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

Шестая конференция «Свободное программное обеспечение в высшей школе»

Переславль, 29–30 января 2011 года

Тезисы докладов

Москва,
Альт Линукс,
2011

Шестая конференция «Свободное программное обеспечение в высшей школе»: Тезисы докладов / Переславль, 29–30 января 2011 года. М.: Альт Линукс, 2011. — 94 с. : ил.

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

ISBN 978-5-905167-03-4

Программа конференции

29 января

11.30 А. Е. Новодворский. Открытие. Информация оргкомитета

11.45 С. М. Абрамов. Приветственное слово

12.00 Л. М. Ухлинов. Технологическая платформа «Национальная Программная Платформа» и Высшая школа

13.00–13.20 Кофе-брейк

13.20–15.00 Круглый стол «Национальная Программная Платформа» и Высшая школа. Ведущий: С. М. Ухлинов. Выступают: А. В. Смирнов (ALT Linux), Н. Н. Непейвода (УдГУ) и др.

Н. Н. Непейвода

15.00–16.30 Обед и заселение

Вечернее заседание
17.00–19.00

В. В. Рубанов, Д. В. Силаков, А. В. Хорошилов

А. Г. Кушниренко, А. Г. Леонов, В. В. Яковлев и др.

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг и др.

30 января

Утреннее заседание
10.00–13.00

11.40–12.00 Кофе-брейк

13.20–14.20 Обед

Дневное заседание
14.20–19.00

Вне программы

Н. Н. Непейвода
Ижевск, УдГУ

О Национальной программной платформе и не только об образовании

Несколько дней назад я познакомился с проектом национальной программной платформы. Я поступлю не по Карнеги: начну с самых ярких впечатлений, а не с положительных слов.

Многие отмечают духовную импотенцию т. н. «элиты» современного русского общества. Президент Медведев наивно сокрушался, что вкладываем в науку большие средства, а отдачи нет. Этот проект четко показывает, почему так. Он состоит из «верняков». Нет изюминки, того семени, из которого рождается действительно нечто новое и большое. Соответственно, он не сможет оплодотворить почву.

Конечно же, можно сказать, что чиновники требуют план работ на пять лет вперед. Но какая же это инновация, которую можно рассчитать на пять лет? А механизма быстрого подключения к проекту новых идей, исключающего чиновничьи согласования и любые согласования вообще, кроме экспертизы специалистов, нет. Чиновники могут возразить, что какие же идеи без бизнес-плана? Наоборот, с бизнес-планом это уже не идея. Идея должна быть рискованной. Маленькие деньги на новое должны даваться быстро и без оформления громадного количества идиотских бумажек, которые многие творческие личности просто побрезгуют писать. Если девять из десяти таких идей провалятся, то одна оставшаяся с лихвой окупит все.

Далее. С моей профессиональной точки зрения, в области образования проект просто никуда не годится и противоречит сам себе. Правильно говорится о том, что за пять лет в информатике происходит замена подавляющей части инструментария. Далее говорится о необходимости опережающего обучения. И тут же декларируется, что нужно перестраивать образование под «новейшие технологии», а уже на втором месте, что каждый бюрократ сочтет как второстепенное и необязательное требование, «при сохранении фундаментальности». Напоминаю еще раз: если на первом курсе начинать учить новейшим технологиям, то из вуза выходит специалист, знающий устаревшие.

Вот если бы было сказано, что недопустимы такие скандальные факты, как «выигрыш тендера» какой-то фирмой-производителем конкретных систем на составление стандарта и примерного учебного плана для информатиков, то хоть что-то полезное было бы сделано. Ведь эта фирма явно будет использовать вузы как дешевую и эффективную маркетинговую среду для своих продуктов, и о каком же «опережающем и фундаментальном» образовании тогда может идти речь? Если говорить с некоторым чувством юмора, то даже государственное ведомство типа Министерства обороны в качестве победителя такого тендера было бы лучше.

Представляется принципиально неверным конъюнктурное использование термина «опережающее обучение» для образования в столь широкой и непредсказуемо меняющейся области, как информатика. Здесь нужно *устойчивое обучение*. Специалист должен суметь сориентироваться в любом мире, в который он попадет, и сохранить это умение как минимум на двадцать лет после выпуска.

Единственный путь к этому — фундаментальное обучение, но такое, когда абстрактные знания немедленно связываются с *возможностями* их практического использования, а практические системы выбираются по принципу той же фундаментальности и ортогональности, а не новизны и конъюнктурности. Поэтому здесь нужно прежде всего использовать классические системы, которые прошли проверку временем и не содержат ничего лишнего и случайного, и выбирать их по принципу полноты их базиса, охватывающего все стили программирования и показывающего их совершенно различные логики. После такого на выходе из вуза специалист умеет быстро войти в любую конкретную среду и в любую предметную область. А подготовка операторов или программистов какой-то конкретной прикладной системы, даже очень большой — дело не вузов, а фирменных курсов.

Это не значит, что я против преподавания, например, такой почтенной системы, как 1С, в вузе. Ему свое место: на последнем курсе на правах спецкурса по выбору студента. И так же для любой конкретной «новейшей» системы. Это место нужно зарезервировать и отдать на выбор самим вузам, а не включать в общие программы. Ведь здесь зачастую даже за год не предскажешь, какой интересный специалист с интересной технологией появится в поле зрения и пожелает работать в данном вузе.

В числе авторов НПП я не увидел ни одного *творчески работающего* педагога из высшей школы. Их не так уж много в России, и

можно было бы кого-то привлечь. С этим, видимо, связана принципиально неверная ориентация НПП в области образования.

Еще одно место, где НПП явно бьет мимо цели — суперкомпьютеры. Когда ставится цель эффективно использовать возможности суперкомпьютеров, телега водворяется впереди лошади. Я уже говорил в своих выступлениях в образовательном сообществе, что ректор вуза, хвастающийся терафлопами, на самом деле хвастается своей организационной беспомощностью. Ведь если бы пятую чать этих денег вложить в мозги, то через три года он получил бы еще лучшие мозги, а любая такая суперсистема через три года окажется морально устаревшей и деньги выброшены на ветер. Вопрос не в том, как загружать системы, построенные ради престижа или «освоения» ассигнований, выделенных некомпетентными чиновниками под красиво звучащие темы, а в том, как быстро создавать суперкомпьютеры под реально возникающие задачи. И ценнейшие наработки здесь, скажем, у наших переславльских хозяев, есть.

А теперь о положительных факторах. В НПП нет сверхценной идеи типа той, которая была в японском проекте 5-го поколения, и это замечательно. В принципе для проекта на пять лет сделать упор на стандартизацию и интерфейсы — совершенно правильно. Только нужен еще эффективный и понятный интерфейс проекта с творческими личностями. Разработка такого интерфейса была бы громадным шагом вперед во всей организации российской науки, а не только в информатике. В НПП решительно разделались с попытками монополизма какого-то направления, а тем более фирмы. Особенно ценно, что такую позицию заняли и некоторые фирмы, являющиеся фактическими монополистами в своей области в России.

И, наконец, о предложениях. Есть одна фраза, которая, может быть, сыграла бы роль «изюминки». Русские специалисты выигрывают тогда, когда они вовсю применяют лучшие стороны русского негативного мышления, являющегося естественной альтернативой англосаксонскому позитивному, которое пытается возвести себя на роль «общечеловеческой ценности». Может быть, эту идею явно озвучить в НПП?

Далее, нужно поставить целью НПП создание саморегулирующейся устойчивой среды для информационных технологий, включающей и правовые, и человеческие, и программные компоненты. Хоть один шаг к этому был бы большим достижением.

В. В. Рубанов, Д. В. Силаков, А. В. Хорошилов
Москва, ИСП РАН
http://linuxtesting.org/

Программы поддержки студентов и аспирантов для работы в открытых проектах

Аннотация

В настоящее время в мире существует целый ряд программ и мероприятий, нацеленных на привлечение студентов и аспирантов к работе в рамках открытых проектов. На сегодняшний день, доля учащихся из России в таких мероприятиях невелика. В то же время, участие в них крайне полезно — помимо финансовой поддержки, студенты получают опыт работы над реальными проектами, востребованными сообществом. Основываясь на опыте ИСП РАН по отбору и оценке заявок в различных программах и конкурсах, в нашем докладе мы постараемся дать рекомендации студентам по грамотному представлению своих предложений для участия в такого рода программах.

Программы для студентов

Участие в открытых проектах для студентов крайне полезно — помимо навыков программирования, они получают опыт работы над реальными проектами, востребованными сообществом, а также опыт общения с сообществом и работы в распределенной команде.

По всему миру регулярно проводятся различные конкурсы, программы и прочие мероприятия, призванные стимулировать участие студентов в открытых проектах. Часто такие мероприятия позволяют не просто поучаствовать в разработке и сделать что-то полезное, но и получить за это материальное вознаграждение.

Наиболее известной программой такого рода является Google Summer of Code™ (GSoC)[1], в рамках которой оплачивается работа студентов и аспирантов со всего мира над открытыми проектами в течение лета. По ее образу и подобию организованы такие мероприятия, как Ruby Summer of Code[2], Season of KDE[3], Fedora Summer

[1] `http://code.google.com/soc`
[2] `http://rubysoc.org`
[3] `http://blog.lydiapintscher.de/2010/04/26/gsoc-and-season-of-kde-2010`

Coding[4] и другие. Зачастую в рамках таких программ поддерживаются проекты, для которых не хватило мест в GSoC.

Помимо регулярных (как правило, ежегодных) программ время от времени проводятся и одноразовые мероприятия и конкурсы. Например, в 2006 году ИСП РАН совместно с Novell проводил конкурс работ студентов и аспирантов, направленных на улучшение стандарта Linux Standard Base, а также на улучшение совместимости дистрибутивов линейки SUSE с этим стандартом.

Не все мероприятия подобного рода направлены на написание кода — например, Season of Usability[5] посвящен повышению удобства использования свободного ПО.

К сожалению, нельзя сказать, что студенты и аспиранты из России проявляют большую активность в таких программах. Так, по количеству участников Google Summer of Code Россия не входит даже в первую десятку, уступая, помимо прочего, Польше и Шри-Ланке. На наш взгляд, одной из причин такой ситуации является недостаточное понимание процесса представления, рассмотрения и оценки предложений для участия в той или иной программе и, как следствие, неумение грамотно составить собственную заявку.

Составление заявок

Подача заявок студентов и их оценка организаторами является одним из ключевых этапов многих программ и конкурсов. Авторы доклада имеют опыт оценки подобных заявок; в частности, в 2009—2010 годах мы выступали в роли менторов в проектах консорциума Linux Foundation в программе *Google Summer of Code* (см. следующий раздел). На примере этой программы, мы и рассмотрим представление студентами своих предложений.

Программа стартует в начале марта; в это время организации, желающие получить студентов для работы над своими открытыми проектами, подают заявку в Google, включая идеи по конкретным проектам. Администраторы из Google оценивают эти заявки и выбирают организации, которые будут допущены к участию в программе. Такие организации получают статус менторских организаций (mentoring organization) и им выделяется определенная квота на количество про-

[4] `http://fedoraproject.org/wiki/SummerOfCode`
[5] `http://season.openusability.org/`

ектов, которые будут финансово поддерживаться Google, после чего организация объявляет конкретные предложения для студентов по конкретным проектам.

Затем в течение одной-двух недель (в 2010 году — с 29 марта по 9 апреля) студенты подают заявки на выполнение проектов. Оценка предложений студентов занимает около двух недель, и в конце апреля оглашаются результаты. Каждому студенту назначается ментор из числа сотрудников менторской организации, который отвечает за работу студента в проекте. Студентам и менторам отводится месяц на знакомство друг с другом и проектом, а в конце мая начинается собственно «лето кода» — студентам предлагается начинать писать код, а Google начинает рассылать первые платежи.

В середине июля менторы оценивают прогресс студентов — если есть серьезные сомнения в успехе, то проект останавливается. К середине августа все работы завершаются, и менторы подводят итоги проектов.

Основываясь на собственном опыте, а также на опыте коллег из других организаций, мы сформулировали следующие рекомендации для студентов, желающих принять участие в программе:

1. При выборе проекта, необходимо ознакомиться с требованиями конкретных организаций к навыкам студента — требования к степени знакомства с кодом продукта и с используемыми технологиями в разных проектах сильно различаются.
2. Желательно как можно раньше «влиться» в сообщество, сформировавшееся вокруг продукта. Активное участие в жизни сообщества, даже не связанное с написанием кода (ответы на вопросы, полезные замечания в рассылках и чатах) является одним из лучших способов продемонстрировать свое знание продукта. Кроме того, полезно заранее оценить, насколько студент и сообщество подходят друг другу в плане стиля общения и подходов к разработке ПО.
3. При наличии собственной идеи проекта, вполне возможно и часто очень целесообразно проактивно предложить ее сообществу (включающему потенциальных менторов).
4. Даже при подаче заявки на идею, предложенную организацией, не стоит ограничиваться формальным заполнением анкеты на сайте. Полезно выйти на связь с потенциальными менторами — составить представление о студенте в ходе неформального

общения гораздо проще. Многие организации поощряют предварительное общение студентов с менторами, до подачи заявки на сайте.

5. До составления заявки, необходимо четко представить себе детальный план и сроки работ. Этапы проекта с описанием промежуточных результатов должны быть представлены в заявке.
6. При составлении заявки, необходимо придерживаться формы, предоставляемой организацией. Многие организации размещают образцы заявок или успешные заявки предыдущих лет в качестве примеров.
7. В самой заявке, не стоит дублировать содержание идеи проекта. Идея может быть достаточно абстрактной, заявка должна быть более технической.
8. Одним из ключевых аспектов заявки является обоснование необходимости выполнения проекта с точки зрения студента — какую выгоду получит целевой продукт и его пользователи в результате. Практика показывает, что студенты не всегда осознают цели проекта; одним из признаков такого недопонимания считается простое копирование целей из описания идеи от организации.
9. При описании опыта разработки, стоит избегать абстрактных фраз, и приводить больше конкретных фактов — завершенные проекты, найденные ошибки, предложенные патчи.

Наконец, ряд материалов, обобщающих опыт различных организаций, доступен на страничке Google с советами для студентов[6].

Безусловно, часть приведенных рекомендаций специфична для *Google Summer of Code* и схожих с ней программах, однако мы надеемся, что они помогут и при участии в других мероприятиях.

