были написаны при участии А. В. Карпова и В. И. Хачко. Тестирование системы было выполнено, в основном, Н. М. Субочем и А. В. Карповым.

Авторы благодарят Т. П. Кубышева, В. И. Хачко и Ю. А. Беляцкого за помощь при разработке системы КуМир-2, Я. Н. Зайдельмана за тестирование промежуточных версий системы и многочисленные плодотворные обсуждения, К. Ю. Богачева за помощь при установке системы КУМИР на механико-математическом факультете МГУ. М. А. Ройтберг благодарит А. Н. Табунова за высокопрофессиональные советы, которые способствовали успеху разработки.

Работа выполнялась в отделе учебной информатики НИИ Системных исследований РАН по теме «Разработка методических основ учебных курсов программирования и программного обеспечения учебного процесса».

Литература

[1] *Кушниренко, А. Г.; Лебедев, Г. В.; Сворень, Р. А.* Основы информатики и вычислительной техники. Изд-во: М.: Просвещение, 1993 г.

[2] *Кушниренко А. Г., Лебедев Г. В., Зайдельман Я. Н.* Информатика. 7-9 класс: Учебник для общеобразовательных учреждений. — 3-е изд., стереотип. - М.: Дрофа, 2002.

Д. А. Костюк, Д. А. Ильяшевич Брест,
Брестский государственный технический университет

Опыт внедрения свободного ПО в учебный процесс для специальностей информатики и радиоэлектроники

Аннотация

Рассмотрены практические особенности использования свободного программного обеспечения в учебном процессе ряда специальностей на примере белорусского технического вуза. Даётся обзор применяемого свободного ПО в виде средств обеспечения учебного процесса, в качестве вспомогательных программных средств, а также в качестве предмета практической части изучаемых дисциплин. Анализируется целесообразность и возможность сокращения доли коммерческого ПО, ближайшие перспективы свободных программных средств в учебном процессе с учётом текущей картины трудоустройства молодых специалистов.

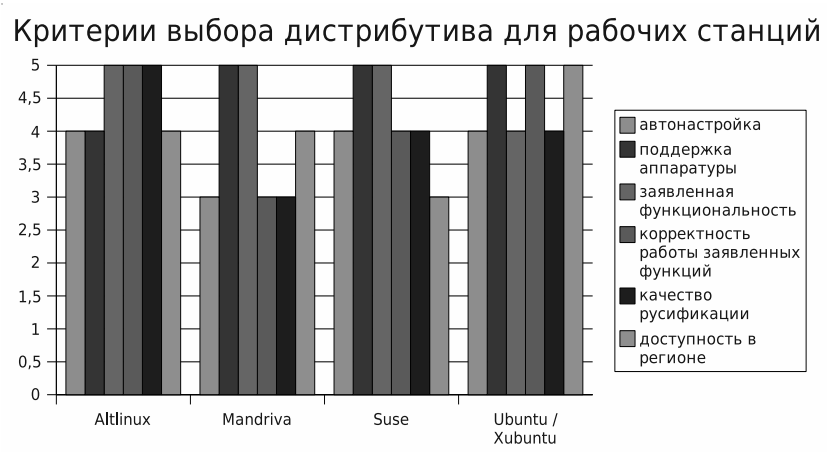
Области применения свободного ПО в БрГТУ можно разделить на три категории:

– средства обеспечения учебного процесса (системное ПО);

– ПО, преподаваемое в рамках учебных дисциплин;

– вспомогательное ПО, используемое, в частности, при подготовке и оформлении учебных работ.

Имеющийся в регионе спрос на специалистов, обусловленный сложившейся практикой и не очень внимательным отношением к лицензионному ПО, обосновывает необходимость изучения ОС Windows и ряда коммерческих программ. Свободное системное ПО на рабочих станциях ограничено ролью альтернативной ОС либо присутствует на терминальном сервере. Ранее использовались дистрибутивы Alt Linux (из-за универсальности, удобства управления приложениями, хорошей русификации). В 2007 г. лидерство сместилось в пользу Ubuntu (из-за широкой доступности и неплохого качества дистрибутива). Результат сравнения дистрибутивов применительно к имеющемся аппаратуре и задачам можно видеть на рисунке.



В 2006 г. выполнен экспериментальный перевод устаревших рабочих станций на роль тонких клиентов. Однако результат признан малоэффективным для университета, как не вполне отвечающий требованиям преподаваемых дисциплин, не решающий проблем лицензионной чистоты из-за политики лицензирования Microsoft, не соответствующий имеющемуся парку ЭВМ (доля наиболее подходящих рабочих станций в университете незначительна из-за того, что в годы их производства экономический кризис препятствовал закупке оборудования). Следует отметить случаи более удачного внедрения терминальных решений другими белорусскими вузами [1].

Серверное ПО выполняет функции межсетевых экранов, почтового и веб-сервера, предоставления персонализированного доступа к Интернет, файл-серверов и серверов аутентификации. Две последние роли в университете традиционно выполнялись ОС Novell Netware, а для организации файрволов использовались различные версии Debian. В связи с деактуализацией Netware проведён поиск альтернативы с учетом требуемой специфики (наличие большого числа мигрирующих пользовательских аккаунтов и рабочих станций с Windows). В результате сегменты сети двух кафедр переведены под управление контроллера домена на базе Samba 3.x, дополненного стандартным механизмом пользовательских квот.

Вспомогательное ПО, предлагаемое студентам, включает в себя стандартный набор OpenOffice.org + Firefox, а также с 2007 г. LaTeX

для вёрстки учебных работ (для последнего разработаны методические материалы и шаблоны, меры поощрения студентов). Широкому применению LaTeX мешает сложность начального освоения и отсутствие аудиторных часов; вместе с тем применение упрощает процедуру нормоконтроля.

В БрГТУ выполняется обучение по четырём специальностям группы информатики и радиоэлектроники, три из которых связаны с программированием и информационными технологиями (ИТ), а четвёртая ориентирована на промышленную электронику. На начало 2008 г. наименее задействовано свободное ПО последней категорией, отчасти из-за профориентации студентов. С текущего года введён практикум численных расчётов в среде SciLab на базе результатов, представленных в [2]. Разрабатывается ознакомительный практикум по САПР GNU Electric.

На ИТ-ориентированных специальностях преподаётся курс на базе [3], включающий лабораторные работы по утилитам и основам администрирования Linux. В 2007 г. в регионе отмечен сдвиг среди предлагаемых вакансий в сторону корпоративных работодателей, ориентированных на коллективную разработку ПО. Это смещает фокус от популярных ранее RAD-систем фирмы Borland в сторону средств разработки Microsoft или свободной платформы Eclipse. Внедрение последней в учебный процесс планируется, но отчасти затруднено её высокими аппаратными требованиями.

Преподаются курсы программирования на Java (с 1999 г.) и веб-программирования (с 2006 г.), условно относимые к свободным платформам. В 2007 г. в курсе системного программирования введён раздел по основам низкоуровневого программирования в Linux, в 2008 г. — практикум по средствам разработки и отладки программ для микроконтроллеров и спецЭВМ на основе инструментария GNU. Более широкое преподавание программирования под свободные платформы ограничивается недостатком учебных часов и преподавательского состава. В качестве дополнительной меры поощряется самостоятельное освоение студентами свободного ПО при выполнении курсового и дипломного проектирования, использование свободных технологий в НИРС, участие в конференциях.

Литература

[1] *Бойко В.К. и др.* Информационные ресурсы факультета математики и информатики ГрГУ и перспективы их развития / Современные информационные компьютерные технологии; сб-к научных статей. Гродно, 2006.

[2] *Журавлев В.А., Котяшев Е.О.* Scilab в учебном процессе в Белорусско-Российском университете и Могилевском государственном университете им. А.А. Кулешова // LVEE-2007: тезисы конференции. `http://www.lvee.org.by/index.php/2007_18`.

[3] *Танненбаум А.* Современные операционные системы. СПб.: Питер, 2006.

А. Н. Непейвода

Ижевск,
Удмуртский Государственный Университет

Автоматизированное построение сетевого графика проекта

Как известно, правильный выбор инструментария для решения задачи может сократить длину программы в несколько десятков раз[2]. За примерами далеко ходить не надо. Рассмотрим типичную задачу построения сетевого графика проекта. Её суть заключается в том, что есть несколько взаимосвязанных друг с другом заданий, каждое из которых требует определённого времени и не должно выполняться раньше, чем набор заданий-предшественников (которых может и не быть). Например, издательство, выпускающее книгу, не может делать переплёт, не закончив редактирования, тогда как дизайн обложки может продумываться параллельно вёрстке. Задача построения сетевого графика — это задача представления списка заданий в виде графа, на котором видны зависимости между заданиями и так называемый «критический путь» — набор работ, которые надо начинать немедленно, как только это станет возможно.

Далее мы будем использовать ряд терминов, принятый в теории графов:

Событие — момент окончания работы и/или начала новой. Два важных события — это начало и завершение проекта.

Реальная связь между событиями **A** и **B** — задание, которое необходимо выполнить, чтобы после события **A** наступило событие **B**.

Виртуальная связь между событиями **A** и **B** — это указание на то, что событие **B** не может наступить раньше события **A**, хотя они и не связаны заданием. Виртуальные связи возникают потому, что для начала некоторых работ требуется завершение более чем одного задания, которые, в свою очередь, являются предшественниками также более чем одной работы.

Правильно рассчитанный сетевой график должен содержать как можно меньше виртуальных связей.