Заключение

В 2009—2010 гг. авторы доклада выступали в качестве менторов по следующим проектам в программе GSoC:

1. Make OpenJDK LSB Compliant
2. Annotation Support for API Sanity Autotest

[6] `http://code.google.com/p/google-summer-of-code/wiki/AdviceforStudents`

3. Linux Device Drivers Quality Inspector
4. LSB Test Result Analytic System
5. Linux Upstream Tracker
6. Cross Distribution Package Dependency Translation

Все проекты были успешно выполнены. Google также сдержал все обещания по финансовой поддержке и даже прислал отдельные подарки.

Авторы считают, что участие российских студентов и аспирантов в подобных мероприятиях нужно расширять. Кроме финансовой поддержки это дает возможность участникам получить опыт работы в международном сообществе, и даже процесс оформления заявок для подобных программ сам по себе ценен, так как формирует умение грамотно представлять свои идеи, обосновывать предлагаемые решения, а также демонстрировать свою компетентность в тех или иных вопросах. Подобные навыки являются ничуть не менее ценными, чем опыт написания кода.

А. Г. Кушниренко, А. Г. Леонов, К. А. Прокин, М. А. Ройтберг, В. В. Яковлев
Москква, НИИСИ РАН

Проект: КуМир и ПиктоМир http://www.piktomir.ru/

ПиктоМир: опыт использования и новые платформы

Аннотация

В прошлом году появилась система программирования для дошкольников «ПиктоМир», ориентированная на составление программ из графических примитивов — пиктограмм. В докладе будет рассказано о том, каких успехов удалось добиться при работе с реальными детьми, и о том, что планируется сделать в ближайшее время.

ПиктоМир на практике

Система ПиктоМир прошла две итерации реальной апробации с на детях младшего школьного и дошкольного возраста с последующими доработками системы.

В марте на проходившей в г.Пущино ежегодной Зимней Школе был проведен курс программирования для младших школьников.

В течение недели дети налетали на шероховатости системы, которые оперативно устранялись.

Затем, уже летом, аналогичный курс проведен при проведении 5-дневного цикла занятий в Мультиматематической летней школе (директор-организатор М.Сегинева), которая проходила на базе Филипповской школы г. Москвы. На данной итерации был уточнен предлагаемый набор заданий.

За 5 занятий по 2 часа детям удалось успешно освоить составление программ и такие понятия как «подпрограмма» и «цикл». Некоторые дети также освоили условные переходы, другие до них не дошли за отведенное время.

Что не удалось — дети так и не научились пользоваться отладкой методом выполнения программы по шагам. Им проще отлаживаться методом проб и ошибок.

Кроме того, явно обнаружились дополнительные требования к оборудованию: необходим достаточно большой экран (нетбуки оказались непригодны по этой причине), а некоторым детям — которые помладше — необходима мышь как можно меньшего размера.

ПиктоМир для портативных устройств

Дети дошкольного возраста (3 года) интуитивно пытаются нажимать на экран, не очень хорошо владея мышью. Тут очевидно напрашивается идея, что для снижения планки возраста освоения программирования возможно только на устройствах с сенсорным экраном (touchscreen). Это могут быть не только планшеты вроде Apple iPad, но также и любые другие (как правило — более доступные) устройства: автонавигаторы и мобильные телефоны.

На данный момент существуют планы по реализации ПиктоМира на следующих программно-аппаратных платформах:

1. мультимедийные устройства на базе WindowsCE;
2. планшетные компьютеры и мобильные телефоны на базе Android;
3. планшетные компьютеры на базе Maemo;
4. Apple iOS (iPod/iPhone/iPad).

Реализация версий для WindowsCE и Maemo основана на единой с настольной версией кодовой базе. Реализации для Android и iOS — это отдельные задачи.

На данный момент существуют прототип мобильной версии для WinCE/Maemo и завершается работа над прототипом для Android. Также ведется работа по портированию на мобильную платформу Apple.

После реализации первого прототипа (который был для нас «пробой пера») объявились некоторые особенности, которые в мобильной версии необходимо перепроектировать. Например, стало ясно, что нежелательно использование 2.5D графики, поскольку это во-первых дополнительная нагрузка на блок операций с плавающей точкой (слабое место ARM-процессоров), а во-вторых — и без того дефицитное пространство на экране расходуется нерационально (из-за потерянного пространства по углам).

Данные прототипы станут доступны для скачивания после нескольких итераций тестирования и доработки.

ПиктоМир на iPhone (iPad и пр.) имеет уникальный код, в нем модифицирован дизайн и для увеличения привлекательности, следуя тезису «Учись программировать играя!», добавлена история-сказка и элементы соревнования для повышения мотивации. Для создания сценария заданий имеется визуальный редактор.

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг,
Т. Р. Джелядин, А. А. Ефремов, А. В. Карпов, Д. В. Хачко,
В. В. Яковлев
Москва, Пущино, НИИСИ РАН, АльтЛинукс

Проект: КуМир http://www.niisi.ru/kumir/

Новые возможности системы КуМир

Аннотация

В системе КуМир используется школьный алгоритмический язык с русской лексикой и встроенными исполнителями Робот и. Чертежник. Система КуМир отличается от производственных систем программирования на порядок более высоким уровнем дружественности при отладке небольших учебных программ. В 2010 г. в КуМир появились новые возможности: поддержка подготовки и выполнения курсов, а не только отдельных заданий, средства разработки новых исполнителей, средства, связанные со сдачей ЕГЭ.

Сегодня, в начале 21 века, можно прожить, не зная высшей математики, но трудно прожить, не умея складывать, вычитать и умножать хотя бы двузначные числа, не зная азов арифметики, не понимая простейших формул, не понимая языка графиков и диаграмм. Точно так же с каждым годом будет все труднее прожить, не умея хоть чуть-чуть программировать, не понимая азов программирования.

В науке числа и формулы помогают понятнее объяснить физику, биологию, экономику, в промышленности и в быту числа и формулы используются как часть повседневного языка, на котором люди общаются, решая производственные и личные задачи. Точно так же освоение основных понятий программирования помогает члену современного информационного общества как в учебе, в освоении наук, так и в производстве и в быту, где все большую часть времени мы проводим, пользуясь информационными технологиями.

Система КуМир ориентирована именно на начальный этап обучения программиолванию, на массовое обучение. В системе КуМир предложен ряд решений, которые могли бы оказаться полезными и в средах программирования на производственных языках.

А) постоянный синтаксический контроль правильности. 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.

Б) Прозрачный отладчик. 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. В пошаговом режиме КуМир показывает на полях результаты присваиваний и проверок условий. Это позволяет новичку составлят и отлаживать простейшие программы, не пользуясь командами ввода-вывода и обходясь без отладчика.

Основным нововведением 2010 года можно считать поддержку системы курсов, которая позволяет автоматизировать процессы получе-

ния заданий учеником и проверки заданий учителем. Система была апробирована Д.П. Кириенко в школе №179 и более подробно описана в докладе «Поддержка курсов в системе КуМир» [1]. Эта система позволяет учителю значительно увеличить количество задач, которые решают ученики.

Отработана технология создания новых миров-исполнителей. На данный момент в КуМире функционируют исполнители «Водолей», «Кузнечик», «Черепаха». Помимо этих исполнителей, сейчас готовятся новые исполнители для очередной поставки: это исполнитель «Роботор» [2] и исполнитель «Рисователь», создаваемый по спецификациям, предложенным д.т.н. проф. К.Ю. Поляковым, преподавателем информатики школы 173 г. Санкт-Перербурга [3]. Советы и замечания К.Ю.Полякова были очень полезны при работе над системой КуМир в 2010 г.

Осенью 2010 г. система КуМир была использована при проведения пробного ЕГЭ по информатике и ИКТ в компьютеризированной форме [4]. Для этого была реализована специальная версия системы КуМир-ЕГЭ, а также автономная версия интерпретатора, который использовался в связке с системой ejudge для автоматизированной проверки решений задач ЕГЭ. В ходе курсов [5], которые были проведены в рамках подготовки к проведению ЕГЭ в компьютерной форме, с системой КуМир познакомились десятки учителей мз всех регионов страны.

Литература

[1] Д. В. Хачко, Д.П. Кириенко, А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг. Поддержка курсов в системе КуМир, наст. сборник.

[2] А.Г. Кушниренко, А.Г. Леонов, А.А. Ефремов. Космический робот РобоТор. Информатика. № 21 1—15 ноября 2010. Издательский дом «1 сентября».

[3] `http://kpolyakov.narod.ru/index.htm`

[4] А.Г. Кушниренко, А.Г. Леонов, В.Р. Лещинер, Д.А. Путимцев, П.С. Шестаков. Структура программного обеспечения для проведения ЕГЭ по информатике и ИКТ в компьютеризированной форме, наст. сборник.

[5] http://www.mos-cons.ru/mod/forum/view.php?f=227

Д. В. Хачко, Д.П. Кириенко, А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг
Москва, Пущино, Научно-исследовательскии институт системных исследовании РАН, Институт математических проблем биологии РАН, Средняя школа №179 г.Москвы
http://lpm.org.ru/kumir/

Поддержка курсов в системе КуМир

Аннотация

Модуль поддержки курсов (МПК) для системы КуМир позволяет учителю создавать, а ученику выполнять учебные курсы по информатике на базе системы КуМир. Во время сеанса работы МПК передает системе КуМир тексты заготовок программ и описания обстановок, получает информацию о результатах тестирования и тексты подготовленных пользователем программ для последующего сохранения в файле состояния выполнения. Использование модуля значительно повышает количество задач, которые может решить ученик.

Структура курса

Под *курсом* мы понимаем:

1) систему заданий, которую должен выполнить ученик;
2) *методику,* которая описывает рекомендованный порядок выполнения заданий.

Задание предлагает ученику написать программу, удовлетворяющую предлагаемому описанию. *Описание задания* включает в себя:

1) текстовое описание задания;
2) шаблон программы и, если нужно, обстановку исполнителя;
3) программу тестирования и, если нужно, обстановки, используемые при тестировании.

Прохождение курса

При выполнении курса учеником, МПК для каждого задания помнит текущую *оценку* — целое число от 0 до 10. Оценка показывает, выполнял ли ученик данное задание и, если да, — насколько успешно он справился с этим заданием.

Описание курса включает в себя *методику* — указание на то, какие задания доступны для выполнения в зависимости от уже полученных оценок. Простейшие методики — это «свободная» методика, когда, независимо от полученных оценок, доступны все задания, и «игровая», когда задания линейно упорядочены и очередное задание доступно только после выполнения всех предыдущих. В более сложных случаях методика может выделять в курсе, например, задачи повышенной трудности, которые становятся доступными только при хороших оценках за задания основного курса, и вспомогательные задачи, которые предлагаются ученику, если он не справляется с некоторыми основными заданиями.

Проверка задания

Проверка правильности выполнения задания выполняется с помощью тестирования. Разработчик курса для каждого задания готовит систему тестов. Эта система оформляется как (возможно, скрытая) часть заготовки программы.

При получении запроса на тестирование программы, МПК вызывает алгоритм без параметров со стандартным именем *@тестирование.* Этот алгоритм может, если нужно, вызывать другие алгоритмы. Алгоритм *@тестирование* и его вспомогательные алгоритмы должны быть включены в заготовку программы как неудаляемые (и, возможно, невидимые) фрагменты. Эти алгоритмы могут вызывать программу пользователя (ее заголовок фиксируется в заготовке как неудаляемый, но видимый, фрагмент) шаблона.

Выполнение задания считается успешным ,если тестирование прошло успешно на всех тестовых обстановках для данного задания.

Подготовка курса

Курс состоит из файла описания в XML формате. В нем хранится дерево заданий. И дополнительных файлов (шаблоны, описания ,обстановки). Учитель может редактировать XML файл любым удобным редактором или воспользоваться специализированным редактором заданий, встроенным в МПК.

Литература

[1] С курсами, разработанными Д.П.Кириенко, можно ознакомиться по адресу: http://server.179.ru/wiki/wakka.php?wakka=Informatika/7B&v=13q& (ссылка: «Скачать архив с курсом заданий для выполнения дома»)

А. Г. Кушниренко, А. Г. Леонов, В. Р. Лещинер,
Д. А. Путимцев, П. С. Шестаков
Москва, НИИСИ РАН, МИОО

Проект: http://ktsege.mioo.ru/

Структура программного обеспечения для проведения ЕГЭ по информатике и ИКТ в компьютеризированной форме

Осенью 2010 года Московский институт открытого образования (МИОО) успешно завершил выполнение госконтракта по разработке ПО для проведения ЕГЭ по информатике и ИКТ в компьютеризированной форме. В докладе описана структура разработанного ПО.

Ядром разработанного ПО является система КТС ЕГЭ (Контрольно-Тестирующая Система Единого Государственного Экзамена). Это система состоит из выделенного веб-сервера, на котором хранятся задания ЕГЭ и текущие результаты выполнения экзаменационной работы школьниками, и стандартных браузеров, установленных в узлах экзаменационной сети (локальной или глобальной). Ввиду открытости на настоящий момент списка используемых специализированных ИКТ инструментов и сред программирования, отсутствия нормативной базы для выбора таких инструментов, выполнение требований госконтракта возможно только по открытой схеме, путем обеспечения возможности связи КТС ЕГЭ с достаточно произвольными внешними программными системами. Эту связь КТС ЕГЭ должна обеспечить как на этапе выполнения экзаменационной работы, так и на этапе ее проверки. В предлагаемой реализации КТС ЕГЭ это реализовано следующим образом.

При необходимости обеспечить во время экзамена использование специализированного ИКТ-инструмента, например, среды программирования Free Pascal, на рабочем месте обучаемого помимо кли-

ентской части КТС ЕГЭ должен быть установлен Дополнительный Интерфейсный Модуль (ДИМ) — собственно система программирования, работающая на компьютере обучаемого в диалоговом режиме. Параллельно с ДИМ должен быть подготовлен Дополнительный Проверочный Модуль (ДПМ) для осуществления проверки заданий, которые обучаемый выполняет с использованием ДИМ (например, составляет программу на языке Паскаль пользуясь средой программирования Free Pascal). При выполнении тех заданий экзамена, которые предусматривают использование ДИМ, обучаемый

- запускает ДИМ и работает с ним до получения требуемого результата;
- сохраняет этот результат в файле средствами ДИМ;
- средствами КТС ЕГЭ включает этот файл (например, программу на языке Pascal) в свою экзаменационную работу, как развернутый ответ.

При таком методе подготовки развернутого ответа, система КТС ЕГЭ, независимо от количества и номенклатуры разрешенных ДИМ, во время проведения экзамена должна обеспечить одну единственную интерфейсную функцию, а именно:

- включение выбранного экзаменуемым файла в экзаменационную работу в качестве развернутого ответа.

На этапе проверки, КТС ЕГЭ проверяет задания частей А и В самостоятельно, но не может выполнить проверку заданий части С, подготовленных экзаменуемым с помощью выбранного им ДИМа. Поэтому КТС ЕГЭ должна подать файл с развернутым ответом на вход внешней проверяющей системе и получить на выходе оценку в баллах и текстовый комментарий для информирования экзаменуемого и экзаменационной комиссии. КТС ЕГЭ может использовать результаты проверки независимо от того, в какой форме она проводилась (экспертная или автоматизированная). Если предполагается автоматизированная проверка результатов, полученных с использованием некоторого ДИМа, то в проверяющую систему должен быть включен ДПМ, дополнительный проверяющий модуль, который возвращает КТС ЕГЭ не только оценку (первичный балл), но и некоторый текстовый комментарий. В то время как оценка в баллах используется в КТС ЕГЭ для формирования результатов экзаменационной работы, текстовый комментарий системой КТС ЕГЭ никак не анализируется и

лишь сохраняется в результатах экзаменационной работы. Примером подобного комментария может быть результат проверки синтаксически неправильной программы экзаменуемого. Этот результат состоит из значимого для системы КТС ЕГЭ нулевого первичного балла, а также включает комментарий, в котором сообщается, что в программе обнаружена синтаксическая ошибка и указывается ее характер. (Этот комментарий необходим для проведения аппеляций, общественных проверок и т.п.)

Тем самым, на этапе проверки, система КТС ЕГЭ, независимо от количества и номенклатуры используемых ДИМов, и характера проверки (экспертная, автоматическая) должна поддерживать еще две элементарные интерфейсные функции:

- передача проверяющей системе содержимого файла в качестве развернутого ответа;
- включение в результаты проверки экзаменационной работы по каждому заданию не только первичного балла, но и текстового комментария;

Для тестирования перечисленных выше возможностей КТС ЕГЭ разработчиками были выбраны три Дополнительных Интерфейсных модуля: свободно распространяемые среды программирования КуМир и Free Pascal и свободно распространяемая офисная система Open Office.

Поскольку Дополнительные Интерфейсные Модули предположительно будут использованы не только во время проведения экзамена на компьютерах ППЭ, но и для подготовки к экзамену на компьютерах образовательных учреждений и домашних (персональных) компьютерах школьников, должна быть обеспечена возможность установки указанных трех ДИМ на компьютерах, снабженных возможно большим количеством различных операционных систем. В качестве минимального набора таких систем выбраны: операционные системы компании Microsoft типа Windows XP, Windows Vista, Windows 7 и операционные системы семейства Linux: Alt Linux из Пакета Свободно Распространяемого ПО (ПСПО), Alt Linux 5.0 Школьный

А. Г. Кушниренко, А. Г. Леонов, М. А. Ройтберг
Москква, НИИСИ РАН

Проект: КуМир и ПиктоМир `http://www.niisi.ru/kumir/`

Принципы выбора языков и сред программирования при проведении ЕГЭ по информатике и ИКТ в компьютеризированной форме

Аннотация

В 2010 году был проведен пробный единый экзамен по информатике и ИКТ в компьютеризованной форме. В связи с этим встал вопрос о выборе программного обеспечения для возможного в недалеком будущем проведения ЕГЭ по информатике и ИКТ в компьютеризованной форме. В докладе описаны принципы выбора такого обеспечения и, в частности, обосновывается необходимость использования свободного программного обеспечения.

Введение в единый государственный экзамен (ЕГЭ) по информатике и ИКТ вопросов по программированию, более того, вопросов, ответом на которые являются программы, породило ряд технических, социальных и организационных проблем. Программы являются очень сложными объектами, как по форме, так и по содержанию. Формы записи программ, даже из числа используемых в школьной практике, чрезвычайно многообразны. Не используя компьютер, трудно проверить, правильно ли программа записана. Еще труднее понять, как работает программа и дает ли она требуемый в условии задачи результат. Теоретически доказано, что вообще не существует конечного числа правил, руководствуясь которыми можно проверить любую программу.

На первых ЕГЭ по информатике и ИКТ все перечисленные проблемы были тем или иным путем, с той или иной степенью успеха решены. Программы разрешалось записывать на любом известном экзаменуемому языке программирования. Задачи, ответом в которых являлась программа, проверялись экспертами, которые подходили к делу «по-человечески»: за ошибки в форме записи программы оценка не обнулялась, а лишь снижалась, неправильная работа программы, если она обнаруживалась экспертами-проверяющими, сверялась со списком типичных возможных ошибок, прилагаемым к условию

задачи в целях объективизации процедур проверки. Как ни удивительно, эта система работала.

Однако жизнь поставила задачу компьютеризации ЕГЭ по информатике и ИКТ и старые методы обеспечения процедур решения задач ЕГЭ по программирования и проверки этих решений стали неприменимыми. Компьютеризованный ЕГЭ целесообразно проводить только на базе языков программирования, удовлетворяющих ряду условий:

- должна существовать среда программирования на языке X с не слишком сложным интерфейсом;
- реализация этой среды программирования должна быть многоплатформенной (работающей в различных версиях операционных систем типа MS Windows и Linux;
- как программный продукт эта среда программирования должна быть свободно распространяемой;
- должна существовать организация, способная и желающая осуществлять сопровождение и поддержку среды программирования в течение нескольких лет.

Из сказанного следует, что на компьютеризированном ЕГЭ, когда бы он ни был введен, в 2012 году или позже, не будет предоставлена возможность составления программ на любом из 38 языков, использованных школьниками при решении задач по программированию в ЕГЭ 2010 года.

Осенью 2010 года Московский институт открытого образования (МИОО) успешно завершил выполнение госконтракта по отработке процедур и модели компьютеризированного ЕГЭ.

Для отработки модели компьютеризированного ЕГЭ по информатике и ИКТ, закончившейся пробным экзаменом 29 ноября 2010 года, удалось найти два языка и две системы программирования, удовлетворяющих перечисленным выше условиям. Это система программирования КуМир на школьном алгоритмическом языке и система программирования Free Pascal на языке программирования Паскаль. Обе системы взялся поддерживать НИИ системных исследований российской академии наук (НИИСИ РАН).

Тем самым, удалось опробовать многоязыковую систему проведения ЕГЭ по информатике и ИКТ в компьютеризированной форме. Сочетание двух языков и двух сред программирования, учебной и производственной, представляется удачным стартом. Можно прогно-

зировать, что как учебные, так и производственные языки программирования найдут свою нишу в системе подготовки к ЕГЭ.

Не исключено, что к моменту внедрения компьютеризированной формы экзамена по информатике и ИКТ номенклатура поддерживаемых (разрешенных к использованию) языков и сред программирования будет расширена. Однако уже сейчас можно назвать направления, в которых она расширена не будет.

- Не будут добавлены языки программирования, для которых имеется только среда программирования, установка которой требует приобретения лицензий.
- Не будут добавлены языки программирования, для которых имеется свободно распространяемая среда программирования только для ОС типа MS Windows или только для ОС типа Linux.
- Не будут добавлены языки программирования, для которой имеется свободно рспространяемая многоплатформенная среда программирования, которую не берется поддерживать ни одна авторитетная российская организация, учреждение или компания.

Переход к компьютеризированной форме проведения ЕГЭ по информатике и ИКТ, сопутствующее ему увеличение числа задач, ответом к которым является программа, требует повышения наглядности и эффективности среды программирования с целью уменьшить затраты обучаемого на технические проблемы и увеличить долю времени уходящую на преодоление алгоритмических трудностей. В этом направлении в системе КуМир предложен ряд решений, которые могли бы оказаться полезными и в средах программирования на производственных языках. Некоторые из них описаны в других докладах, посвященных системе КуМир, и представленных на конференции.

И. А. Хахаев
Санкт-Петербург, Институт Международного Бизнеса и Права СПб ГУ ИТМО
http://www.impip.com

Экспресс-анализ свободных систем автоматизации документооборота

Аннотация

Определяются функциональные требования к системе электронного документооборота (СЭД) и проводится анализ нескольких свободно распространяемых СЭД на соответствие этим требованиям. Указываются аргументы для окончательного выбора варианта реализации СЭД.

Институт Международного Бизнеса и Права Санкт-Петербургского государственного университета информационных технологий, механики и оптики (ИМБиП) в основном работает по программам дополнительного образования (второе высшее и повышение квалификации) по направлениям организации внешнеэкономических связей и таможенного дела, в также менеджмента (по договорам).

Организационной особенностью является наличие нескольких территориально удалённых площадок, а также скользящий график работы преподавателей и сотрудников.

По направлениям основных образовательных программ ИТМО обеспечивает как СЭД для сопровождения учебного процесса, так и систему дистанционного обучения (СДО).

В настоящее время в ИМБиП реализован универсальный документооборот на основе передачи файлов с использованием локальной сети и каталога общего доступа. Все пользователи сети имеют полный доступ к этому каталогу, в котором накоплено около 500 Гбайт документов и аудио- видеоматериалов, в том числе с кириллическими именами. При увеличении количества преподавателей и сотрудников такое решение оказывается организационно и технологически неудовлетворительным.

Поставлена задача модернизации СЭД в соответствии со следующими требованиями:

- распределённый доступ к хранилищу документов;
- разграничение прав доступа;
- возможность назначения событий, оповещение о событиях;

- система мгновенных сообщений между пользователями;
- протоколирование работы;
- наличие (или возможность организации) системы учёта оказываемых услуг и выписки финансовых документов (как пожелание);
- наличие техподдержки в Санкт-Петербурге или России (как пожелание).

Исходя из современных условий, реализация СЭД должна иметь web-интерфейс, включать в себя подсистему управления событиями, обеспечивать подключение произвольного узла файловой системы, сопровождаться системой мгновенных сообщений и иметь возможности доработки и адаптации.

В любом случае имеется этап адаптации (внедрения), на который уходят значительные ресурсы, поэтому нет смысла платить ещё и за собственно ПО для СЭД. Следовательно, рассматриваются только свободно распространяемые СЭД. Требование техподдержки частично можно удовлетворить при наличии сообщества пользователей и разработчиков и информационных ресурсов этого сообщества.

Рассматривались 4 варианта:

1. Drupal с модулями WebFM, WorkFlow, ChatRoom, EventBooking (и все модули, необходимые для удовлетворения зависимостей — CCK, Views, FileField и пр.);
2. Plone с дополнительными продуктами CMFNotification, Reflecto, Solgema.fullcalendar, Products.FileExchange, Products.Ploneboard, Products.IMS;
3. Alfresco
4. eGroupWare

Drupal 6.19 (стек LAMP) — использование возможно , но не совсем по назначению. Такое решение более подходит для сайта сообщества, чем для СЭД. Чат реализуется естественным образом. Есть проблемы с локализацией. Бизнес-процесс (workflow) организован примитивно. Какая-либо учётная система отсутствует. Поддержка — в виде форумов сообществ.

Plone 3.3.5 (собственный сервер, Python) — Хорошо организованы базовые бизнес-процессы. Дополнительные продукты, в принципе, обеспечивают нужную функциональность, однако именно эти продукты и не работают. Документация на основу системы имеется, но дополнительные продукты вообще не документированы. Reflecto игно-

рирует кириллические имена файлов и каталогов. Какая-либо учётная система отсутствует.

Alfresco 3.4.b (интегрированная community-сборка, MySQL/Java). Заранее заданные бизнес процессы отсутствуют, правила нужно писать самостоятельно. Неточности перевода приводят к проблемам при создании правил. Есть конфигурируемые отчёты о действиях, потенциально учётную систему можно организовать. Поддержка существует как в виде форумов сообщества, так и коммерческая («Корус Консалтинг»). Очень сложна в освоении. Неясно, как подключить существующее хранилище.

eGroupWare 1.6 (стек LAMP/LAPP) — модульная система, позволяет организовывать бизнес-процессы. Модуль pERP позволяет организовать учётную систему. Хорошо реализовано управление событиями. Потенциально (и реально) можно подключить чат. Имеющееся хранилище может быть подключено через механизм VFS. Освоение возможно по частям. Проблемы — локализация pERP и подключение чата — технические. Поддержка — сайт сообщества.

На этапе окончательного выбора остаются только Alfresco и eGroupWare. В пользу последней дополнительно работают такие возможности, как интеграция с почтовым сервером и возможность воспроизвести на начальном этапе внедрения структуру существующего хранилища файлов, что облегчает переход от универсального к операционному документообороту.

И. В. Лысенко, И. А. Хахаев
Санкт-Петербург, Русская Христианская Гуманитарная Академия (РХГА)
http://www.rchga/ru

Курс «Математика и информатика» в Русской Христианской Гуманитарной Академии на основе СПО

Аннотация

В статье рассказывается об опыте использования СПО для обучения студентов-гуманитариев Русской Христианской Гуманитарной Академии по дисциплине математика и информатика , а также приводится таблица сравнения серверов Moodle и e-Learning Server по некоторым параметрам.

Русская Христианская Гуманитарная Академия — старейший негосударственный ВУЗ на Северо-Западе России. Функционирует с 1989года, с1994 г. как самостоятельный ВУЗ.

Сейчас в академии существуют факультеты: мировых языков и культур; философии, богословия, религиоведения; психологии. Осуществляется подготовка по специальностям: философия, теология, религиоведение, история, культурология, психология, искусствоведение, филология (классические, восточные, русский, финский,итальянский , английский языки). Общее количество студентов академии на всех формах обучения около 900 человек. В основу учебного процесса положена концепция целостного гуманитарного образования.

Существующий государственный стандарт[1]требует, чтобы основная образовательная программа подготовки студентов по упомянутым специальностям включала обязательный блок дисциплин по математике и информатике.

В 2010 г. были установлены партнёрские отношения между РХГА и компанией ALTLinux, результатом чего стал лицензированный центр ALTLinux при Институте дополнительного образования РХГА. Центр занимается переподготовкой методистов и работников учреждений образования, в частности, в области законного применения программного обеспечения и использования СПО неспециалистами[2].

Тогда же был создан обучающий курс для гуманитарных специальностей на основе СПО. Курс включает 200 часов, из них 96 — аудиторных.

Для практических занятий был создан специализированный LiveCD на основе ALT Linux «Пятая платформа». Лекции в виде онлайн презентаций были размещены на `http://moodle.altlinux.ru` поскольку, во-первых, существовал план по формированию вузовского курса информатики на основе СПО, а во-вторых, авторов не устраивал имеющийся в РХГА образовательный портал на основе HyperMethod e-Learning Server.

Обучение ввиде практических занятий и лекций были проведены для 10 групп студентов 1-го и 2-го курсов.