К этой задаче можно подступиться с разных сторон. Структурное программирование отлично поддерживает вычисления, но требует весьма громоздких конструкций при работе со сложными логическими структурами. Однако существуют особые языки программирования, не столь «заточенные» под вычисления, которые ловко справляются со сложной логикой. Поскольку построение сетевого графика — задача больше логическая, чем вычислительная, имеет смысл использовать один из таких особых языков. Нами был выбран язык программирования Prolog[1], созданный на теоретической основе метода резолюций. Он оказался очень хорош для программирования логики графов, и программа расчёта графика на Prolog занимает всего чуть более трёхсот строк, тогда как оценки минимума длины такой программы на универсальном языке C/C++ дают больше полутора тысяч строк. Но на последнем этапе возникла трудность. Рассчитанный план было бы нагляднее всего представлять в виде графа, но Prolog очень слаб в области вв ода-вывода. Поэтому графический модуль для изображения уже рассчитанного графика пришлось писать на C с использованием библиотеки OpenGL. Однако расчёт координат событий и стрелок производился также в Prolog — в пределах тех же трёхсот строк. Сетевой график нельзя предполагать планарным, поэтому следовало так расположить его вершины, чтобы стрелки пересекались пореже. Итак, вся сложная логика уместилась в трёхстах строчах — многоязыковость в полной мере оправдала себя.

Данная работа позволяет использовать в практике преподавания сетевое планирование, не прибегая к проприетарному программному обеспечению типа MS Project.

Литература

[1] *И. Братко* Алгоритмы искусственного интеллекта на языке PROLOG, М.: «Вильямс», 2004

[2] *Н. Н. Непейвода, И. Н. Скопин* Основания программирования, Москва-Ижевск: Институт компьютерных исследований, 2003.

Е. Д. Патаракин Переславль-Залесский,
Институт Программных Систем

Проект: Летописи.ру http://letopisi.ru

Рост знаний в сообществе Летописи.ру

Аннотация

Представлен анализ развития сообщества Летописи.ру, отмечающего в феврале 2008 два года существования. Зарегистрировано более 13 тысяч участников, создано более 25 тысяч статей.

Введение

В феврале 2008 года проекту Летописи.ру исполняется 2 года. Статистика проекта на 15 января 2008 года:

- Всего в базе данных содержится 64 324 страниц;
- 25 447 страниц, которые считаются полноценными статьями;
- Зарегистрировались 13 500 участников, из которых 59 администраторов;
- Для классификации материалов участники создали 2019 категорий. Наибольшее число категорий связано со школами и регионами. Категория:Учитель — 784 объекта, Категория:Ученик — 727 объектов;
- Для автоматизации работы используется около 300 шаблонов.

Летописи использовались как площадка для продвижение новых информационных технологий. Одним из результатов проекта можно считать появление нескольких региональных Вики-площадок в России и странах СНГ (Украина, Казахстан). Эти площадки копировали идею использования МедиаВики как среды для организации учебной деятельности. Многие из них пошли дальше Летописи в плане дополнительных технических возможностей, специального регионального контента.

Как и когда происходит рост знаний? Как мы можем судить о том, что в этой среде, в этом сообществе происходит коллективное накопление знаний и участники развивают здесь новые умения?

- Кирпичики. В среде есть маленькие кирпичики, которые используются многократно;
- Пирожки. Участники делятся знаниями не только как кирпичиками, но и как пирожками с рецептом. Можно не просто использовать (съесть), но и узнать как он готовится и научиться выпекать пирожки самому;
- Кухня. Мы постоянно что-то готовим в этой среде. Это кухня, где совершаются постоянные рутинные операции. Здесь может происходить одомашнивание, присвоение знаний и умений;
- Кипящий бульон культуры.
- Место, где поговорить о своем коллективном мышлении.

Статьи МедиаВики как кирпичики

Летописи использовались как среда для создания, хранения и использования множества цифровых учебных объектов. Здесь наибольший интерес представляет повторное использование цифровых объектов — одно из важный умений XXI века, которым должны овладевать школьники, студенты и учителя.

По мере развития технологий в сферу построения значимых продуктов попадают всё новые маленькие кирпичики — цифровые учебные объекты, пригодные для повторного использования в образовательных целях.

Примером многократного использования материалов может служить Категория:Веб2007, статьи которой многократно воспроизводятся и вклеиваются внутрь лекций, семинаров и круглый столов на базе Летописи. Далее последует врезка статьи Программа тренинга по Веб2007, которая содержит множества примеров и отсылок на материалы Летописи. Обратите внимание на механизм врезки в текст другой статьи через механизм шаблонов. Благодаря механизму шаблонов нет никакой необходимости копировать программу из статьи в статью. Она превратилась в кирпичик, который можно любое число раз вкладывать в разные стенки.

Летопись как кухня

Внутри сложных современных информационных комплексов происходит совместная эволюция, в которую вовлечены люди и средства. Как только мы начинаем использовать новые средства, они изменяют привычные условия нашего существования. Подход, нацеленный на саморазвитие, заставляет нас постоянно перестраивать наш мир. Нам приходится осваивать новые роли, вырабатывать новые привычки и осваивать новые методы деятельности. Эти изменения в нас происходят только потому, что появляются новые технические средства, которые мы сами же и придумали. После того, как новые средства деятельности вносятся в сообщество, между средствами и людьми, которые их используют, происходит совместный процесс эволюционных изменений. Новые технические средства ничего не добавляют к среде обучения, они просто всю эту среду меняют. Алан Кей поясняет эту мысль следующей метафорой: после того, как в Австралию выпустили кроликов, ситуация вовсе не выглядела суммой = Австралия + Кролики. Кролики поменяли весь ландшафтный дизайн Австралии. Развитие сетевых технологий не добавляет технологии к привычной учебной практике. Это просто другой мир. И учебный класс, где у каждого ученика свой личный ноутбук, подключенный к сети Интернет, это новое и иное качество, а не тот же прежний класс, но с добавкой компьютеров. И мы сами, когда к нам добавили новые средства, стали другими. И это постоянно проживаемое, практикуемое изменение, готовность к изменению, способность к адаптации вместе с техническими средствами — есть одно из умений XXI века. Умение есть дело привычки. Умения формируются через деятельность внутри сообщества обмена знаниями. Одно из основных поло-

жений, на которых базируется теория сообщества практики, состоит в том, что знания и умения всегда осваиваются в определенном контексте. Теория сообщества практики тесно связана с конструктивизмом, согласно которому деятельность и действия являются основой психического развития. Сформированный Пейпертом на основе конструктивизма конструкционизм подчеркивает значимость конкретных объектов — программ, рисунков, фильмов и схем, которыми могут обмениваться авторы. Конструкционистские сетевые сообщества: ВикиВики, Скретч, Сквик, NetLogo — формируют пространства, где вокруг средства конструирования и конструктивной деятельности складываются сообщества авторов. Авторы пишут свои статьи, разрабатывают свои проекты, конструируют истории или интерактивные игры и обсуждают свою деятельность, представляют сообществу свои объекты. Везде мы видим интенсивный обмен знаниями. В Летописи можно не только играть с маленькими кирпичиками статей и шаблонов, но и пользоваться другими выразительными средствами.

Летописи как кипящий бульон

Большой интерес представляют механизмы решения проблем и конфликтов, постоянно возникающих у участников процесса совместного редактирования. Средства коллективного создания и редактирования создают энергетически насыщенную среду взаимодействия участников, который можно назвать «кипящим бульоном культуры». Нетерпимость к чужому мнению, ошибкам (фактическим, орфографическим и пунктуационным), стилю самовыражения и поведения, постепенно сменяется желанием помочь и улучшить начатое другими. Администраторы проекта на ночь в обязательном порядке перечитывают Джона Холта и учатся не протягивать ученикам свои умелые ручки помощи в пожарном порядке. Делать это очень трудно. Просто сидеть рядом и смотреть как новички заводят себе странные имена, дают невообразимые названия статьям и никогда не читают правила — всё это очень тяжело. В идеальном мире учителю надо было бы потратить целый вечер для того, чтобы вначале подготовить для ученика-участника хорошие шаблоны описания зверей и птиц, людей и деревень, полистать Википедию и подумать, какие статьи на вырост можно было бы сделать вместе. Но в реальности учителя информатики и прочих школьных предметов не тратят много времени на подготовку, по сути своей, абсолютно нового педагогического и

информационного события, а «с места в карьер» запускают татаро-монгольское иго детей в вики-среду и уходят куда-то по своим учительским делам...

Место, где поговорить о своем коллективном мышлении

Мы можем отслеживать изменения в своем коллективном сознании. Мы можем говорить, обсуждать, сравнивать и наблюдать структуры своего коллективного сознания. У нас есть технологические возможности для коллективной рефлексии. Используя записи и связи, существующие внутри базы данных коллективного гипертекста, мы можем анализировать вклад отдельных участников, динамику роста страниц, вклад отдельных категорий и т. д. Самое главное — мы можем вовлекать студентов в реальную сетевую исследовательскую деятельность.

Заключение

Всегда важно знать, какими средствами люди пользовались, готовя текст статьи или презентации. В этом плане презентация о бедах российской культуры с жалобами на засилье западных стандартов, выполненная со стандартными картиночками Паблишера, несёт дополнительное скрытое сообщение: «Так нет же больше никаких средств и стандартов». Данный текст готовился, редактировался, обсуждался внутри МедиаВики проекта Летописи.ру[1].

Из маленьких кирпичиков мы построили себе кухню, напекли пирожков, накипятили бульону. Сидим, разговариваем о хорошем, растим свои знания, принимаем друзей.