---

[1] http://www.fepo.ru/index.php?menu=structs_demo

[2] http://www.altlinux.ru/training/courses/st-petersburg/

Таблица 1: Сравнение некоторых характеристик Moodle и e-Learning Server

	Moodle	e-Learning Server
Общие характеристики:		
Лицензия	Свободная, GPL	Коммерческая
Разработчик	Сообщество	IBS-HyperMethod
Платформа	Любая (xSQL/PHP)	Windows/MSSQL
Цена	0р.	168300р.
Доступность документации	свободно	После регистрации или покупки
Форматы импорта/-экспорта	SCORM, IMS/AICC	SCORM, IMS/AICC, ЦОР
Характеристики с позиций преподавателя:		
ПО для создания контента	нет	eAuthor (28710р.)
Он-лайн редактирование контента	есть	нет
Возможности интерактивного обучения	тесты, задания, семинары	тесты
Возможности изменения настроек курса (вид, компоновка)	есть	нет
Закачка файлов и их привязка к элементам курса	есть	есть
Контроль работы студентов	есть	есть
Интеграция с «Электронным деканатом»	Нет в базовой комплектации	Есть с базовой комплектации

Кроме того, была поставлена задача дальнейшего развития и модернизации портала академии `http://rhga.ru`, созданного на основе HyperMethod e-Learning Server.

Ниже приведена таблица сравнения некоторых характеристик Moodle и e-Learning Server.

На основании опыта осеннего семестра и результатов использования он-лайн лекций в Moodle был сделан вывод об эффективности дальнейшей работы со свободным программным обеспечением и возможности внедрения на основе СПО обучения и по другим дисциплинам.

Е. В. Андропова, Т. Н. Губина
Елец, Проект: Центр свободного программного обеспечения
Проект: http://www.fosscenter.elsu.ru

Свободное программное обеспечение как фактор процесса формирования информационно-технологических компетенций: достигнутые результаты

Аннотация

В докладе представлен опыт сотрудников Центра свободного программного обеспечения ЕГУ им.И.А.Бунина по формированию информационно-технологических компетенций в области свободного программного обеспечения у студентов, преподавателей средних и высших учебных заведений.

На протяжении нескольких последних лет обсуждаются проблемы перехода образовательных учреждений на использование свободного программного обеспечения. Но, как показывает практика, на уровне многих регионов РФ дело дальше обсуждения не продвигается. Одной из причин такого «перехода» является то, что многие учителя, преподаватели, инженеры привыкли получать команду «сверху» и только потом приступать к её выполнению. Если же такой команды не поступило, то стараются оттянуть время наступления событий как можно дальше. Так складывается ситуация с переходом на свободное программное обеспечение и в Липецкой области.

Многие учителя школ, а также преподаватели университетов, предпочитают работать с «привычным» программным обеспечением, а переход на «непривычное» программное обеспечение вызывает негативную реакцию. Как показал проведённый нами опрос среди преподавателей и учителей г.Ельца и Елецкого района Липецкой области, 48% респондентов считают, что свободное программное обеспечение менее функционально, не отвечает в должной степени требованиям к уровню подготовки обучающихся, не удобно в использовании. Однако на вопрос: «Какие свободные программные продукты Вы использовали?», — 31% из респондентов ответить затруднились [1].

С целью изменить сложившуюся ситуацию, в 2008 году при Елецком государственном университете им. И.А. Бунина был создан Центр свободного программного обеспечения. За время его существования уже проведена большая работа и получены некоторые результаты.

В первую очередь работа Центра СПО была направлена на разработку курсов повышения квалификации по изучению средств информационных и коммуникационных технологий на базе свободного программного обеспечения, способных заменить проприетарное программное обеспечение, используемое в учебном процессе вуза. В настоящее время апробированы программы формирования готовности преподавателей к работе со свободным программным обеспечением, необходимым для использования в профессиональной деятельности: «Основы работы в операционной системе Linux», «Информационные и коммуникационные технологии в образовании», *«Свободное программное обеспечение в системном процессе информатизации высшего профессионального образования»* (по этой программе были проведены курсы за счёт средств федерального бюджета в 2010 году, Приказ Рособразования № 428 от 11.05.2010).

При разработке программ мы опирались на анализ состояния применения средств информационных и коммуникационных технологий в процессе профессиональной подготовки и переподготовки педагогов на уровне региона. Мы пришли к выводу о том, что требуется корректировка содержания и методики информационно-технологической подготовки и переподготовки специалистов в современных экономических и социальных условиях, а именно, сдвиг акцента на свободное программное обеспечение. В основу программ были положены структурно-содержательная модель информационно-технологических компетенций и технологическая модель процесса формирования информационно-технологических компетенций, которые позволили обосновать статус информационно-технологических компетенций как ключевых в составе профессиональной компетентности, установить содержание и последовательность процесса информационно-технологической подготовки будущих педагогов [2] и переподготовки работников средних и высших учебных заведений.

Слушатель, освоивший программу, должен обладать информационно-технологическими компетенциями, включающими в себя способность:

- отслеживать и анализировать реализацию государственной политики в области образования;
- производить установку и настройку ОС Linux;
- ориентироваться и работать в различных графических интерфейсах пользователя;
- осуществлять администрирование ОС Linux;
- применять стандартные приложения ОС Linux при работе на компьютере;
- к использованию образовательных прикладных программ в профессионально-педагогической деятельности;
- применять графические редакторы в профессионально-педагогической деятельности;
- применять офисный пакет OpenOffice.org в профессионально-педагогической деятельности.

В настоящее время Центр СПО готовит учебно-методические комплексы для сопровождения программ повышения квалификации по изучению средств информационных и коммуникационных технологий на базе свободного программного обеспечения, а также учебного процесса по дисциплинам информационного профиля, в рамках которых может использоваться свободное программное обеспечение: Maxima, OpenOffice.org, GIMP, Free Pascal и др.

С помощью LMS Moodle сотрудниками Центра СПО создан электронный учебно-методический курс «Решение дифференциальных уравнений в СКМ Maxima», который проходит внедрение и апробацию в учебном процессе физико-математического факультета ЕГУ им.И.А.Бунина: `http://www.dist.elsu.ru`.

В перспективе планируется издание подготовленных учебно-методических комплексов, а также разработка новых курсов, базирующихся на свободном программном обеспечении.

Литература

[1] Андропова Е.В., Динамика развития и внедрения свободного программного обеспечения на региональном уровне // «Информационные технологии на базе свободного программного обеспечения»: материалы Всероссийской научно-практической конференции с международным участием. — Елец: ЕГУ им. И. А. Бунина, 2010. — С. 135—145. — ISBN 978—5-94809—449-6.

[2] Губина Т.Н., Андропова Е.В., Медведев В.Е., Формирование информационно-технологических компетенций будущего учителя математики и информатики. — М.: Изд-во МГОУ, 2010. — 204 с.

В. В. Кузнецов
Москва, ALT Linux

Apache Mahout. Применение модели MapReduce для задач машинного обучения

Аннотация

В докладе рассказывается про модель распределённых вычислений MapReduce, её свободную реализацию Apache Hadoop и окружающую экосистему продуктов. Подробно рассматривается библиотека алгоритмов машинного обучения и data mining Apache Mahout, приводится пример её применения для задачи классификации изменений исходного кода.

MapReduce

MapReduce — программный фреймворк, разработанный компанией Google для решения задач обработки больших объёмов данных на многомашинных кластерах. Вычислительная задача формулируется в двух следующих шагах:

- *«Map»*: распределение входных данных мастер-узлом по вычислительным узлам кластера. Вычислительные узлы обрабатывают свою порцию данных и возвращают ответ мастер-узлу.
- *«Reduce»*: вычисление мастер-узлом результата исходной задачи исходя из результатов, полученных вычислительными узлами.

Канонический пример — подсчёт количества одинаковых слов в наборе текстов.

```
public static class TokenizerMapper
      extends Mapper<Object, Text, Text, IntWritable>{
   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();
   public void map(Object key, Text value, Context context
                ) throws IOException, InterruptedException {
```

```
    StringTokenizer itr = new StringTokenizer(value.toString());
    while (itr.hasMoreTokens()) {
      word.set(itr.nextToken());
      context.write(word, one);
    }
  }
}
public static class IntSumReducer
     extends Reducer<Text,IntWritable,Text,IntWritable> {
  private IntWritable result = new IntWritable();
  public void reduce(Text key, Iterable<IntWritable> values,
                     Context context
                     ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
      sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
  }
}
```

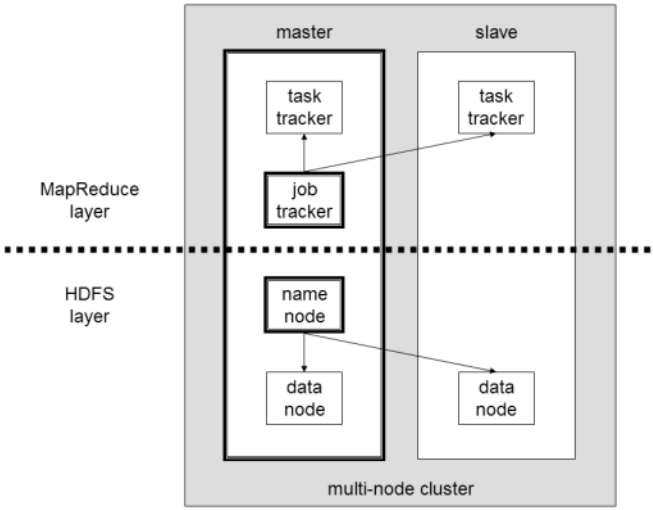
Реализация MapReduce от Google является закрытой и проприетарной. Известно, что она написана на C++ и функционирует поверх закрытой Google File System. Однако в публичном доступе имеются статьи, описывающие функционал и логику взаимодействия данных систем.

Apache Hadoop

Apache Hadoop [1] — свободный программный фреймворк, реализующий модель MapReduce и распределённую файловую систему HDFS (аналог Google File System). Изначальная разработка была произведена Doug Cutting в Yahoo!. В настоящий момент Hadoop — проект верхнего уровня в Apache Software Foundation. Язык реализации — Java.

Hadoop-кластер состоит из слоя реализации HDFS (name node и data nodes) и MapReduce (job tracker и task trackers):

В настоящий момент Hadoop используется в Yahoo!, Facebook, Last.fm, Linkedin, Twitter и других крупных проектах. Экосистема Hadoop состоит из следующих продуктов:



- Hadoop Common — HDFS и MapReduce-фреймворк;
- ZooKeeper — координатор распределённых приложений;
- Hbase — распределённая NoSQL база данных;
- Hive — инфраструктура «складирования» данных (data warehousing);
- Pig — язык написания MapReduce-заданий высокого уровня;
- Mahout — библиотека алгоритмов машинного обучения и data mining.

Apache Mahout

Apache Mahout [2] — библиотека алгоритмов машинного обучения и data mining, реализованная поверх Hadoop. Цель проекта — разработать масштабируемые реализации алгоритмов машинного обучения.

В настоящий момент в Mahout поверх MapReduce реализованы следующие алгоритмы:

- Классификация: NaiveBayes, Complementary Naive Bayes, Logistic Regression, Random Forests;

- Кластеризация: Canopy, K-means, Fuzzy K-means, MeanShift, Dirichlet Process, Latent Dirichlet Allocation, Spectral Clustering;
- Рекомендации: Distibuted/Non-distributed recommenders;
- Эволюционные алгоритмы.

В активной разработке находятся многие другие алгоритмы, в том числе: Neural Network, Perceptron and Winnow, Restricted Boltzmann Machines, Expectational Maximization, Hierarchical Clustering, Locally Weighted Linear Regression и другие.

Как пример применения Apache Mahout рассмотрим задачу автоматизации классификации изменений в репозиториях исходного кода. Для данной задачи было предложено [3] использовать алгоритм кластеризации векторов метрик изменений:

- Число добавленных, изменённых, удалённых строк кода;
- Сложность добавленного, изменённого, удалённого кода;
- Число изменённых файлов;
- Число добавленных и удалённых классов, интерфейсов и структур.

К данным метрикам применяется алгоритм k-means для кластеризации изменений исходного кода с дальнейшей экспертной классификацией получившихся кластеров. К примеру, изменения могут быть поделены на следующие классы:

- Реализация новой функциональности;
- Исправление ошибки (bugfix);
- Рефакторинг кода;
- Несущественное изменение кода (форматирование, документирование и т.п.).

Литература

[1] Apache Hadoop, http://hadoop.apache.org.

[2] Apache Mahout, http://mahout.apache.org.

[3] Князев Е.Г. Автоматизированная классификация изменений исходного кода на основе кластеризации метрик в процессе разработки ПО. СПбГУ ИТМО, 2009.

П. Г. Сутырин, Г. В. Курячий
Москва, ВМиК МГУ, Альт Линукс

О вычислительном обеспечении практикума по программированию на младших курсах

Аннотация

В докладе дается обзор методики проведения семинарских занятий по практикуму на первом курсе очного отделения специалистов факультета ВМК МГУ им. М.В. Ломоносова. Основное внимание уделяется принципам и особенностям использования вычислительной техники и программного обеспечения, в том числе свободного.

Проведение семинарских занятий по практикуму

Вначале дадим обзор программистских дисциплин, изучаемых на первом курсе отделения специалистов. В первом семестре это основы теории алгоритмов (машины Тьюринга, нормальные алгоритмы Маркова), основы алгоритмизации с использованием языка программирования Паскаль (стандарт ISO 7185), динамические структуры данных и алгоритмы их обработки.

Во втором семестре изучаются основы архитектуры ЭВМ и язык ассемблера для машины на основе ЦП Intel 8086 (MASM4). Вначале изучается концепция машин фон Неймана (кодирование команд, хранимая программа, последовательность выполнения команд) на примере модельных машин с разной системой команд (одно-, двух- и трехадресные машины, безадресные стековые машины), затем систематически изучается язык ассемблера процессора 8086, а также язык макроассемблра (все это на примере MASM 4.0).

Семинарские занятия проходят частично в аудиториях с решением задач у доски, частично в машинном зале, где каждый студент практикуется в программировании на отдельной машине.

Решение задач в машинном зале в рамках семинарских занятий предполагает определенные требования к соответствующему вычислительному обеспечению.

Требования к обеспечению учебного процесса

На время каждого занятия в машинном зале каждый студент должен работать за своей машиной, в типовом программном рабочем окружении. Его личные файлы должны быть недоступны для других студентов. Кроме этого, преподаватель группы должен иметь доступ в каталоги студентов, чтобы раскладывать им общие или индивидуальные задания, а также просматривать (и даже дополнять своими комментариями) их решения. Каждому студенту должны быть доступны инструменты разработки, отвечающие изучаемым темам.

Требования к программному обеспечению

В случае с первым курсом в настоящее время актуальным является следующий перечень требований и соответствующих программных инструментов:

1. Реализация машины Тьюринга и нормальных алгоритмов Маркова.
   - чаще всего используется студентами самостоятельно в первые недели, когда теоретически изучаются эти формализмы.
2. Реализация ISO Pascal (или его расширения, но с возможностью контроля используемого диалекта языка). Варианты:
   - Borland Pascal 7.0 (DOS-окно)
     - интегрированная среда с редактором и отладчиком;
     - библиотеки для построения текстового UI.
   - GNU Pascal
     - хорошо поддерживает ISO Pascal и другие диалекты (BP, Free Pascal, и еще несколько);
     - реализует свое расширение Паскаля, достаточно полезное для практического программирования;
     - увы, последние несколько лет не разрабатывается (есть проблемы с современными версиями GCC).
   - Free Pascal
     - имеет IDE, аналогичный BP;
     - хорошо интегрируется в современные ОС (но это не нужно для первого семестра);

  - увы, компилятор языка содержит грубые расхождения даже с базовым стантдартом;
  - не поддерживается ограничение ISO Pascal.

3. Реализация ассемблера и макроассемблера.
   - MASM 4.0 (DOS-окно)
     - использовался произвольный редактор, не специализированный для текстов программ (чаще всего из Norton Commander, т.к. дело происходило в DOS-окне).
     - отладка программ отягощалась крайне лаконичной и однообразной диагностикой системы на чаще всего допускаемые ошибки (неправильный доступ к памяти).

Опыт использования инструментария, в т. ч. основанного на СПО

В рассматриваемый период времени (с 2000 года) использовались два варианта обеспечения.

- Решение на основе Windows NT4 + Novell Netware, трудоемкое в администрировании и несовместимое с новой аппаратурой. Инструменты разработки запускались в режиме совместимости DOS-приложений для Windows.
- Загрузочный образ на основе FreeBSD, по возможности сохраняющий исходные инструменты (посредством запуска в эмуляторе DOSBox).

В процессе эксплуатации загрузочного образа FreeBSD было выявлено объективных и субъективных недостатков:

- слабая поддержка внешних носителей (не все флешки студентов определялись, и приходилось тратить семинарское время на решение проблем);
- пришлось запретить мультилогин с разных машин, т.к. студенты быстро догадались, что во время решения задач на зачете можно из соседнего машинного зала отредактировать файл в домашнем каталоге у товарища и помочь ему сдать зачет;
- Некоторые примеры из методических разработок при компиляции в Free Pascal дают не соответствующие стандарту (и «правильному ответу») результаты;

- DosBOX «из коробки» не вполне подходит для работы в Borland Pascal (например, по ‘Alt+F9‘ в BP происходит запуск программы, а в DosBOX — полный останов эмулятора; необходима русификация);
- Некоторое постоянно используемое ПО из комплекта FreeBSD, например, большинство оконных диспетчеров, по умолчанию непригодны для использования в практикуме без предварительной настройки.

Вместе с тем, сама мультизагрузочная среда (вполне стандартно реализованная на базе FreeBSD) позволила:

- синхронизовать управление пользователями с общим реестром;
- существенно упростить профилирование клиентских ОС;
- полностью исключить необходимость администрирования ПО и ОС на клиентских машинах;
- обеспечить загрузку произвольного количества подготовленных загрузочных образов;
- создать базу для перехода учебного процесса на СПО.

Загрузочный образ на основе ALT Linux 6.0

Предлагаемый загрузочный образ (для краткости — «прошивка») сделан на базе LiveCD дистрибутива семейства ALT Linux 6.0 Centaurus. Технология создания описана в статье «Сетевая установка и сетевая загрузка» [4], существенных изменений не претерпела. Особенности прошивок такого типа:

- автоматическая адаптация к широкому спектру аппаратного обеспечения;
- работа без монтирования физической файловой системы на запись (позволяет полностью исключить модификацию со стороны клиента);
- простота модификации и обновления «прошивки» (после chroot <корень_прошивки> на сервере администратор попадает в обычное окружение с работающей сетью, пакетным диспетчером и т. п.) Эти особенности предопределили особенности поддержки учебного процесса:

- Отказ от персональных аккаунтов в пользу вики-движка с хранением всей необходимой информации, включая персональные странички пользователей и их файлы
  - Унификация рабочего окружения. Зачастую начинающие пользователи настраивают свои рабочие места до полной неработоспособности. Теперь достаточно только перезагрузить компьютер.
  - Хранение постоянно используемых файлов и методических материалов в едином информационном пространстве провоцирует привлечение внимания к методически материалам
- Модификация рабочего окружения в реальном времени. При необходимости что-то изменить в окружении всех рабочих мест (например, положить на рабочий стол файл, донастроить IDE и т. п.) достаточно скопировать нужные файлы на сервер (в домашний каталог пользователя внутри «прошивки») и перезагрузить рабочие места.
- Широта и гибкость применения. Можно подготовить любое количество «прошивок» с любым содержанием, модификация прошивки (при условии заранее сделанной резервной копии) — безопасная операция, допустимая в учебном процессе, в отличие от локальной переустановки ОС и ПО В частности, последнее свойство позволяет расширить список ПО, устанавливаемого в прошивку, предлагать различные варианты, экспериментировать. Так, помимо «старых» инструментов для DOS, в новом загрузочном образе используются:
- Для обеспечения практикума по машине Тьюринга — разработка с сайта «cmcmsu.no-ip.info» [3]
- Для обеспечения практикума по нормальным алгорифмам Маркова — ещё одна разработка с сайта «cmcmsu.no-ip.info» и более сложный вариант с сайта Йорга С. Мейера[1]
- Для обеспечения практикума по Pascal — Free Pascal (планируется GNU Pascal)
- Для обеспечения практикума по Ассемблеру — NASM (планируется также JWASM, имеющий совместимый с MASM синтаксис)

[1] http://nic-nac-project.de/ jcm/index.php?nav=projects

Кроме того, предполагается вынести на суд преподавателей «промежуточный» вариант с использованием DOS-овских компиляторов, запускаемых в режиме командной строки из Linux-IDE (например, Geany).

В обеспечении учебного процесса важную роль предполагается отвести т. н. сайту практикума, на котором будут размещены учебно-методические и справочные материалы (либо ссылки на них), а также заведены персоналдьные страницы учащихся для хранения файлов. Требования к работе этого сайта (например, определение прав доступа преподавателя и студентов к файлам и страничкам, запрет доступа на запись во время зачёта и т. п.) будут определены в дальнейшем, после согласования с преподавателями.

Нелишне заметить, что загружаемый образ на базе ALT Linux третий год используется для проведения практических занятий в Вечерней Математической Школе при ВМиК МГУ (программирование на языке Python, с этого года — также и на C).

Выводы

Технология бездискового клиента, использованная в «прошивке» на базе ALT Linux, позволяет создавать и видоизменять произвольное унифицированное окружение для проведения широкого спектра практических занятий. Для эффективной организации учебного процесса необходимо соблюдение трёх требований:

- наличие актуальных Linux-версий используемого в учебном процессе ПО (добавление и/или удаление работоспособного ПО не составляет даже технической проблемы);
- достаточное информационное наполнение
  - технического обеспечения учебного процесса (сеанс работы пользователя, работа с файлами, запуск компиляторов и т. п.),
  - методического обеспечения (задачи и примеры, работающие в данном окружении);
- готовность учебного персонала к переходу, в немалой степени достигается сохранением «старых» программных средств с использованием и предварительной настройкой эмуляторов.

Литература

[1] ANSI ISO Pascal (http://standardpascal.org/)

[2] Сайт кафедры алгоритмических языков с некоторыми материалами и комплектами ПО для практикума (http://al.cs.msu.su/classes.html)

[3] Практикум 1 курса (неофициальный, но весьма полный сборник материалов) (http://cmcmsu.no-ip.info/1course/)

[4] Курячий Г. В. «Сетевая установка и сетевая загрузка» (http://uneex.ru/FrBrGeorge/BookP5/NetworkInstall)

Г. Г. Куликов, А. Г. Михеев
Уфа, Москва, Уфимский Государственный Авиационный Технический Университет, Консалтинговая группа РУНА / МЭСИ

Проект: RunaWFE `http://www.ugatu.ac.ru`, `http://wf.runa.ru/rus`

Разработка и внедрение на базе свободной системы управления бизнес-процессами и административными регламентами с открытым кодом RunaWFE лабораторного практикума по дисциплине «Автоматизированные информационные системы в производстве» в Уфимском государственном авиационном техническом университете

Аннотация

В докладе обсуждаются вопросы разработки и внедрения лабораторного практикума «Изучение методологии BPMN на примере программного продукта RunaWFE» в Уфимском Государственном Авиационном Техническом Университете. Сегодня практикум включает две апробированные в учебном процессе лабораторные работы, следующие две работы находятся в стадии разработки. Задача практикума — получить навыки практической реализации бизнес процессов в автоматизированном режиме. Изучить и закрепить практические навыки применения нотации BPMN в локальной сети INTRANET.

Процессный подход к управлению организацией

В настоящее время в организации управления предприятием наиболее перспективным является процессный подход. В этом случае деятельность предприятия представляется в виде множества выполняющихся экземпляров бизнес-процессов в определённом информационно-экономическом пространстве. Процессная организация управления организацией позволяет повысить эффективность менеджмента за счет формализации регламентов выполнения бизнес-процессов, возможности их быстрого изменения в ответ на новые условия деятельности предприятия, а также за счет повышения производительности труда работников управления.

Для реализации процессного подхода применяются схемы и правила, рекомендуемые международными стандартами (ISO 9000, ISO 15288 и др.) поддерживаемые информационными предметно-ориентированными системами классов: ERP, CAD, CAM, CAE и др. Сегодня разработан универсальный класс информационных технологий для разработки, реализации и управления бизнес-процессами и административными регламентами. Выполнение бизнес-процесса или регламента производится в компьютерной среде. Его можно представить в виде элементов работ, перемещающихся по определенному маршруту между исполнителями в соответствии с заданными правилами. Последовательность информационного сопровождения выполняемых элементов работ определяется схемой бизнес-процесса, которая разрабатывается в специальном редакторе. В узлах схемы система управления бизнес-процессами и административными регламентами распределяет задания исполнителям и контролирует их выполнение.

Задачи лабораторного практикума по изучению методологии BPMN на примере программного продукта RunaWFE

В Уфимском Государственном Авиационном Техническом Университете студентам кафедры АСУ специальностей АСОиУ и ПИЭ. преподаются дисциплины CASE технологии (SADT, UML, ARIS, Rational Rose), Компьютерные сети, ВЕБ технологии в которых приобретаются знания и навыки разработки системных моделей (включая модели БП)и их реализации на языках высокого уровня. Далее в курсе Автоматизированные информационные системы в производстве студен-

ты изучают процессный подход к управлению предприятием, строят системные модели отдельных БП и переводят их в нотацию BPMN. Для изучения основных элементов редактирования (перевода), реализации и исполнения (управления) бизнес-процессами и приобретения практических навыков исполнения бизнес-процессов используется лабораторный практикум по технологии BPMN на основе программного продукта RunaWFE».

В практикум включены разделы «Ознакомление с интерфейсом системы RunaWFE» и «Построение и исполнение Процесса выполнения и защиты типовой лабораторной работы» студентом преподавателю. В следующей работе планируется изучение методов коллективного выполнения БП.

В рамках данного практикума были определены следующие требования к знаниям студентов:

- Базовые понятия из Теории систем и Системного анализа. — Система. Системная модель системы. Процесс как система. Системная модель процесса. Системный проект.
- Из курса CASE технологии. — Языки системного моделирования IDEF, UML и др.
- Из курса Компьютерные сети. — Топология и адресация в INTRANET сетях. ВЕБ языки.
- Из курса АИС в производстве. — Логика и схемы основных БП и регламентов управления производством.

Методическими особенностями обучения в данном случае являются

- Процессный подход к организации и выполнения лабораторного практикума.
- Оформление отчета в форме системного проекта и демонстрация выполнения БП.

Свободные системы управления бизнес-процессами и административными регламентами с открытым кодом

В течение последних лет активно развиваются свободные системы управления бизнес-процессами и административными регламентами с открытым кодом, в настоящее время они начинают составлять реальную конкуренцию коммерческим проприетарным системам.

Преимущества свободных систем с открытым кодом, распространяемых бесплатно, при использовании в учебном процессе:

- Отсутствие затрат на приобретение как у студентов, так и у ВУЗа.
- Неограниченное количество инсталляций.
- Простота установки (отсутствие ключей, различных ограничений, лицензионных файлов и т.п.)
- Отсутствие каких-либо ограничений при выполнении учебных и производственных работ студентами в случае использовании свободного ПО, возможность свободной передачи результатов работы в рамках учебного процесса
- Доступность кода системы, возможность изучения и изменения кода системы независимыми разработчиками
- Возможность участия пользователей системы, в ее дальнейшем развитии.

Система RunaWFE

RunaWFE — свободная, масштабируемая, ориентированной на конечного пользователя система управления бизнес-процессами и административными регламентами с открытым кодом. Система платформонезависима (написана на Java), разрабатывается Консалтинговой группой РУНА.

RunaWFE свободно распространяется вместе со своими исходными кодами на условиях открытой лицензии LGPL. Система бесплатная. Скачать дистрибутивы и исходный код ее можно через интернет с портала разработчиков свободного программного обеспечения sourceforge.

Использование системы RunaWFE в учебном процессе обладает еще одним преимуществом: RunaWFE — это российский проект. Команда разработчиков находится в Москве, к разработчикам легко обратиться с вопросами, предложениями и пожеланиями.

При разработке и во время проведения лабораторного практикума были также отмечены некоторые недостатки системы RunaWFE:

- Необходима уникальная адаптация (если существуют необходимые условия) для взаимодействия с другими системами: электронного документооборота, электронных подписей, ERP, CAD, CAM, CAE и др.

- Не очевидна возможность обеспечения свойств иерархичности (вложенности) серверов.
- Непонятно, могут ли в этом случае быть использованы для них технологии виртуальных машин.