Литература

[1] http://letopisi.ru/index.php/Рост_знаний_в_сообществе_Летописи

Б. Б. Ярмахов Нижний Новгород,
Нижегородский государственный университет (ФСН)

Проект: OLPC http://laptop.org

OLPC как модель массового внедрения свободного ПО в образование

Аннотация

Созданный учеными Массачуссетского Технологического Института проект «Ноутбук каждому ребёнку» (OLPC) построен на принципах конструкционизма и предполагает внедрение в практику образования модели «1 учащийся : 1 компьютер». Основным инструментом проекта является ноутбук XO — ультрапортативный компьютер с операционной системой Sugar (разработанной на основе Red Hat Linux) и предустановленным программным обеспечением, ориентированном на самостоятельную творческую работу учащегося в сотрудничестве с учителями и другими учащимися. В настоящее время в России ведутся работы по локализации программного обеспечения ноутбука, складывается сообщество заинтересованных в проекте преподавателей и ИТ-специалистов.

Массачусетский Технологический Институт (MIT) сегодня является крупнейшим университетским центром, реализующим научные разработки в самых различных областях промышленности и в социальной сфере. При университете существует ряд структур, занимающихся внедрением наукоёмких технологий и разработкой бизнес-проектов, ориентированных на получение прибыли (Центр технологических инноваций, форум предпринимателей MIT и т. д.). Объединённый оборот всех компаний, основанных выпускниками и преподавателями MIT, позволяет готовить об институте как о 24-й по величине в мире экономической системе, в которой работает 1,1 миллиона человек, и ежегодный объём продаж которой по всему миру составляет 232 миллиарда долларов.

Вместе с этим, MIT является активным инициатором некоммерческих проектов, ориентированных на масштабные социально-культурные изменения. Одним из наиболее активных инициаторов такого рода проектов является Лаборатория Медиа Исследований (MediaLab). Лаборатория, основанная в 1980 году, стала крупнейшим

центром конструкционизма — образовательной методологии, предполагающей освоение знания ребенком не в результате заучивания учебного материала наизусть, а в процессе его воссоздания.

С 2005 года учеными Media Lab MIT реализуется проект One Laptop Per Child (OLPC), ключевым компонентом которого является разработка и создание портативного компьютера-ноутбука для учащихся начальных классов развивающихся стран. Инициатива по созданию и внедрению в образовательную практику недорогих, но, в то же время, функциональных портативных компьютеров — ноутбуков — была представлена мировому сообществу на Мировом саммите по созданию информационного общества, который прошел в ноябре 2005 года в Тунисе. Говоря о важности этого начинания, генеральный секретарь ООН Кофи Аннан подчеркнул: «попадая в руки детей, эти надёжные и многофункциональные компьютеры помогут им быть более вовлечёнными в учебный процесс. Благодаря им дети смогут осваивать учебный материал не зубрёжкой, а через практику. Изменится и само образование — с их помощью дети смогут учиться друг у друга».

Ноутбук XO представляет собой ультропортативное устройство (диагональ экрана 7 дюймов), обладающее рядом преимуществ по сравнению с обычными ноутбуками. Он выполнен в прочном корпусе, водоустойчив, его экран работает в двух режимах — при комнатном и солнечном освещении. Кроме этого, он может быть легко трансформирован в планшетную конструкцию, что позволяет использовать его в качестве устройства для чтения книг.

Ноутбук обладает повышенной коннективностью (за счет использования ячеистой сети). Это позволяет организовывать работу школьников в различных режимах: как в индивидуальном, так и в парном и в групповом.

Первоначальная комплектация ноутбука предполагала оснащение его мускульным генератором. Несмотря на то, что настоящее время мускульный генератор не входит в базовую комплектацию модели, он может быть поставлен в виде дополнительного устройства. Кроме того, ведутся разработки по обеспечению питания ноутбука с помощью альтернативных источников энергии (например, тягловой силы).

Проект OLPC построен на принципах свободно-распространяемого программного обеспечения. Создатели ноутбука утверждают, что их приверженность свободному программному обеспечению носит «не

религиозный, но прагматический» характер, поскольку оно более соответствует образовательным потребностям.

В качестве операционной системы ноутбука используется вариант Red Hat Linux — Sugar. При создании интерфейса ноутбука его создатели осознанно отходили от офисного набора метафор, характерных для «больших» коммерческих компьютеров. Понятия «рабочий стол», «приложение», «папка», «директория», привычные пользователям Windows сменили «занятие», «дневник», «мой дом» и т. д. В комплект поставки входит ряд специализированных программ («занятий»), разработанных специально для использования их в начальной школе: Etoys — визуальное средство, созданное на базе языка LOGO, TamTam — детский мультимедийный редактор, Measure — программа для обработки данных, получаемых с помощью сенсорных датчиков, текстовый редактор Write, браузер и ряд других программ, необходимых для успешной работы школьника в современных информационных средах.

В своей работе OLPC руководствуется пятью основными принципами:

1. Ноутбук должен становиться собственностью школьника, который использует его в обучении;
2. Ноутбук ориентирован на школьников младшего возраста;
3. Ноутбук является инструментом развития всего сообщества, в котором живет ребенок;
4. Ноутбук должен обеспечить школьника максимально возможной связью;
5. Он ориентирован на использование бесплатного ПО с открытым кодом (FOSS).

Предлагая предоставлять ноутбук XO в собственное пользование детям, создатели OLPC предлагают сделать следующий, по сравнению с традиционной концепцией компьютерных классов, шаг. Они считают, что для школьника младшего возраста этот ноутбук станет привычным инструментом для игры и обучения. Делая его инструментом освоения культуры и общества, они, таким образом, будут способствовать позитивным изменениям в своем сообществе и своей стране.

Пилотное внедрение ноутбука OLPC осуществляется в ряде стран Африки, Азии и Латинской Америки. С осени 2007 года началось

массовое внедрение ноутбука в школах Уругвая. Мониторинг проекта показывает значительное повышения мотивации обучения у школьников, которые используют ноутбук.

В настоящее время силами волонтерского сообщества ведутся работы по локализации программного обеспечения для ноутбука на русский язык. Ноутбук XO может стать ключевым средством для предотвращения «цифрового неравенства», особенно в удалённых и плохо обеспеченных компьютерной техникой регионах страны.

А. А. Панюкова
Москва,
ЮУрГУ / МГТУ им. Н.Э. Баумана

Проект: Обучающий курс для детей на базе Linux
http://heap.altlinux.ru/engine/Mex3andMe/Summer10DayCourse

Создание обучающего курса для детей на базе Linux

Аннотация

На базе ЮУрГУ второй год проводится летний обучающий курс для детей, отдыхающих в оздоровительном лагере. Курс посвящен работе с графикой, анимацией и видео, используется свободное программное обеспечение (ОС ALT Linux). По итогам работы в этом году созданы Live CD и инсталлятор, содержащий всё необходимое ПО, методические указания и программу курса.

В ряде дисциплин, читаемых студентам (например, «Пакеты прикладных программ», «Компьютерные издательские системы» и т. п.), присутствуют разделы, связанные с созданием рисунков, коллажей, обработкой фотографий, анимацией, созданием потокового видео (клипов). Несмотря на очевидную привлекательность этих разделов, некоторые трудности организации обучения имеют место. Например, преподавателю приходится приспосабливаться к значительным различиям начального уровня подготовки студентов, различиям по знаниям, интеллекту, творческим способностям, умению самостоятельно работать и т. д.

Данная работа показала, что для демонстрации основных идей обработки рисунков и фотографий, построения клипов можно достаточно успешно использовать свободно распространяемое программное обеспечение (ПО). В данном случае не только решается проблема

юридических рисков, связанных с использованием нелицензионного ПО, и прочими сопутствующими проблемами, но и открывается дорога к решению ряда этических проблем, а именно воспитанию уважения к авторскому праву у массового пользователя.

Несмотря на достаточно распространённое мнение, что учить студентов системе Linux и его приложениям не стоит (т. к. они им не пользуются в домашних условиях как при выполнении заданий на самостоятельную работу, так и на пути реализации личных интересов), что Linux сильно специализирован и сложен и т. п., использование свободно распространяемого ПО дало возможность преподавателям удобно и корректно настроить систему с точки зрения пользователя и достигнуть желаемого результата.

Была организована и в рабочем порядке скорректирована система (среда для работы) с чёткой организацией отдельных её элементов (например, построена система организации хранения данных и общего доступа к ним, сформирован набор исходных материалов (видео, аудио) и заранее продумано и регламентировано его содержание). Хранение созданных файлов и общего доступа к ним — всё организовано таким образом, чтобы необходимые файлы располагались у всех участников одинаково и чтобы создавалось впечатление, что все файлы пользователь берёт со своего компьютера, чтобы не объяснять архитектуру сети.

Как показала практика, работа в этой системе вполне реальна, и достигнуты некоторые положительные результаты. Она дала возможность снизить проявление некоторых негативных моментов, которые изначально являются организационными особенностями проводимых занятий. Например, возможность задать вопрос преподавателю или послать сообщение с компьютера на другой компьютер (с помощью jabber) сделало более корректным общение на занятиях. По отношению к пользователю система достаточно хорошо проработана, и дальнейшие изменения целесообразно делать только за счёт решения каких-либо более принципиальных вопросов, связанных с простотой администрирования, простотой установки системы преподавателем, с вопросами организации сервера и т. п.

Проведён достаточно большой объем работы по подбору наиболее подходящих программных продуктов. Среди большого многообразия имеющихся продуктов предпочтение было отдано программе Kdenlive в качестве инструмента для создания клипов. В нём есть переходы и эффекты. Анимацию и растровую графику было решено

давать в рамках GIMP (создание gif-анимации), векторную графику — в XaraLX.