Литература

[1] Изучение методологии *BPMN* на примере программного продукта *RunaWFE*: Лабораторный практикум по дисциплине "Автоматизированные информационные системы в производстве"/ Уфимск. гос. авиац. техн. ун-т; Сост: Г.Г. Куликов, А.Г. Михеев, М. В. Орлов, Р.К. Габбасов, Д.В. Антонов. (в печати)

[2] OnLine demo системы RunaWFE доступно по адресу: `http://wf.runa.ru/Online_Demo`

[3] Ссылка на сайт проекта RunaWFE: `http://wf.runa.ru/rus`

Е. А. Чичкарев, А. И. Симкин
Мариуполь, Приазовский государственный технический университет

Опыт внедрения свободного ПО в учебный процесс для специальностей факультета информационных технологий

Аннотация

Рассмотрены практические особенности использования свободного программного обеспечения в учебном процессе ряда специальностей факультета информационных технологий. Приведен краткий анализ свободного ПО, применяемого в учебном процессе при подготовке IT-специалистов. Показаны возможности использования свободного ПО в процессе преподавания ряда специальных дисциплин — теории автоматического управления, моделирования, системного анализа, функционального анализа.

Совершенствование государственных стандартов подготовки IT-специалистов, внедрение новых учебных дисциплин обуславливают расширение использования свободного ПО в учебном процессе высших учебных заведений Украины. Естественно, что далеко не все учебные задачи разрешимы при помощи открытых программных

средств, но изменение отношения руководства ВУЗов к лицензированию и унификации ПО требуют развития инфраструктуры и методической базы, полностью или частично основанной на открытом ПО.

В настоящее время при подготовке специалистов по программированию, компьютерно-интегрированным технологиям, автоматизации уделяется дисциплинам, связанным с моделированием и идентификацией сложных систем, теорией управления, разработке приложений для систем реального времени (в частности, для ОС QNX), включающим лабораторный практикум с использованием ЭВМ.

Отказ от от популярных ранее RAD-систем фирмы Borland с переходом на использование средств разработки Microsoft или свободных средств разработки: Eclipse, NetBeans, Kdevelop и др. Подготовка специалистов по автоматизации и компьютерно-интегрированным технологиям, обладающих навыками программирования для POSIX-совместимой ОС QNX, обуславливают использование gcc/g++ для начального обучения программированию на 1—2 курсах.

Для организации лабораторного практикума по моделированию и теории автоматического управления во многих ВУЗах широко используют проприетарное ПО (MatLab, Simulink, MathCad). Используемые модели преимущественно включают композицию стандартных блоков, описываемых обыкновенными дифференциальными уравнениями, что и обуславливает популярность системы визуального моделирования Simulink. Совершенствание пакета SciLab (в частности, блока обработки сигналов — Signal Processing), а также средств визуального моделирования Xcos (совершенствование и расширение палитры блоков, улучшение интерфейса и т. п.) делают этот пакет достаточно привлекательной альтернативой Matlab+Simulink. Xcos предлагает стабильные и эффективные решения для промышленных и научных потребностей: функциональные возможности для моделирования электрических схем, систем управления, механических систем, гидравлических контуров, системы управления и т. д. Опыт выполнение курсовых и дипломных работ, выполнения лабораторного практикума по ТАУ, моделированию и идентификации, ряду других курсов показали, что средства Scilab+Xcos вполне достаточны для эффективной организации учебного процесса.

Широкое использование в учебном процессе на 3—5 курсах пакета SciLab и его расширений привели к отказу от MathCad на 1—2 кур-

сах и переходу к углубленному изучению m-языка, математических и графических возможностей Scilab.

Опыт подготовки специалистов по информатике и прикладной математике показал, что большинство задач, связанных с манипулированием абстрактными математическими объектами, успешно решаются студентами с использованием пакета символьных вычислений Maxima. В настоящее время отработано методическое обеспечение для выполнения домашних и учебно-исследовательских работ по математическому и функциональному анализу, элементам вариационного исчисления при помощи Maxima.

Однако широкое использование открытого ПО и в настоящее время основано преимущественно на энтузиазме преподавателей и аспирантов вследствие медленного изменения концептуального подхода к информатизации ВУЗов и сильной инерции системы образования.

О. В. Знаменская, С. В. Знаменский
Красноярск, Переславль Залесский, СФУ, УГП имени А.К. Айламазяна

Проект: mfpic3d,LaTeX-pearls
`https://sourceforge.net/projects/latex-pearls/`,
`https://sourceforge.net/projects/mfpic3d/`

Новые средства изображения кривых и поверхностей в LaTeX

Аннотация

Демонстрируются пакеты pearls и mfpic3d — новые средства иллюстрирования трёхмерных геометрических объектов на основе LaTeX, MFPic, Perl и Metapost. Пакеты свободно доступны, не имеют лицензионных ограничений, обладают отсутствующими в родственных инструментах возможностями и активно используются в научных и методических разработках СФУ и УГП имени А. К. Айламазяна.

Учёным и педагогам нередко приходится создавать учебники и монографии с большим числом формально описываемых иллюстраций, таких как изображения аналитически описываемых кривых, поверхностей и других многомерных объектов. Для этой цели создаются и используются специализированные графические языки и библиотеки к различным языками программирования[1].

[1]Работа поддержана грантом президента РФ для поддержки ведущих научных школ НШ-7347.2010.1

Пакет Pearls для включения в LaTeX исходных текстов иллюстраций

Иногда в исходный текст документа включают графику в формате EPS, или прилагают к нему графические файлы. Чтобы не запутаться, в текст обычно вставляют закомментированные исходные тексты графических иллюстраций. Работа в таком стиле требует особого внимания, поскольку при редактировании требуется согласованно обновлять картинку и комментарий с её описанием.

Пакет LaTeX-Pearls[2] содержит макропакет на TeX и скрипт на Perl, решающие эту проблему.

Не секрет, что, как самая насыщенная содержанием часть текста, математические формулы в LaTeX окружаются долларами. Поскольку исходный код описания графической иллюстрации так же являются ценнейшей частью документа, то его описание в исходном тексте окружается жемчужинами (`\pearls`).

Скрипт при запуске сверяет контрольные суммы жемчужин с имеющимися обработанными и создаёт новые картинки если после редактирования возникла новая версия исходного кода, либо ассоциирует уже готовые с соответствующим номером по порядку в особом индексном файле.

Стилевой файл обеспечивает включение результатов обработки в итоговый PDF.

Таким образом, файл с приложенными дополнениями можно редактировать и обрабатывать без скрипта если сами картинки и порядок их следования при этом остаются неизменными. После изменения картинок или порядка их следования необходим запуск скрипта.

Визуализация исходного кода таких иллюстраций часто производится медленно работающими программами, нередко обработка картинки занимает несколько минут. Использование пакета радикально сокращает затраты, исключая обработки при перестановках картинок и возвратах к старым версиям.

Другое важное преимущество использования скрипта состоит в возможности включения в документ исходных кодов картинок, выполненных в разных пакетах и на разных языках. В частности, это могут быть автоматическая раскраска исходного кода или использование конфликтующего макропакета.

[2] `https://sourceforge.net/projects/latex-pearls/`

Пакет MFPic3d для включения в LaTeX трёхмерной графики

Наличие возможностей визуализации таких объектов в современных средствах компьютерной алгебры не решает, к сожалению, проблемы создания иллюстраций полиграфического качества. Это прослеживается уже на двумерной графике: график функции одной переменной могут рисовать сотни программных приложений, но для красивого и единообразного иллюстрирования изложения математического анализа графиками функций, кривых и областей с ясным указанием выколотых точек, асимптот и других особенностей поведения пакет mfpic (являющийся надстройкой над MetaPost) предоставляет более гибкий и удобный для пользователя LaTeX пользовательский интерфейс, чем универсальные Asymptote PSTricks или PGF/TikZ.

Пакет mfpic3d[3] содержит макропакет на TeX и скрипт на Perl, расширяющие возможности пакета mfpic использованием трёхмерных координат в качестве аргументов функций mfpic и функциями рисования поверхностей.

Пакет имеет ряд уникальных особенностей, делающих его в ряде случаев незаменимым:

1. При наложении нескольких поверхностей ближние фрагменты автоматически естественно перекрывают дальние.
2. Все команды пакета mfpic могут использоваться с аргументами, заданными в трёхмерных декартовых координатах в активных скобках `\(X,Y,Z\)`.
3. Результат представляется векторной графикой, поверхности заполняются криволинейными многоугольниками.
4. Легко задаются поверхности, ограниченные неравенствами.
5. Легко задаются перфорированные поверхности.
6. Точка зрения задаётся трёхмерными координатами.

Некоторые из перечисленных особенностей наглядно иллюстрирует изображённые на рис. 1, 2 пример и его исходный код:

[3] `https://sourceforge.net/projects/mfpic3d/`

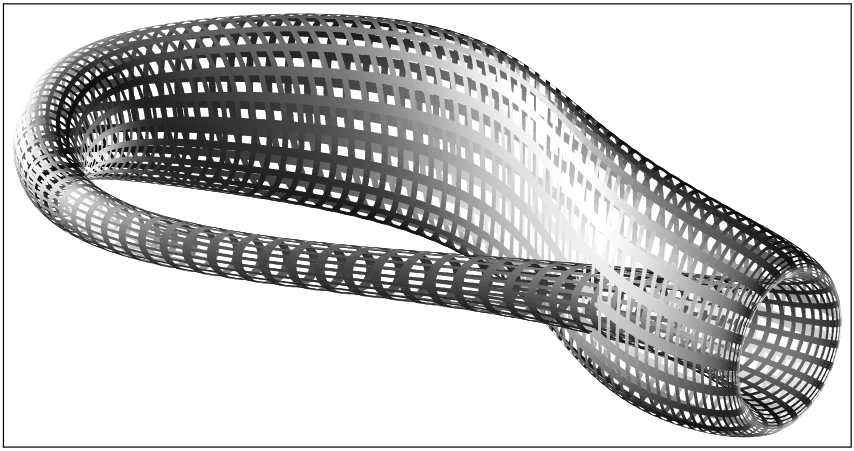


Рис. 1: Бутылка Клейна с перфорацией

```
\def\Xc{(1-cos(T))}
\def\Yc{sin(T)*(sin(\Xc))**2}
\def\R{(0.2*sin(T)+0.3)}
\begin{mfpic}[81]{-3}{3}{-2}{2.5}
\Lights{.4*white}{.3*white}{.5*white}
\ViewPoint{-8,8,4}
\Surface1{
    -2+2*\Xc-\R*sin(T/2)*sin(S),
    -\Yc+\R*cos(T/2)*sin(S),
    \R*cos(S)
    }{S=-pi..pi;T=-pi..pi}{1}
\Fill1{21,400}{.97white}A{T3}{1}
\Fill1{70,120}{.97white}A{S3}{1}
%\Fill1{15,100}{0.01magenta+.85white}A{T3}{1}
%\Fill1{50,30}{0.01magenta+.85white}A{S3}{1}
\end{mfpic}
```

Рис. 2: Исходный текст бутылки Клейна

Геннадий Кушнир, Михаил Кушнир
Москва, Гимназия №45
Проект: РУЖЭЛЬ http://www.rujel.net

Электронный классный РУЖЭЛЬ в ШС 5.0.2

Аннотация

Доклад кратко описывает основные функции и отличительные характеристики продукта «РУЖЭЛЬ». Описываются опыт внедрения РУЖЭЛЬ и преимущества включения его в ШС 5.0.2, типичные проблемы внедрения в учебный процесс.

Электронный классный журнал РУЖЭЛЬ является достойной свободной альтернативой проприетарным решениям в этой области, превосходящей их в функциональности и удобстве использования по целому ряду параметров. С другой стороны, РУЖЭЛЬ работает только с теми данными, которые хранятся в традиционном классном журнале. Решения РУЖЭЛЬ часто отличаются от популистских подходов, известных на рынке[1]. При этом учитывается законодательство о персональных данных[2].

Основным недостатком РУЖЭЛЬ до последнего времени являлась трудоемкость при его установке и первичной настройке [3]. С выходом обновленного дистрибутива Альт Линукс «Школьный сервер» 5.0.2 эти задачи заметно упростились — теперь для запуска РУЖЭЛЬ достаточно выполнить лишь несколько простых действий [4].

Кроме чисто технических проблем, внедрение электронного журнала сопряжено также с проблемами организационного, правового и методического характера. Опыт внедрения РУЖЭЛЬ позволяет систематизировать возникающие проблемы и дать свои рекомендации по их разрешению[5].

РУЖЭЛЬ предоставляет возможности анализа учебных данных посредством расширяемого модуля выборок. Результаты выборок могут быть экспортированы в формат CSV. До конца учебного года планируется развитие этого модуля для анализа агрегированных данных и конвертации результатов анализа в произвольную форму посредством XSLT.

Литература

[1] Отличительные особенности, `http://www.rujel.net/about/spec`

[2] Юридические вопросы, `http://www.rujel.net/about/Questions/femida`

[3] Руководство по установке, `http://www.rujel.net/distribute/install`

[4] Запуск в ШС 5.0.2, `http://www.altlinux.org/Rujel_123`

[5] Вопросы по внедрению, `http://www.rujel.net/about/Questions/insertion`

Евгений Алексеев, Оксана Чеснокова, Марина Сегеда
Донецк, Донецкий национальный технический университет

Об опыте использования свободных математических программ в курсе «Высшая и прикладная математика»

Аннотация

Рассмотрен опыт использования свободных математических программ Scilab, Maxima, Octave при изучении университетского курса математики студентами экономического факультета.

Курс математики в университетах традиционно читается, как сугубо теоретический курс, однако в этом учебном году на кафедре «Вычислительная математика и программирование» Донецкого национального технического университета появилась возможность прочитать курс «Высшая и прикладная математика» для студентов экономических специальностей. Курс «Высшая и прикладная математика» включает в себя лекции, практические занятия и лабораторные работы. На лекциях излагается теоретический материал по классическому курсу «Высшая математика». На практических занятиях студенты под руководством преподавателя решают задачи из классического курса высшей математики. Лабораторные работы посвящены решению математических задач с помощью современных программных средств. В качестве программ для решения задач высшей математики выбраны современные свободные математические приложения: Scilab, Maxima и Octave.

В первом модуле курса «Высшая и прикладная математика» студенты изучают линейную алгебру и аналитическую геометрию. На лабораторных работах они знакомятся с пакетами Scilab и Octave и с их помощью приступают к решению конкретных задач: находят определители, вычисляют обратные матрицы, решают СЛАУ различными методами, иллюстрируют задачи векторной алгебры и аналитической геометрии, используя графические средства пакетов.

Второй и третий модули курса посвящены математическому анализу. На первых лабораторных работах второго модуля студенты знакомятся с принципами проведения аналитических вычислений в пакете Maxima. Далее они решают задачи математического анализа (пределы, производные, исследование функций, дифференцирование, интегрирование, решение дифференциальных уравнгений и др.) «на бумаге» на практических занятиях и с помощью пакета Maxima на лабораторных работах.