Построенная система явилась результатом обобщения опыта выездного обучения детей младшего и среднего школьного возраста в компьютерном классе на территории базы отдыха «Наука» и ДОЛ «Берёзка» ЮУрГУ. В течение двух сезонов проведено более 160 занятий (в виде мастер-классов) для очень «неудобной» аудитории, особенности которой включали:

- «разношерстность» аудитории, как по возрасту, так и по всем другим возможным признакам (по начальному уровню подготовки, по знаниям, по интеллекту и т. п.), вплоть до цели посещения;
- свободный график присутствия на занятиях;
- нежелание обучаться серьёзно (особенность времени года и места проведения занятий);
- неконтактность, нежелание работать в паре с кем-либо;
- слабую начальную подготовку: наличие некоторой части детей, не умеющих читать (чаще — отсутствие привычки читать), незнание английского языка, неумение набирать текст на клавиатуре, компьютеробоязнь (отсутствие навыков управления мышкой);

В той или иной мере эти особенности есть и у студентов, однако в данном случае аудитория дополнительно имела ещё достаточно много особенностей. Тем не менее преподаватели стремились обеспечить в меру высокое качество обучения и сделать всё возможное для того, чтобы участникам обучения захотелось не только прийти на следующее занятие, но и самостоятельно продолжить обучение по возвращении домой.

Опыт проведения занятий реализован авторами в решении трех задач: разработаны курс обучения (программа и методика), Installer (установочная система для конкретного курса) и LiveCD (компакт-диск с системой, который дети смогут увезти домой в качестве подарка). Подобран список дополнительных приложений, которые использовались для заполнения небольших пауз в работе (KStars, KGeography, KTuberling, Psi).

М. В. Быков Москва, РГГУ

Tagged Thesaurus древнегреческого языка

Мы привыкли к словарям табличной формы. Слева слово, справа много значений, может быть, несколько десятков. Это неудобно. Вспомните, дочитали ли вы хоть раз в жизни статью на слово «put» в словаре Мюллера? Отсюда и наше знание языков.

Мы, однако, живём в третьем тысячелетии. Должно быть так: я указываю (курсором) слово, и программа:

1. анализирует контекст и выбирает нужное мне единственное значение;
2. совершает (простейший) грамматический анализ примера;
3. находит подобные примеры, из которых она это значение и вывела.

Потому что, вдобавок, дело не просто в неудобстве. Дело гораздо серьёзнее — словари с табличной форме вообще не имеют права на существование. Это костыль. Слова вообще не имеют фиксированного и «утвержденного» значения. И произнесение, и понимание слова — поступок, т. е. действие, каждый раз уникальное. Словари в табличной форме умрут, и умрут скоро.

Для создания современного толкового словаря нужно составить тезаурус *всех* предложений данного языка, с указанием источника. Для двуязычного словаря — тезаурус *всех* имевших место быть переводов, с указанием источника. И — систему определения сходства и сравнения нашего, исследуемого контекста и «образцовых» контекстов. В первом приближении можно использовать просто грамматическое сходство предложений.

Ясно, что подобная задача по плечу лишь гигантам типа Google. И ясно также, что она легко осуществима. В качестве иллюстрации я предлагаю вам небольшой тезаурус древнегреческого языка, каждое предложение которого размечено ярлыками грамматических правил, по английски — «tagged thesaurus». Примеры взяты из нескольких ещё советских учебников и российских дореволюционных словарей и открытых источников в Сети.

Каждое предложение на древнегреческом может иметь несколько переводов, множество грамматических и синтаксических тегов и единственное указание на источник, напр. Матф.1-23 или Платон 217a. Программа снабжена полнотекстовым поиском по предложениям, переводам (на русском и английском) и тегам. Таким образом можно быстро создать рабочую выборку примеров на определённую тему, например «сослагательное наклонение» или «абсолютный причастный оборот».

Для учебных целей выборку можно просматривать в случайном порядке, пытаясь самостоятельно определить подходящие грамматические правила для каждого предложения и перевести его. Знакомые предложения можно из выборки удалять. Постепенно выборка уменьшается, а знания учащегося, соответственно, увеличиваются. Программа работает по принципу вики, то есть зарегистрированный пользователь может добавлять предложения, переводы и грамматические теги.

Программа написана на Rubi-on-Rails 2.0. В качестве полнотекстового индексатора использовался плагин Ferret. И программа, и содержание тезауруса являются свободным программным обеспечением.

Д. А. Варенов

Москва,
ФГУ ГНИИ ИТТ «Информика»

Проект: OpenPower http://openpower.itbu.ru

Разработка системы на базе архитектуры POWER по поддержке программистов и проектов на основе открытых исходных текстов в рамках проекта OpenPower

Проект OpenPower — это группа единомышленников, объединённых общей целью — продвижением ОС Linux.

Реализуемая в рамках этого проекта система представляет собой платформу для портирования существующих приложений на ОС Linux, а также разработки и совершенствования программ, созданных на базе открытых исходных текстов. Основной функцией данной системы является предоставление программистам возможности опробовать ОС Linux на серверах POWER путем открытия для них «не

администраторского» (non-root) SSH доступа к серверу IBM System P5 расположенного по договоренности с компанией IBM на базе Российского Государственного Университета Инновационных Технологий и Предпринимательства (РГУИТП). Проект OpenPower является одной из составляющих программы «Linux On Power», проводимой IBM по всему миру. На данный момент несколько передовых университетов уже имеют подобные серверы. Таким образом, у пользователей появляется возможность выбрать и использовать систему, исходя из разницы в их реализации и географического признака. И, тем не менее, данная система уникальна, поскольку это единственный проект такого рода, изначально ориентированный на русскоязычное сообщество пользователей где бы они не находились.

Удаленный доступ будет открыт для любого разработчика, соблюдающего правила пользования системой, а также придерживающегося одной из лицензий, одобренных OSI (Open Source Initiative). То есть лицензий, в рамках которых любой код должен быть доступен для бесплатного распространения.

Основные задачи

1. Практическое доказательство того, что ОС Linux на архитектуре Power жизнеспособна, является совместимой и стабильной платформой для работы с критически-важными приложениями в реальном 64-х битном окружении.
2. Предоставление ресурсов сервера обширному сообществу, а также опробация самого передового оборудования, разработанного для ОС Linux.
3. Выявление новых сфер применения и маркетингового продвижения для дальнейшего более широкого распространения Linux-серверов на рынке.
4. Обеспечение свободного доступа к системе для конечных пользователей и фирм — возможных покупателей данной системы в будущем.

Ожидаемые результаты от внедрения системы

1. При выполнении вышеизложенных задач появляется возможность организации сообщества пользователей системы, способ-

ствующего развитию, совершенствованию и популяризации ОС Linux и архитектуры Power в странах СНГ.
2. Создается реально действующая общедоступная демонстрационная система, которая обеспечивает собственный маркетинг на рынке за счет своей надежности и отказаустойчивости, позволяющая конечным пользователям самим оценить все достоинства архитектуры и ОС.
3. Внедрение данной системы облегчает и способствует процессу портирования приложений на ОС Linux.
4. Пуск в эксплуатацию данной системы дает возможность производителю оборудования (компании IBM) безвозмездно получить полную статистическую информацию, проверенные данные и характеристики работы системы.

М. В. Пономарева

Санкт-Петербург, СПбГУ

Проект: Электронная антология русской литературы XVIII века http://antology18.iling.spb.ru

Электронная антология русской литературы XVIII века: пример применения СПО в филологических исследованиях

Аннотация

Доклад посвящен применению СПО в филологических исследованиях. На примере создания "Электронной антологии русской литературы XVIII века" демонстрируется, как использование СПО и открытых стандартов позволяет создать: а) модель комплексного филологического анализа литературных текстов, б) систему представления текстов художественной литературы, в) инструмент для учебно-методической и научно-исследовательской работы, а также обеспечить верифицируемость результатов исследований.

1. Расширение области применения СПО — литературоведение.
2. Одна из актуальных задач современного литературоведения — многоаспектный анализ художественного текста, совмещающий

описание формальных характеристик текста (таких, как дата написания, время и место публикаций, жанровая принадлежность, посвящение и проч.) и содержательных его составляющих (напр., тема, топика, мотивы и т. д.).

3. На факультете филологии и искусств СПбГУ с сентября 2007 г. в рамках семинара "Русский XVIII век" разрабатывается проект "Электронная антология русской литературы XVIII века" (ЭА) (http://antology18.iling.spb.ru). Участники проекта: преподаватели, сотрудники и студенты СПбГУ, научные сотрудники ИЛИ РАН и ИРЛИ РАН.
4. Задачи проекта:

   - выработать модель комплексного филологического анализа литературных текстов;
   - создать модель представления текстов художественной литературы, позволяющую рассматривать материал с различных точек зрения, сократив при этом элемент субъективной интерпретации;
   - создать инструмент для учебно-методической и научно-исследовательской работы.

5. Подготовка текстов в электронном формате осуществляется в соответствии с рекомендациями TEI (*Text Encoding Initiative*, `http://www.tei{}-c.org`), а их обработка основывается на программе OpenJade (`http://openjade.sourceforge.net`). Использование СПО и открытых стандартов позволяет обеспечить верифицируемость результатов исследований. Это особенно важно для гуманитарных наук, которые зачастую отличаются некритическим отношением к данным, полученным посредством компьютерного моделирования.
6. Что позволяет СПО:

   - предложить принципиально новый подход к работе с художественными текстами, новые возможности для научных исследований;
   - описывать каждый текст, исходя из ряда установленных параметров. Результат: литературное наследие определенного исторического периода или автора представляется единым текстом, в котором каждое произведение потенциально свя-

зано по многим параметрам с целым рядом других произведений;

- задавать самые разные параметры описания текстов;
- предоставить возможность многовариантного анализа литературного процесса. Пользователь, обращающийся к ЭА, будет иметь возможность согласно заданным параметрам (одному или нескольким) выбирать определенную группу текстов для изучения. Таким образом, весь корпус текстов можно будет трансформировать в соответствии с теми или иными параметрами, в результате чего литературная жизнь XVIII в. окажется представленной с разных точек зрения.
- не ограничивать набор параметров, при помощи которых описывается текст.
  Особенность ЭА, состоит в том, что набор параметров, описывающих текст, потенциально неограничен. Описав текст один раз, впоследствии можно усложнять это описание, добавляя к характеристике текста какие-либо новые параметры, которые необходимы для продолжения исследования. Это открывает возможность самостоятельной научно-исследовательской работы. Применение открытой кодировки позволяет сделать систему настраиваемой на те или иные специфические задачи. Иными словами, добавляя новые параметры (теги), исследователь может решать свои индивидуальные задачи.

Вывод: подобная модель представления текстов открывает новые возможности работы с текстом, которая является основой любого литературоведческого исследования и позволяет проводить комплексное описание и изучение как творчества отдельных писателей и эпох, так и литературы в целом.

7. Перспективы: В перспективе предлагаемый подход может привести к созданию автоматизированного рабочего места литературоведа, то есть инструмента, облегчающего литературоведу решение его повседневных научных задач.

И. А. Хахаев

Санкт-Петербург,
ГОУ ВПО СПбТЭИ

«Лёгкие» пакеты научной графики

Аннотация

Рассматривается несколько свободно распространяемых пакетов научной графики, обеспечивающих быстрое построение двумерных и трёхмерных диаграмм по имеющимся данным, их обработку и анализ. Такие задачи являются типичными как при выполнении лабораторных работ, так и при проведении научных исследований.

Имеется множество свободно распространяемых пакетов для обработки и визуализации данных. На `sourceforge.net` в разделе «Scientific/Engineering» одних только проектов визуализации данных насчитывается более 2000. Однако такие мощные и известные пакеты, как *PAW++*, *Root*, *Scilab*, *Maxima*, *OpenDX* нерационально использовать для простых задач типа построения графиков по имеющимся данным, особенно если этих данных не так уж много. Поэтому предметом рассмотрения будут достаточно специализированные и простые в использовании пакеты, ориентированные именно на построение различных видов графиков по известным точкам или по формулам.

Ещё одна часто встречающаяся в лабораторных работах и исследованиях задача — оцифровка графиков и диаграмм, полученных на бумажном носителе. Для этих целей могут использоваться пакеты *g3data* (`http://www.frantz.fi/software/g3data.php`), *imview* (`http://experimental.act.cmis.csiro.au/imview/`) или *nip2* (`http://www.vips.ecs.soton.ac.uk/index.php`), причём наиболее соответствующим задаче является пакет *g3data*.

Для быстрого получения высококачественных двумерных графиков можно использовать электронные таблицы *Gnumeric* (`http://www.gnome.org/projects/gnumeric/`) и пакет *Grace* (*xmgrace*) (`http://plasma-gate.weizmann.ac.il/Grace/`). *Gnumeric* позволяет строить на одном графике независимые пары XY, а также получать несколько вариантов регрессии для одного набора данных. На русском языке имеется краткое описание особенностей построения диаграмм в *Gnumeric* (`ftp://ice.spb.ru/pub/articles/gnumeric_guide_listscharts.odt`). *Grace,* в свою очередь, позволяет проводить

достаточно сложную обработку данных, включая извлечение признаков и нелинейную подгонку. Есть также возможность использовать различные шкалы и масштабы для первой и второй осей X и Y, возможность увеличения произвольной области графика и произвольного аннотирования графика. Данные могут импортироваться из текстового файла или из канала (pipe). Пакет может работать в фоновом режиме (без GUI), забирая данные из канала и автоматически создавая выходные графические файлы. Существует русский перевод учебника по *Grace* (`ftp://ice.spb.ru/pub/articles/grace_Tutorial.html.zip`), а также небольшой пример построения регрессии (`ftp://ice.spb.ru/pub/articles/grace_simple_regression.odt`).

В качестве приложений для работы с двумерными и трёхмерными графиками можно рассматривать пакеты *LabPlot* и *qtiplot*. *LabPlot* позволяет строить графики по данным из текстового файла и графики 2D- и 3D- функций. Среди возможностей анализа — сглаживание, численное дифференцирование и интегрирование, поиск экстремумов, выделение сезонных компонент, полиномиальная регрессия. Имеются возможности произвольного аннотирования графиков. Графики могут экспортироваться в PostScript, PDF, SVG и большое количество растровых форматов. Сайт проекта (`http://labplot.sourceforge.net/`) оформлен в виде Wiki, документации на русском языке не существует. *Qtiplot* похож на LabPlot, однако имеет несколько другие возможности. Возможностей аннотирования графиков несколько меньше, а возможностей анализа — несколько больше. Интересны возможности по приближению кривых гауссовыми и лоренцевыми функциями. Также имеются интересные возможности нелинейной подгонки. Графики могут экспортироваться в EPS, SVG и растровые форматы. На сайте проекта (`http://soft.proindependent.com/qtiplot.html`) и в составе бинарных пакетов документация на русском языке отсуствует. Пакет есть в репозитарии ALTLinux, но в имеющейся там версии есть некоторые проблемы локализации.

Каждый из описанных выше пакетов имеет присущие только ему функции, так что полной взимозаменяемости не наблюдается. Однако есть возможность выбора пакета, исходя из конкретного круга задач и характера отображаемых и анализируемых данных. Поэтому текущей задачей, по-видимому, становится перевод документации и создание информационной поддержки этих пакетов на русском языке.

Е. А. Чичкарев Мариуполь, Украина,
Приазовский Государственный Технический Университет

Использование Linux и open-source программного обеспечения в преподавании математических дисциплин и объектно-ориентированного моделирования

Аннотация

Проведено опробование ряда свободных программных средств для организации практических и лабораторных занятий по различным курсам. Установлено, что пакет Maxima обладает достаточной функциональностью и легко осваивается студентами при изучении курсов математического и функционального анализа, численных методов и т. п. Для преподавания курсов, связанных с объектно-ориентированным моделированием, опробованы Umbrello, ArgoUML, StarUML. Для студентов наиболее удобным оказался StarUML.

Современная тенденция к сокращению объёма аудиторных занятий в преподавании большинства дисциплин (в т. ч. математических — математического и функционального анализа, численных методов и т. п.) и акцент на развитие самостоятельной работы студентов требует существенных изменений методики преподавания, направленных на улучшение содержательности и наглядности выполняемых студентами заданий, сосредоточение их внимания на существе изучаемого материала.

С другой стороны, по мнению Э. Дейкстры, «в стандартном математическом курсе студент часто видит проблему настолько «маленькой», что он имеет дело с только одним семантическим уровнем. В результате многие студенты видят в математике скорее искусство организации символов на листе бумаги, чем искусство организации своих мыслей».

Эффективным инструментом обучения математике (при условии надлежащей постановки заданий) являются системы компьютерной математики (Maple, Maxima и т. п.). В частности, в условиях организации учебного процесса на кафедре информатики ПГТУ СКМ “Maxima” используется для выполнения домашних заданий по курсам «Математический анализ», «Функциональный анализ», «Уравнения

математической физики». Входной язык систем Maxima хорошо усваивается студентами и достаточно приспособлен для решения учебных и учебно-практических задач не только математического, но и алгоритмического содержания.

Использование систем компьютерной математики естественно сочетается с традиционными методами решения математических задач при условии специальной подготовки или адаптации заданий, включающих элементы исследования.

Внедрение в учебный процесс компьютерных классов, укомплектованных компьютерами с Linux, ставит вопрос об организации учебного процесса по различным дисциплинам на базе свободного ПО.

Хорошо известным решением является использование пакетов Scilab или Octave, либо Python+SciPy для решения задач моделирования технических или экономических систем. Однако обучение IT-специалистов требует решения задач не столько численного, сколько объектно-ориентированного моделирования и разработки прототипов программных систем.

Общеизвестно использование Rational Rose или Sybase PowerDesigner для визуального моделированиея с использованием UML.

Однако достаточно широкие возможности для решения большинства задач предоставляют и полностью открытые системы. В частности, для организации лабораторного практикума на кафедре информатики ПГТУ опробованы:

- в компьютерном классе с установленным Linux — Umbrello UML Modeller и dia;
- в компьютерном классе с установленным Windows — StarUML и ArgoUML.

Установлено, что все опробованные пакеты достаточно функциональны и могут служить базой как для проведения практикума, так и курсового и дипломного проектирования, хотя каждому из них присущ и целый ряд специфических недостатков. Для студентов наиболее удобным оказался StarUML.

В настоящее время разработывается комплект документации для организации учебного процесса на базе свободногое ПО (методические указания по различным дисциплинам, руководство по лабораторному практикуму и т. п.).

Е. Р. Алексеев, Е. Рудченко, О. Шамота Донецк,
Донецкий национальный технический университет

Проект: www.open-educationsoft.land.ru

Использование свободно распространяемого программного обеспечения при изучении курса информатики

Аннотация

В работе предлагается программа общеобразовательного курса по информатике для студентов инженерных и экономических специальностей на базе свободного программного обеспечения.