В заключительной части курса планируется знакомство студентов с элементами численных методов, а также совместное использование Scilab, Octave и Maxima для решения практических экономических задач.

Параллельно с курсом «Высшая и прикладная математика» эти же студенты на кафедре «Вычислительная математика и программирование» изучают курс информатики. Это позволяет согласовать программу курсов математики и информатики для более глубокой информационной и математической подготовки будущих специалистов.

Авторы надеются, что опыт полученный в результате чтения этого курса позволит подготовить современный учебник по математике. Книга будет содержать не только теоретический материал, но и являться пособием по решению задач с помощью современных свободных математических программ.

Иван Филь
Донецк, Донецкий национальный технический университет

Применение Xcos для моделирования задач электротехники

Аннотация

Рассмотрено свободно распространяемое приложение Xcos, как среда для моделирования электрических цепей и энергосистем. Показан пример моделирования переходных процессов в электрических цепях с элементами различной структуры.

Расчет электрических цепей — это одна из основных задач энергетики. Моделирование электрических цепей в программных пакетах облегчает и ускоряет решение таких задач. В данном докладе приложение Xcos рассматривается как среда для моделирования электрических схем.

При запуске Xcos появляются два окна. Первое из них — Palette browser, браузер компонентов (рис. 1), служит для выбора элементов моделирования. Затем элементы «перетаскиваются» во второе окно (рис. 2), которое предназначено для создания и редактирования моделей.

В окне для создания моделей (рис. 2) можно также настроить время моделирования, точность и другие параметры, для чего используется команда Simulation-Setup. При двойном щелчке мыши по элементу открывается диалоговое окно для изменения параметров данного элемента.

На рис. 1 видно некоторые элементы библиотеки Electrical, в которой имеются модели измерительных приборов (датчиков) для напряжения, тока, индуктивности, конденсатора, активного сопротивления, диода, транзисторов, трансформатора, выключателя, разных источников напряжения, тока.

В Xcos можно смоделировать переходные и установившиеся режимы в электрических цепях. В качестве примера рассмотрим задачу моделирования переходного процесса в электрической цепи, представленной на рис. 2. Через некоторый промежуток времени выключатель отключает цепь от источника напряжения и происходит переходный процесс. На рис. 3 изображены графики изменения тока на индуктивности (график 2) и напряжения на емкости (график 1).

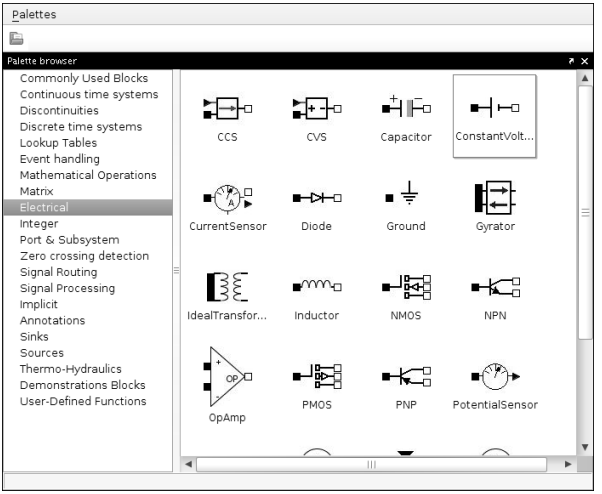


Рис. 1: Браузер компонентов

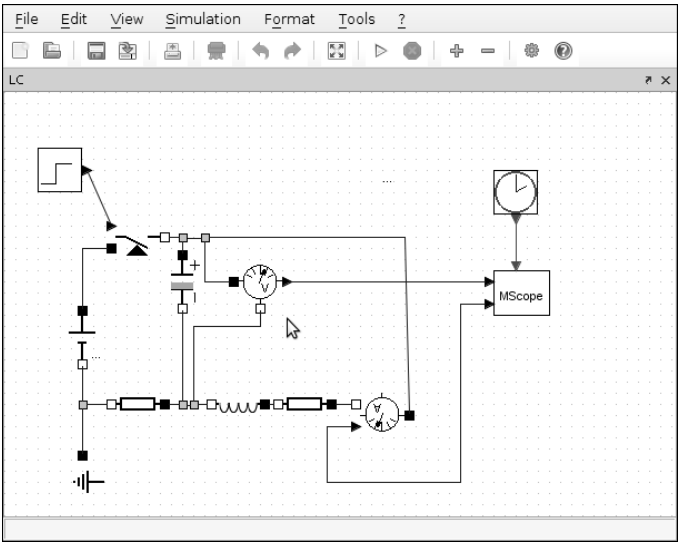


Рис. 2: Окно для редактирования моделей

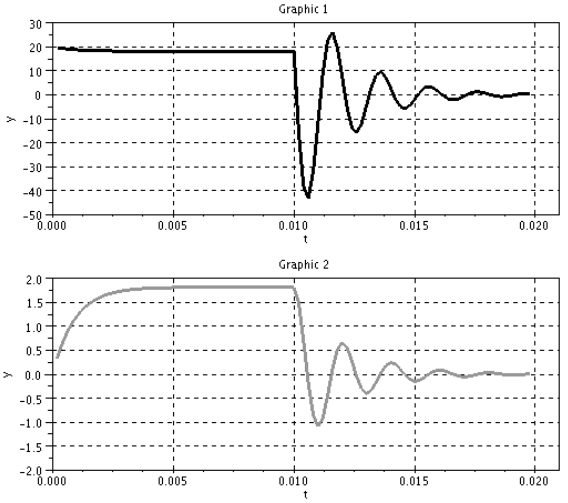


Рис. 3: Графики изменения тока на индуктивности, напряжения на емкости

На рис. 4–7 представлены схемы и графики моделирования разряда батареи конденсатора и переходного процесса в цепи с индуктивностью.

Последнюю версию Xcos можно применять при решении задач электротехники. Внедрение приложения в учебный процесс и исследовательскую деятельность усложняется недостаточно полной англоязычной документацией и отсутствием какой-либо литературы на русском языке.

При моделировании переходных и установившихся режимов в электрических цепях любой сложности и конфигурации система моделирования Xcos может использоваться наряду с проприетарной программой Simulink.

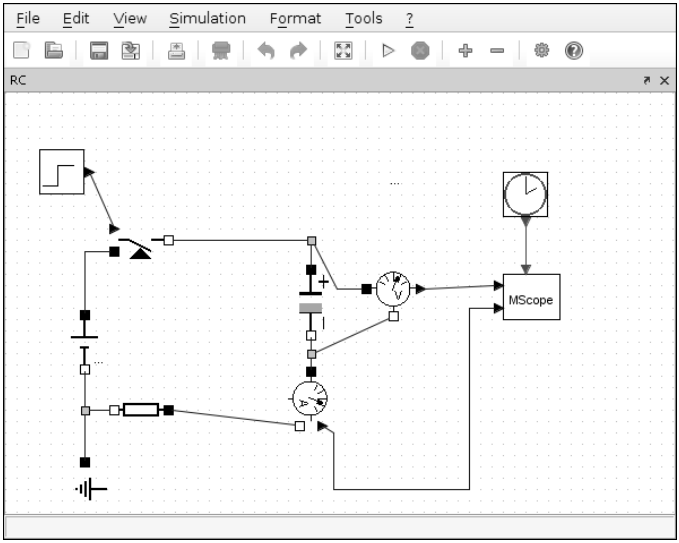


Рис. 4: RC-схема

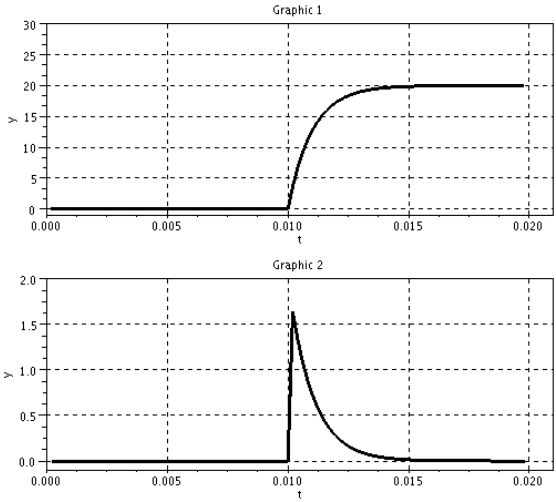


Рис. 5: Заряд батареи конденсатора. График 1 — график напряжения на конденсаторе. График — график тока на конденсаторе

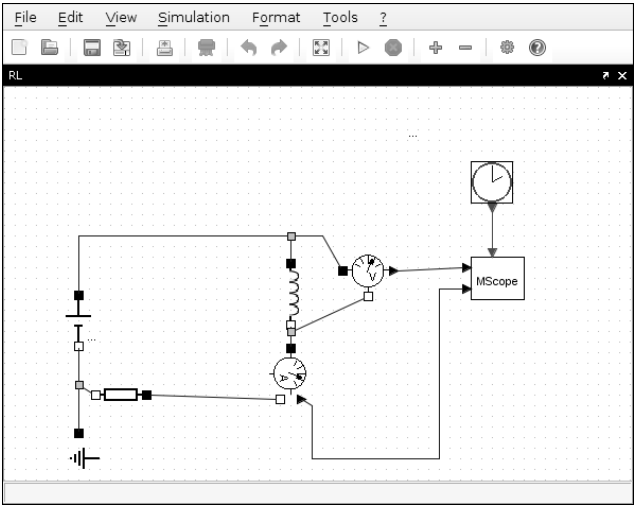


Рис. 6: RL-схема

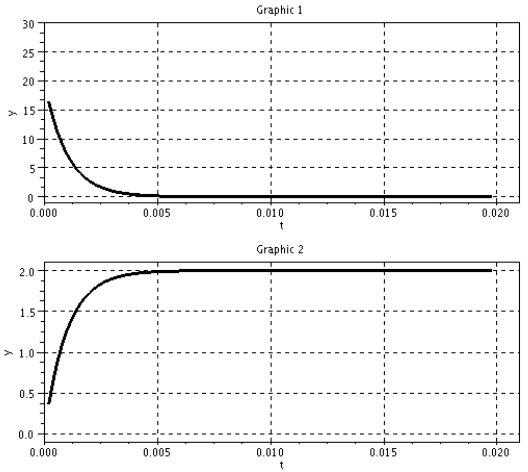


Рис. 7: Переходный процесс в RL-схеме. График 1 — график напряжения на индуктивности. График 2 — график тока на индуктивности

П. Б. Берестов
г. Асбест, Свердловская обл., компания «Профит»
http://spo-school.blogspot.com

Опыт внедрения ПСПО ALTLinux в образовательные учреждения

Аннотация

Мотивация перехода на ПСПО. Распространённые заблуждения. Влияние административного ресурса в ходе миграции. Особенности внедрения СПО в образовательные учреждения. Этапы внедрения и временные затраты. Трудности при переходе и способы их решения.

Мотивация перехода на ПСПО

- Иметь лицензионно чистую компьютерную сеть
- Максимально адаптировать программные продукты к учебному процессу
- Сократить расходы на покупку лицензий
- Иметь стабильную и надёжную компьютерную сеть

Демотиваторы перехода

- Низкая подготовка пользователей даже для проприетарных систем
- Противодействие внедрению со стороны руководителей и ключевых сотрудников
- Недоверие к штату своих инженеров

Распространённые заблуждения

- Установку и настройку НЕ способны провести обслуживающие инженеры
- На уроке информатики изучают конкретную ОПЕРАЦИОННУЮ СИСТЕМУ
- Работать в ОС Linux слишком сложно
- Отсутствие совместимости форматов файлов

Влияние административного ресурса в ходе миграции

Административный ресурс является основной составляющей успеха при переходе на СПО.

Именно поэтому необходимо всегда выполнять следующие пункты:

- Инициатива всех изменений обязана исходить от директора учебного учреждения
- Важно получить поддержку от Управления образованием
- Необходимо получить полную информацию об учебном процессе и четко поставить задачу
- При составлении технического задания, привычки пользователей техническими специалистами не учитываются
- Технические специалисты обязаны реализовать задание в максимально разумный период времени и максимально комфортно для пользователей

Необходимо понимать, что сдача позиций руководителем является мощнейшим демотиватором команде внедрения. Поэтому: Директор — основной участник перехода на СПО.

Особенности внедрения СПО в образовательные учреждения

Следует делить переход на два этапа:

- Перевод учебного процесса
- Перевод бухгалтерии

В докладе рассматривается в первую очередь перевод учебного процесса. Полностью исключить использование ОС Windows нельзя. Внедрение «строится» на использовании гетерогенной среды.

Этапы внедрения и временные затраты

- Изучение и сбор информации.
- Составление технического задания
- Развертывание тестовых систем и анализ работоспособности
- Обучение

- Ввод в тестовую эксплуатацию
- Сбор отзывов
- Полный переход на новую систему

Трудности при переходе и способы их решения

- Привычки пользователей

  Наш опыт работы в этой области показывает, что основная часть сложностей все-таки связана с привычкой работать с определенными программными продуктами. Полных аналогов не существует. Необходимо использовать ПО максимально похожее на привычное и провести обучение.

- Технические проблемы

  Есть и некоторые объективные технические проблемы — отсутствие драйверов для некоторых периферийных устройств. В данный момент развития Linux таких устройств очень мало. При покупке нового оборудования необходимо консультироваться со специалистами.

- Игнорирование информации о СПО в Управлениях образованием

  Вследствие чего тормозится или даже срывается процесс внедрения. Ведется переписка в проприетарных форматах, что вынуждает использовать проприетарное ПО. Для решения этого вопроса необходимо информировать и проводить обучение управляющих структур.

- Большинство бухгалтерских программ работают только под Windows.

  На сегодняшний день большинство образовательных учреждений использует набор бухгалтерских программ, большая часть из которых коммерческая и предоставлена им различными инстанциями (клиент-банки, системы электронного документа оборота и др). Необходимо использовать Windows если это действительно необходимо.

Д. А. Костюк, А. М. Приступчик
Брест, Брестский государственный технический университет

Особенности прозрачной виртуализации небезопасных и уязвимых систем для упрощённого администрирования учебных классов

Аннотация

Рассмотрен подход к организации прозрачной виртуализации, применённый для изолированного запуска ОС семейства Windows на типовых компьютерах учебных классов. Приведены особенности совместного применения ряда свободных программных продуктов, позволяющие избавить пользователя от взаимодействия с интерфейсом виртуальной машины, а также делегировать функции авторизации и работы с сетью базовой ОС GNU/Linux при сохранении иллюзии сетевой работы гостевой ОС. Обеспечивается экономия средств на антивирусных лицензиях, а также более высокие стабильность и скорость работы гостевой ОС по сравнению с ее невиртуализованной установкой.