Курс информатики включает в себя несколько традиционных разделов: *устройство персонального компьютера*, *операционная система*, *офисные пакеты*, *алгоритмизация и программирование*, *использование математических пакетов для решения инженерных и экономических задач*, *основы работы в Интернет*.

Традиционно этот курс в ВУЗах Украины преподаётся на основе проприетарного программного обеспечения. Это связано, в первую очередь, со стандартами образования. Мы хотим предложить альтернативную концепцию преподавания курса информатики.

Университетский курс информатики должен показать тенденции в развитии современного программного обеспечения, одной из которых является развитие свободно распространяемого ПО. Курс информатики позволяет сформировать у будущих специалистов понимание того, что операционной системой MS Windows современные IT-технологии не ограничиваются.

Раздел «*Операционные системы*» может включать в себя особенности файловых систем ОС Windows и Linux. Далее имеет смысл рассмотреть процесс установки операционной системы Linux, чтобы убедить студента, что установка современной ОС семейства Linux (например, Alt Linux, Ubuntu) не сложнее, чем установка Windows. Следует обратить внимание студентов на то, что после установки операционной системы пользователь получает большое количество установленных программ и доступ к репозиторию программ.

В курсе информатики рационально использовать кроссплатформенное свободно распространяемое программное обеспечение, что

позволит в бюджетной организации использовать легальное ПО и обучать студентов без привязки к конкретной операционной системе.

Офисное программное обеспечение рационально изучать на примере OpenOffice. Следует обратить внимание на возможность использования электронных таблиц OpenOffice Calc при решении инженерных задач и обработке табличных данных. В качестве альтернативы OpenOffice Calc можно рассмотреть электронные таблицы Gnumeric.

Традиционно для специальностей экономического направления раздел, посвящённый *алгоритмизации и программированию*, излагается на основе языка Basic, студенты инженерных специальностей изучают программирование на основе Pascal или C++. В качестве среды для изучения программирования на Basic можно предложить OpenOffice Calc, Gambas (для Linux). При изучении программирования на базе Pascal можно выбирать между Gnu Pascal, Free Pascal и Lazarus. Программирование на C++ можно изучать, используя компилятор gcc. При изучении программирования под управлением ОС Linux в качестве среды программирования можно предложить использовать Geany.

В качестве *математических пакетов*, предназначенных для решения инженерных задач, можно предложить математические пакеты Scilab, FreeMat, Maxima и ряд других. Особое внимание следует обратить на пакет Scilab, так как в его состав входит система моделирования scicos.

Кроме того, среди свободно распространяемых инженерных и математических программ есть множество специализированных программ (*freefem* — пакет для решения уравнений в частных производных методом конечных элементов, gEDA — многофункциональный пакет для инженеров-электронщиков, позволяющий конструировать электронные схемы, qCAD — простая 2-мерная САПР с открытым кодом), предназначенных не столько для учебных целей, сколько для решения реальных задач.

В курсе информатики изучение *интернет-технологий* ограничивается web-серфингом, электронной почтой, средствами обмена мгновенными сообщениями и основами языка html. Для этого можно использовать программы Amaya, Mozilla Firefox, Mozilla Thunderbird, Mozilla Seamonkey, Nvu, KompoZer, Pidgin.

На сайте `www.open-educationsoft.land.ru` размещены методические разработки авторов, посвящённые некоторым разделам курса информатики на базе свободного программного обеспечения.

Кроме того, авторами был собран DVD-диск со свободными программами для Windows, необходимыми для изучения курса информатики на базе свободно распространяемого программного обеспечения. На этом же диске находится документация по представленным пакетам.

Что же мешает переходу на преподавание информатики на базе свободного ПО? По мнению авторов, основных причин несколько:

- стандарты Министерства Образования (на Украине);
- на курсе информатики базируется большое количество других специальных предметов, переход которых на свободное программное обеспечение может занять достаточно много времени;
- практически полное отсутствие необходимой методической литературы; в последние годы количество книг о свободном программном обеспечении и о Linux увеличилось в несколько раз, но это по-прежнему, в основном, литература, ориентированная на профессионалов; не хватает методической литературы, практически нет литературы, ориентированной на пользователя, начинающего осваивать компьютер, нет книг по прикладному программному обеспечению.

Если всерьёз говорить о внедрении свободного программного обеспечения в среднюю школу и ВУЗ, то, кроме разработки и внедрения программного обеспечения, необходимо решать проблему методического обеспечения.

К. А. Маслинский Петербург, ALT Linux
Проект: ALT Linux Team http://lists.altlinux.org

Списки рассылки как образовательная среда: случай ALT Linux

Аннотация

Образовательный потенциал сообществ, занятых разработкой свободного ПО, сравним с образовательным потенциалом профильных высших учебных заведений. Однако эта тема весьма редко обсуждается на русском языке: возможно, в силу своей очевидности для участников сообщества и малого интереса в свободном ПО со стороны российской академической общественности. В данном докладе мне бы хотелось привлечь внимание аудитории к образовательным процессам, имеющим место в одной из традиционных форм коммуникации в сообществах СПО — списках рассылки.

Списки рассылки как образовательная среда

Модель свободной разработки ПО стала в последнее десятилетие объектом общественной рефлексии, различные ее социальные и культурные свойства обобщаются, проблематизируются и соотносятся с более традиционными. Среди прочего, исследователи и практики обратили внимание на образовательные возможности, которые такие сообщества открывают для своих участников [2]. В этом отношении их принято соотносить с более традиционными *сообществами практикующих* (communities of practice, [6]), с соответствующими неформальными приёмами обучения на практике. Понятно, что образовательный потенциал подобных сообществ лежит преимущественно в той предметной области, которой посвящена их деятельность, и осуществляется в используемых сообществом каналах коммуникации, преимущественно электронных.

Списки рассылки, как одна из форм существования сообщества, работает как ресурс для взаимосвязанных процессов *социализации в сообществе* и *обучения* [1]. Механизмы обучения сходны с ситуативным обучением [3], включают пассивное и активное овладение дискурсивными приёмами, характеризующими члена сообщества и одновременно профессионала в предметной области [4].

Если не рассматривать разработку ПО как самоцель, то определённая степень участия в списке рассылки может служить в первую очередь целям самообразования. С точки зрения обучающегося ресурс такого типа может быть привлекателен как свободой в отношении стоимости и времени (графика), так и отсутствием географической привязки, зачастую предоставляющий доступ к компетентным специалистам в предметной области (ср. онлайновые сообщества с образовательной целью, напр. обучение языкам [5]).

Случай ALT Linux

В этом разделе я рассмотрю списки рассылки сообщества ALT Linux Team с точки зрения именно образовательных возможностей, которые они могут предоставить.

Сообщество ALT Linux Team, занятое разработкой репозитория свободного ПО Sisyphus и выпуском дистрибутивов GNU/Linux на его основе, поддерживает списки рассылки как одну из основных форм коммуникации примерно с 2001 года. Важнейшая черта, отличающая этот проект от всех подобных, — русскоязычные списки рассылки, включающие всех ключевых разработчиков.

Деятельность ALT определяет две основные предметные области с точки зрения образования (обычно разнесённых по разным спискам рассылки для разной аудитории):

- *разработка Sisyphus* — специальные вопросы разработки ПО: сборка пакетов, организация репозитория, информационная безопасность, программирование и т. п.

- *выпуск дистрибутивов* — общие вопросы использования широкого спектра прикладного ПО, соотносимо с образовательной задачей непрофильных курсов информатики (*компьютерная грамотность*).

В докладе будут рассмотрены некоторые конкретные механизмы организации неформального образовательного процесса, работающие в списках рассылки ALT, проанализированы роли обучающих и обучающихся и «педагогические приемы», используемые в списках обеих названных предметных областей.

Вместо заключения

В контексте предстоящего внедрения комплекта свободного ПО в школах России, образовательный потенциал списков рассылки ALT, в первую очередь профильного списка junior@, может сделать очень важный вклад в общий успех проекта. Однако нужно учесть два обстоятельства:

1. Работа списка рассылки сама по себе не гарантирует, что будет происходить образовательный процесс; для этого необходима заинтересованность как со стороны «обучающихся», так и со стороны «обучающих».

2. Нельзя подходить к списку рассылки общественного проекта с теми же установками, что и к формальному образовательному ресурсу; здесь иная структура мотивации, и попытки формализации естественных процессов могут привести к потере интереса к списку у сообщества.

Литература

[1] N. Ducheneaut. Socialization in an Open Source Software Community: A Socio-Technical Analysis. *Computer Supported Cooperative Work (CSCW)*, 14(4):323–368, 2005.

[2] C.M. Johnson. A survey of current research on online communities of practice. *The Internet and Higher Education*, 4(1):45–60, 2001.

[3] J. Lave and E. Wenger. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press, 1991.

[4] J.A. Miller. *Promoting Computer Literacy through Programming Python*. PhD thesis, The University of Michigan, 2004.

[5] J. Simpson. *The Discourse of Computer-mediated Communication: A Study of an Online Community*. PhD thesis, University of Reading, 2003.

[6] E. Wenger. *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press, 1999.

Георгий Курячий Москва, ALT Linux

Проект: ALT Linux Junior
http://heap.altlinux.ru/engine/Edu/Soft/EducationalDistro

Итоги Школы Linux для преподавателей: компьютерный класс под управлением ALT Linux

Аннотация

Школа Linux для преподавателей (Переславль, 2008) посвящена организации компьютерного класса под управлением ОС GNU/Linux. В докладе рассматривается макет класса с использованием т. н. «толстых клиентов»: компьютеров, предназначенных для запуска пользовательских программ, но не нуждающихся в администрировании. В конце доклада будут рассмотрены результаты проведения Школы Linux с использованием макетного класса.

Класс на основе «толстых клиентов»

И сервер, и рабочие места — вполне обычные современные компьютеры. Главное отличие от «просто компьютерного класса» в двух вещах: (1) жёсткие диски рабочих станций либо не используются, либо не содержат критически важных данных, и могут быть восстановлены/размечены заново автоматически; (2) для изменения настроек, пакетного состава и т. п. на всех рабочих станциях достаточно один раз проделать эти изменения в специальной области на сервере.

Каждый клиент класса может работать в трёх режимах:

- Как *бездисковая станция*. В этом случае жёсткий диск не используется, а вся файловая система доступна по сети. Бездисковая станция может быть зарегистрирована как «администраторская», с правом модифицировать содержимое сетевого диска.

- Как *сетевая рабочая станция*. Жёсткий диск в этом случае автоматически устанавливается подготовленный администратором профиль операционной системы со всеми дополнениями и изменениями, а все пользовательские файлы доступны на сетевом диске.

- Как *автономная рабочая станция*. Этот режим — «аварийный», он нужен только если сервер по какой-то причине недоступен, или есть необходимость работать без локальной сети.

Таким образом, класс «толстых клиентов» требует гораздо меньше времени на сопровождение, причём особенно — в случае, когда класс необходимо постоянно адаптировать к изменяющимся требованиям.

Пользовательские свойства класса

- Единая авторизация (на сервере): пользователь может садиться за любую клиентскую машину и работать.
- Все пользовательские файлы находятся на сетевом диске сервера.
- Если сервер недоступен или отказало сетевое оборудование, можно работать в автономном режиме (понимая, что свои данные следует записать на съёмный носитель).
- Пользователю можно дать больше прав на модификацию установленной системы, так как переустановка системы производится автоматически

Эксплуатационные свойства класса

- На клиентских компьютерах может вообще не быть жёстких дисков.
- Администрировать нужно только один компьютер — сервер; в случае сбоев работы автономной системы на клиенте достаточно переустановки.
- «Прозрачная» подготовка профиля клиентской системы: администратор просто ставит и настраивает нужное ПО в режиме «бездисковая станция с правом записи», а получившееся дерево каталогов используется для загрузки «обычных» сетевых станций с этим профилем или установки его на жёсткий диск станции.
- Установка подготовленного профиля на все клиентские машины делается однократной командой с сервера.

- Можно создать *несколько* профилей для загрузки клиентских рабочих станций (например, с набором ПО и материалов только по конкретной дисциплине, с возможностью широковещательного показа действий преподавателя на ученических компьютерах и т. п.).
- Можно использовать заранее созданные разработчиками профили.

Если серверный компьютер не является рабочим местом для преподавателя, его ресурсы можно использовать для организации внутреннего WWW-сервера с элементами портала (CMS, Wiki, форум и т. п.), файлового сервера Windows-сетей и т. п.

Результаты Школы Linux

Школа Linux для преподавателей запланирована на период непосредственно перед началом Конференции, поэтому её результаты не могут входить в настоящие тезисы. Помимо знакомства с собственно Linux, на Школе планируется развёртывание класса и отработка основных его свойств.

Р. А. Ермаков Москва, КТЕ Лэбс

Свободное ПО и комплексная информатизация системы образования

Уже из заголовков докладов на конференции видно, что сообщество разработчиков свободного программного обеспечения (и не только оно) воспринимает задачи информатизации школ весьма однобоко. По большей части речь идет об использовании СПО непосредственно при обучении компьютерной грамотности, причем часто это понимается еще более узко — как обучение программированию. Часть выступающих уделяет также внимание применению компьютера в качестве средства для работы с образовательным контентом.

Однако, на наш взгляд, задача преподавания одной, пусть и важной, дисциплины не тянет на национальный проект, равно как и задача оснащения школ «классными досками нового тысячелетия», хотя бы и интерактивными.

Мы считаем, что нашей целью должно быть внедрение в школах идеологии свободного ПО, а не конкретных дистрибутивов Linux. Это возможно только при комплексном подходе к информатизации и внедрении компьютера в школьную практику не только как ещё одного ТСО, а как полноценного инструмента для всех компонентов учебного процесса и задач образовательного учреждения (ОУ). Для этого необходимо понимание функций школы (в широком смысле этого слова) в современном обществе. Школа сейчас является не только учебным, но и социальным учреждением, а также хозяйствующим субъектом. При этом школы не существуют в вакууме, а встроены в единую иерархическую систему образования Российской Федерации. Следует учесть, что в этой системе присутствуют не только общеобразовательные школы, но и учреждения профессионального образования, социальные учреждения, институты повышения квалификации педкадров и т. д.

В связи с этим задачи информатизации системы образования необходимо описывать в виде матрицы, где одним измерением являются основные блоки образовательной деятельности, а другим — уровни управления системой: отдельное ОУ, муниципальные органы образования, органы управления образованием на уровне субъекта федерации. Соответственно, возникают и задачи обмена информацией между этими уровнями.

Среди основных блоков можно выделить следующие:

1. Контингент (в т. ч. задачи обеспечения непрерывности обучения, контроля за детьми, «выпадающими» из системы образования);

2. Педагогические кадры (в т. ч. повышение квалификации, контроль нагрузки и потребностей в учителях);

3. Административное обеспечение учебного процесса (от составления учебных планов и контроля за успеваемостью до координации межшкольных мероприятий);

4. Разработка, распространение и использование образовательного контента, методическое обеспечение учебного процесса.

5. Социальный блок (внешкольная деятельность, работа с неблагополучными семьями, обеспечение учебного процесса для детей со специальными потребностями и т. д.);

6. Экономический блок («бухгалтерия», причём с учетом специфического зависимого положения и сложной системы тарификации).

Разумеется, все эти блоки тесно связаны между собой. К производным («надстроечным») блокам можно отнести задачи дистанционного образования, работы с родителями, контрольные и координационные задачи, например поддержка работы конфликтных комиссий и педагогических инспекций и др.

Нельзя забывать и об интеграционных задачах: информационное взаимодействие с милицией, органами опеки, медицинскими и социальными службами, региональными и муниципальными органами власти.

К сожалению, в большинстве указанных блоков сообществу разработчиков свободного ПО предложить нечего, т. к. данные задачи носят специальный и комплексный характер и вряд ли могут успешно разрабатываться в инициативном порядке. Только в условиях возросшего интереса государства к свободной модели распространения ПО соответствующие проекты могут быть осуществлены в рамках госзаказа. Однако на настоящий момент все они традиционно отданы на откуп проприетарных разработчиков, что ведёт и к низкому качеству продуктов, слабой интеграции, «зоопарку» решений в субъектах федерации и отдельных ОУ. Как и в случае с внедрением свободных ОС и прикладных пакетов, необходима координация наших усилий, создание консорциумов, обеспечивающих — в лучшем значении этого слова — лоббирование идеи информатизации системы образования на основе СПО.

Справочно: компания «КТЕ Лэбс» (Московские лаборатории экономики и технологии знаний) сотрудничает с Департаментом образования города Москвы и является разработчиком общегородских информационных систем «Кадры» (около 100 тыс. учителей в 1500 образовательных учреждений), «Контингент» (более 1,5 млн. детей), «Комплектование» (контроль очереди в 2500 детских садов Москвы) и др.

И. В. Зайцев Санкт-Петербург,
Санкт-Петербургский Торгово-Экономический Институт
Проект: http://www.spbtei.ru/

Использование открытых средств разработки на языке Java для образовательных целей

Аннотация

На основе опыта разработки обучающей игры и проведения с ее помощью занятий для студентов рассматриваются возможности использования открытых средств разработки на языке Java для создания ПО, применяемого в учебном процессе.

В процессе преподавания информатики и специальных дисциплин в компьютерных классах часто возникают идеи по расширению использования компьютерной техники

Для решения такого рода задач, как правило, требуется создание небольших обычных или серверных приложений, позволяющих автоматизировать некоторые фрагменты учебного процесса, интенсифицировать его или реализовать инновационные подходы к обучению.

При этом реализация новых идей не должна отнимать много времени у преподавателей. Одним из подходов к реализации идей является поиск подходящих бесплатных пакетов, «адаптация» идеи к возможностям выбранного пакета.

Такой подход, если повезёт, может дать быстрый результат, однако при этом имеет следующие очевидные недостатки:

- затраты времени на поиск, изучение возможностей, а также практическую проверку реализации этих возможностей непредсказуемы;
- положительный результат поиска и проверки совершенно не гарантирован;
- чужой программный продукт сложнее совершенствовать и адаптировать к изменяющимся требованиям (если используется несколько разных пакетов, для их адаптации придётся осваивать разные технологии).

Альтернативным подходом является самостоятельное написание необходимого программного кода с его последующей апробацией и совершенствованием в ходе учебного процесса. При этом желательно, чтобы используемый язык программирования обладал возможностями для решения широкого круга задач и для него имелась удобная бесплатная среда разработки. В полной мере этим требованиям отвечает разработанный компанией Sun Microsystems язык Java и интегрированная opensource среда разработки Netbeans, спонсируемая этой же компанией.

Всё необходимое для разработки на Java программное обеспечение бесплатно доступно на сайтах `http://www.sun.com/` (Java Developers Kit) и `http://www.netbeans.org/` (IDE Netbeans), причём для разработки серверных приложений можно скачать дистрибутив Netbeans, включающий в себя opensource сервер приложений Glassfish.

Все продукты великолепно интегрированы между собой, благодаря чему разработчик может значительно сэкономить время на подробном изучении процедур компиляции, сборки проектов и развёртывания, полагаясь на удачность установок по умолчанию, что вполне допустимо при создании ПО для учебных целей, для которых обычно оказывается достаточно производительности и надёжности, достигаемых без тонкой настройки.

В нашем учебном заведении автором с помощью вышеупомянутых бесплатных программных продуктов, установленных в операционной системе OpenSuse 10.1, была разработана обучающая игра, в которой часть студентов является владельцами интернет-магазинов, а остальные — покупателями. Такая игра, с одной стороны, способствует закреплению знаний по курсу информатики, а с другой стороны, позволяет моделировать реальные ситуации, требующие от студентов применения знаний, полученных при обучении другим дисциплинам, например, знания по использованию кредитов, рекламы, оптимизации.

В игре используется технология JSP-страниц (т. е. HTML-страниц с включениями Java-кода). Проект развёртывается на сервере приложений Glassfish, установленном на обычном ноутбуке. Студенты взаимодействуют с сервером с помощью обычных браузеров, т. е. никакого дополнительного ПО в компьютерных классах устанавливать не требуется. В отличие от готовых продуктов, данное решение позволяет легко вносить изменения и добавления по мере накопления опыта проведения занятий с использованием данной игры. В настоящее время работа над игрой продолжается.

И. Н. Трушкин

Пенза,
Пензенский Государственный Университет

Проект: Программная среда «Арифметик»

Разработка программной среды для программирования и доработки программных продуктов без нарушения лицензионного соглашения

Аннотация

Среда программирования получила название «Арифметик». Этот программный продукт не только объединил черты, необходимые для написания универсальных программных модулей обработки данных, но и прост в использовании и доступен людям, имеющим среднее профессиональное образование. Среда работает с основными языками программирования C, Pascal, Basic, Assembler. Основным языком написания сценариев среды является специально разработанный язык, максимально приближенный к литературному русскому.

Сегодня, во времена тщательного исполнения закона об авторском праве, необходимостью является использование лицензионного программного обеспечения. Однако не все пользователи могут себе позволить приобрести полную (профессиональную) версию нужного программного продукта, особенно это касается учебных программ. Кроме того, зачастую сами компании-разработчики предлагают программы, не обладающие всеми необходимыми пользователю свойствами. Часто из-за этого возникает задача «доработки» приобретённого программного обеспечения методом включения подпрограмм, выполняющих конкретные функции. Однако при этом необходимо соблюдать лицензионное соглашение, как правило, запрещающее производить изменение программного кода, что ранее не давало возможности проводить модернизацию программного обеспечения.

Была предложена гипотеза, подтвердившаяся на практике, что решением задачи модернизации программного обеспечения может быть создание и внедрение пользовательских модулей без изменения исходного кода целевого программного обеспечения. В качестве исходных моделей работы программного обеспечения рассматривались из-

вестные структуры создаваемых документов и форматы файлов с целью создания среды, обеспеченной встроенным языком программирования, позволяющим создавать модули, содержащие обработчики и фильтры, позволяющие любым программам работать с тем или иным форматом файлов и выполнять действия, не предусмотренные разработчиками этих программ. Наличие встроенного языка программирования необходимо для создаваемой среды, так как его отсутствие значительно ограничивало бы область применения этой среды, например, в случае отсутствия или недоступности необходимого для работы системы модуля.

Свойствами предлагаемой среды является:

- дополнение функций ранее созданных программ (например, с появлением новых, более компактных, удобных и защищённых форматов файлов старые программы не должны лишиться возможности работать с данными, сохранёнными в этих форматах);

- решение задач для обработки файлов программами, изначально не предназначенными для работы с этим форматом (например, отсканированный набор чертежей необходимо обработать и сохранить в программе для редактирования векторной графики, или открыть в MS Word отсканированный текст в формате JPEG и иметь возможность внесения в него изменений);

- наличие инструментария для выполнения необходимых преобразований (программы часто не имеют интерфейса для взаимодействия с существующими программными продуктами, при этом использование технологии COM не позволит решить данную проблему, так как часто приложения не используют COM-автоматизации при открытии документов, а имеют встроенный конвертер форматов).

Таким образом, предлагаемая среда[1] позволяет не только создавать модули расширения инструментария программ, но и предостав-

[1]Система разработки и модернизации программного обеспечения «Арифметик» зарегистрирована в ОФАП (свидетельство об отраслевой регистрации разработки № 6497 от 29 июня 2006 г). Доклад по данной системе и приложениям, построенным на её основе, был представлен на конференциях в Москве и Новочеркасске в 2003 и 2005 гг. Имеется акт внедрения от ОАО МСС «Поволжье» программного средства «Радар 2003», построенного на базе «Арифметика»

лять уже созданным программам и модулям интерфейс взаимодействия с другими программами без изменения их кода.

Работоспособная среда получила название «Арифметик». Этот программный продукт не только объединил черты, необходимые для написания универсальных программных модулей обработки данных, но и прост в использовании и доступен людям, имеющим среднее профессиональное образование. Среда работает с основными языками программирования: C, Pascal, Basic, Assembler. Основным языком написания сценариев среды является специально разработанный язык, максимально приближённый к литературному русскому.

Учитывая появившееся возможности, созданная среда имеет в своём составе инструменты для интеграции с уже существующими программными продуктами и внедрения пользовательских модулей в готовые приложения. Предусмотрена возможность расширения инструментария программ без внесения изменений в их код.

Среда предполагает открытый исходный код, так как текст программы не преобразуется ни в бинарный, ни в P-код, а интерпретируется непосредственно.

А. А. Трушкина

Пенза,
Пензенский Государственный Университет

Проект: Программная среда «Арифметик»

Практическое применение среды программирования «Арифметик» при разработке программного обеспечения обучающих комплексов

Одним из направлений развития искусственного интеллекта являются обучающие автоматизированные программно-аппаратные комплексы и тренажёры, в которых оператор должен учиться взаимодействовать с моделью реального пространства. Для этого в состав тренажера вводится имитатор визуальной обстановки (ИВО), моделирующий трехмерное виртуальное пространство. При этом, кроме задачи создания визуальной модели, в которой обучаемый мог бы отрабатывать навыки взаимодействия с наблюдаемым пространством, возникает задача размещения в модели реально существующей местности

движущихся объектов, инженерных сооружений, элементов пейзажа. Последняя задача требует учёта рельефа местности.

Решение проблемы моделирования естественной складчатой поверхности возможно с помощью метода триангуляции (опорные геодезические сети). Однако известные варианты их решения предполагают равномерное разбиение или разбиение с учётом известных реперных точек. Исследования показали, что при такой постановке актуальным является вопрос оптимизации размеров и вида триангуляционных треугольников метом их объединения и взаимопоглощения при выполнении ряда условий.

В ИВО, работающих в реальном масштабе времени с полным циклом обновления информации не более 120 мсек, важным вопросом является соотношение качества генерируемого изображения и затрачиваемых машинных ресурсов. Существует несколько вариантов решения этой задачи: разработка аппаратной платформы, позволяющей проводить параллельные вычисления; увеличение скорости обработки данных (что предполагает как совершенствование аппаратной базы, так и разработку новых алгоритмов); предварительная обработка полученных результатов и группировка их таким образом, чтобы при работе в реальном масштабе времени затрачивался минимум времени. Анализ последнего направления показал, что при решении задачи «в лоб» требуется выполнить обработку видеоизображения, разбиение сцены на сегменты, дополнительно проработать геодезические данные реальной местности и смоделировать целевую визуальную обстановку, разместить возможные движущиеся объекты с учётом их взаимозакрываемости. Опыт показал, что это достаточно сложная задача даже для профессионального разработчика, требующая достаточно много времени.

Нами предлагается специальное программное обеспечение с элементами искусственного интеллекта, которое позволяет в качестве исходных данных на проектировании использовать видеозаписи и геодезические планы местности для дальнейшего автоматизированного моделирования трехмерной обстановки с учётом особенностей субъективного восприятия обучаемого. Основным отличием предлагаемого подхода является ориентация на пользователя, имеющего среднее образование. С этой целью при разработке программ максимальное значение уделяется пользовательскому интерфейсу, позволяющему методом визуальной корректировки обрабатывать изображения с учётом предлагаемых пользователем изолиний для разбиения визу-

альной сцены на сегменты и для слияния изображений с нескольких камер с корректировкой цвета и с возможностью корректировки полученных результатов по дополнительной информации, имеющейся на геодезических планах или картах, при наличии видеоснимка этого же участка.

В качестве среды программирования использован «Арифметик», позволяющий быстро разрабатывать программы на специальном скриптовом языке, использовать встроенный инструментарий для создания нейронных сетей. Кроме того, в отличие от дорогостоящих и громоздких сред программирования других производителей, он мобилен, компактен, доступен и не требует установки.

Дальнейшее совершенствование программного обеспечения ведется в направлении разработки нового класса фильтров графических изображений, внедрения алгоритмов, реализующих оптимизацию сегментов визуальной сцены с учётом снижения затрат машинных ресурсов и повышения итогового качества изображения, что позволит в итоге размещать инженерные сооружения на местности с учетом имеющегося рельефа, а также управлять подвижными объектами, в том числе и при движении группы моделей на узнаваемом участке местности (например в ИВО автотренажёра групповой подготовки водителей взаимодействию моделей управляемых ими транспортных средств между собой и с моделью рельефа местности, по которой они двигаются), что существенно повысит качество обучения.