Большую часть времени средний компьютер использует менее 5–7% вычислительной мощности. Системы виртуализации незначительно увеличивают этот показатель, давая пользователю взамен удобную в управлении среду — абстрагированную от разнообразия оборудования, менее подверженную угрозам, мгновенно восстанавливающуюся после сбоев. Виртуализация особенно полезна при исследовании вредоносного кода, а также при работе с ОС, ненадёжность и/или слабая защищённость которых позволяет приравнять их к данной категории.

Виртуализация рабочих станций (РС) встречается в двух вариантах. Виртуальная машина (ВМ) может быть установлена на сервер (с доступом пользователей через энергоэффективные терминалы, обладающие редуцированными ресурсами), либо на полноценную РС, пользователь которой выбирает, работать ли с хост-системой или с гостевой ОС. Прозрачная виртуализация предполагает только опосредованное взаимодействие пользователя с ВМ, скрывая разницу между запуском ОС на хост-системе и в гостевом окружении.

Применению тонких клиентов в учебных классах часто препятствует ряд причин: наличие ранее купленных офисных РС, слабый ценовой разрыв между офисным компьютером и тонким клиентом, возможная невыгодность использования устаревшего оборудования в

качестве тонких клиентов с т.з. распределения бюджета на обновление компьютерного парка, требования к сетевому оборудованию.

Типичным остаётся построение компьютерного класса на базе стандартных офисных PC. Несмотря на отсутствие приемлемых готовых решений, прозрачная виртуализация может быть организована на таких системах с применением ряда свободных продуктов.

Нами решалась задача прозрачной виртуализации учебного класса для помещения ОС от Microsoft (программирование для которых входит в учебные программы профильных специальностей) в изолированные окружения — «песочницы» (sandbox). Полученное решение позволяет:

- автоматизировать восстановление ОС;
- избавиться от антивирусных мониторов на PC (соответственно от обновления их баз и оплаты лицензий);
- осложнить непредусмотренную учебным процессом активность, связанную с динамичной 3D-графикой и сетевым взаимодействием между PC.

Особенности типового оборудования учебных классов (дешёвые процессоры без аппаратной поддержки виртуализации) оставляют VirtualBox в качестве практически единственного приемлемого решения, с переносом его конфигурации в общедоступную область и ссылкой на неё в шаблоне домашних каталогов для раздельного многопользовательского доступа к ВМ.

Т.к. сеть ВМ по умолчанию работает в режиме NAT, гостевая ОС недоступна извне при сохранении доступа из неё к сетевым сервисам. Данный режим предпочтителен для учебного класса когда учебные задания не требуют отработки сетевых взаимодействий.

В применённом нами подходе пользователи авторизуются только на хост-системах, которые работают под управлением Linux. При этом используется сервер аутентификации на OpenLDAP, а хост-системы получают доступ к учётной записи и файлам пользователя с помощью модулей pam_ldap и pam_mount. PC имеют идентичный дисковый образ, не требующий модификации, в отличие от образов ОС от Microsoft, когда те работают под управлением контроллера домена.

Доступ к общему сетевому диску с методическими материалами и к персональным каталогам может строиться на протоколах CIFS или NFS. Сетевые ресурсы монтируются хост-системой при авторизации

пользователя, а доступ гостевой ОС к ним осуществляется через механизм Shared Folders, организующий «сетевой» доступ из ВМ к каталогам хост-системы. Эта дополнительная прослойка между файл-сервером и клиентом обеспечивает:

- согласованность с использованием NAT и ограничением взаимодействия между ВМ;
- идентичность образов гостевой ОС, работающей в однопользовательском режиме;
- абстрагирование от используемого файл-сервером протокола.

Т.о., несмотря на однопользовательский режим работы, гостевая ОС монтирует при запуске каталоги хост-системы, отображающие ресурсы для конкретного пользователя.

В ВМ настроен автоматический откат образа к исходному состоянию при завершении сеанса, благодаря чему работа в ней с правами администратора и без антивируса становится безопасной (при работе антивируса ClamAV в связке с SAMBA или модулем ядра на файл-сервере).

Прозрачная виртуализация гостевой ОС обеспечена включением на хост-системе в список сеансов GDM скрипта, запускающего ВМ с подключением к ней нужных точек монтирования. При необходимости проброс USB-накопителей в ВМ может выполняться автоматически через подсистему udev.

Результатом является тиражируемый без дополнительных настроек образ, рассчитанный на запуск как минимум двух ОС с изоляцией ненадёжной системы. Помимо полной устойчивости к вредоносным воздействиям при делегировании студентам прав администратора, гостевая ОС приобретает также повышенную производительность по сравнению с установкой Windows на PC — за счёт лучшей производительности дисковой подсистемы Linux, отсутствия антивирусного монитора и сохранения отзывчивости, характерной для свежеустановленных ОС от Microsoft.

А.Н.Пустыгин. Б.А.Тарелкин, Ф.Х.Фазуллин, А.А.Ковалевский, В.А.Кулигин, И.Д.Кирилов, Е.А. Огуречникова, Е.В.Арнаутов, А.А.Хлыбов, А.В.Десинов, Н.А.Ошноров
Челябинск, Челябинский Государственный Университет

Представление открытых исходных текстов программ в альтернативных формах

Аннотация

В докладе будут представлен обзор некоторых альтернативных представлений исходных текстов программ, результаты проектирования и реализации утилит, строящих графическое и XML-представления.

На сегодняшний день в современном рынке программного обеспечения все большую долю занимают продукты с открытым исходным кодом, позволяющие использовать готовые механизмы для наших собственных проектов и нужд. Однако на данный момент практически невозможно оценить качество используемого открытого программного обеспечения. При наличии сотен и тысяч файлов с исходными текстами нам предоставляется, в лучше случае, документация по проекту, однако даже этого зачастую не бывает. Вследствие этого возникает серьезная проблема извлечения информации из открытого кода. Для облегчения процесса изучения и структуризации алгоритмов, заложенных в код, его можно представить в каком-либо альтернативном виде, например, блок-схема алгоритма, блок-схема данных, граф взаимо-действий, дерево вызовов функций и т. д. Данная работа направлена на обзор возможных альтернативных представлений открытых исходных кодов.

Целью работы является обзор некоторых альтернативных способов представлений исходных текстов программ, реализованных на различных языках программирования — Java, C, C++, C#, SmallC, Basic, NASM.

Для формирования альтернативных представлений исходного текста в качестве основы было использовано дерево разбора кода [1]. Для получения дерева разбора можно с нуля реализовать синтаксический анализатор, однако наиболее целесообразно использовать

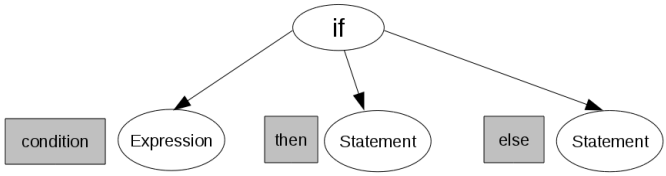
для конкретного языка готовый компилятор с открытым кодом, который уже содержит в себе все необходимые функциональные модули и структуры данных. В следующей таблице дан список компиляторов, использовавшихся для каждого из языков.

Язык	Компилятор
Java	Компилятор javac, входящий в пакет openJDK, реализованный на языке Java.
C	Компилятор gcc.
C++	
C#	DMCS
SmallC	Компилятор SmallC
Basic	Компилятор XBLite (версия XBasic)
NASM	Компилятор NASM

Грамматики этих языков сильно отличаются друг от друга, соответственно сильно различаются и компиляторы для этих языков и получающиеся деревья разбора. Вследствие этого весь проект был разбит на подпроекты, в каждом из которых был рассмотрен свой язык и свои методы формирования альтернативного представления кода.

Немаловажным моментом является способ альтернативного представления кода. В качестве возможных вариантов были рассмотрены XML и сериализация. Напомним, что сериализация — это процесс перевода структуры данных в последовательность бит. Для принятия решения в пользу того или иного представления рассмотрим достоинства и недостатки каждого из них. Безусловными плюсами сериализации являются простота реализации для большинства языков, тривиальность обратного процесса (десериализация), т.е. процесса построения дерева разбора в памяти, что в свою очередь облегчает процесс верификации полученных данных. Существенными недостатками данного подхода являются нетривиальный процесс добавления или удаления элементов в дерево и проблемы совместимости между языками. В положительным сторонам XML можно отнести очевидную наглядность, простоту редактирования дерева и возможность обработки только части дерева. Из недостатков стоит отметить привязку к внешним данным (тегам) и нетривиальный алгоритм обработки дерева. Среди этих двух возможных представлений было выбрано XML-представление как наиболее универсальное.

Для каждого из языков была написана программа, получающая с помощью компилятора дерево разбора и формирующая XML-представление этого дерева. Формат этого представления для каждого из языков специфичен, поэтому для примера рассмотрим формат представления Java-исходников. Полученное с помощью компилятора дерево может иметь узлы 47 различных типов, это можно узнать из исходников компилятора. Конечное XML-представление имеет гораздо больше узлов, потому что некоторые части этих узлов необходимо конкретизировать. Для примера рассмотрим условный оператор if. В исходном дереве для представления этого оператора необходимо 4 вершины трех различных типов.

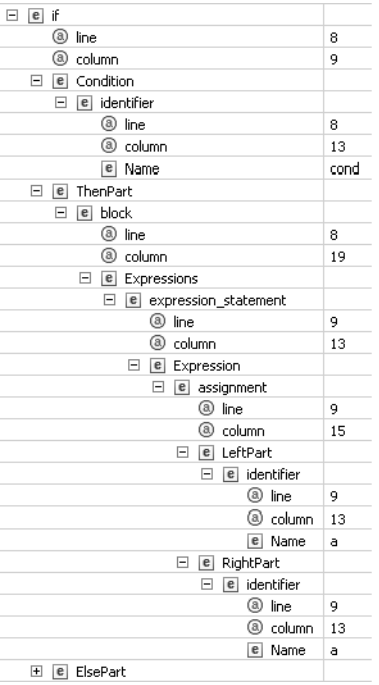


Серым обозначены конкретизирующие элементы, предназначенные для упрощения анализа дерева, представленного в виде XML. В итоге XML-файл будет содержать примерно следующий текст:

```
<if>
 <condition>
 ...
 </condition>
 <then>
 ...
 </then>
 <else>
 ...
 </else>
</if>
```

Для примера возьмем часть кода и представим ветку дерева, соответствующую этому коду. Слева — исходный код разбираемого класса, справа — XML-представление выделенного фрагмента кода (получившийся XML-файл с деревом разбора был открыт в редакторе Eclipse).

```
package com.example.main;
public class Test {
  public void foo() {
    int a,b = 2;
    boolean cond = true;
    if (cond) {
      a = b;
    } else {
      a = -b;
    }
  }
}
```



Верификация полученного представления на данном этапе работы производится вручную путем сравнения исходных текстов и полученного XML-представления. В качестве тестируемых образцов были выбраны стандартные классы Java, входящие в пакет java.lang. После построения XML-представления для всех классов, входящих в данный пакет, были выбраны несколько крупных классов для проведения детального анализа на соответствие исходным кодам.

Литература

[1] Ахо А., Ульман Д. Компиляторы: принципы, технологии и инструменты. Пер с англ. — М.: Издательский дом «Вильямс», 2003 — 768 с. : ил.

А. Н. Пустыгин, Б. А. Тарелкин, Ф. Х. Фазуллин,
А. А. Ковалевский, В. А. Кулигин, И. Д. Кирилов,
Е. А. Огуречникова, Е. В. Арнаутов, А. А. Хлыбов,
А. В. Десинов, Н. А. Ошноров
Челябинск, Челябинский Государственный Университет

Построение инструментов визуальной интерпретации алгоритмов программ с открытым исходным кодом

Аннотация

В докладе будут представлены результаты построения интерпретаций алгорит-мов программ с открытым исходным текстом в виде документов стандартных графических форматов для стандартных графических утилит с открытым кодом

Построение эквивалентных представлений исходных текстов программ может использоваться для анализа реализованных алгоритмов и извлечения знаний из программного кода. Простейшим способом представления алгоритма программы является его визуализация в виде графической форме, например в виде блок-схем алгоритмов.

Первым этапом построения визуального представления является построение XML-файла.

Для кода, написанного на языках C и C++ был использован открытый компилятор gcc. Как известно, этот компилятор предназначен не только для C и C++ языков. При исследовании принципов работы gcc было выявлено, что у него существует два уровня дерева разбора — для целевого языка и для универсального RTL языка. В процессе компиляции gcc сначала формирует дерево разбора языка, а затем переводит его в универсальный RTL. Вследствие этого возникает необходимость выбора между этими представлениями. Выбор пал в пользу представления самого языка, так как оно гораздо ближе к исходным текстам программы.

При реализации данного приложения для C# возникла другая проблема, связанная с тем, что используемый компилятор не содержит функционала, с помощью которого возможно получение дерева разбора в открытом виде. В этом случае пришлось использовать механизмы рефлексии, при помощи которых осуществлялся доступ к внутренним закрытым классам, содержащим в себе дерево разбора.

Компилятор для SmallC написан на этом же языке, являющимся подмножеством C, поэтому в этом случае от концепции неизменяемости компилятора пришлось отказаться и формирование XML-файла было включено в сам компилятор.

Генераторы представлений для языка BASIC строились на основе компилятора XBLite (версия XBasic). Эта утилита под лицензией GNU GPL.

Генераторы представлений для языке ассемблера строились на основе компилятора NASM. Шаблон XML-представления построен с помощью тегов перечисленных ниже типов:

Структура выходного файла .xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/css"
href="style.css"?>
<file>
  <fname></fname>
    <part>
    <label_1> </label_1>
      <label_2></label_2>
      <function str="№"><function>
        <command_1_№_segment str> </command_1_№_segment >
        <command_2_№_segment> </command_2_№_segment >
        <command_3_№_segment> </command_3_№_segment >
        <command_4_№_segment> </command_4_№_segment >
        <operand_№_segment ></operand_№_segment >
      </part>
</file>
```

Краткое описание узла:

1. *<file>* — корневой узел,
2. *<fname>* — отдельный узел для указания имени файла,
3. *<part></part>* — всё, что расположено между этими узлами, соответствует содержимому одной строки исходного файла,
4. *<label_ 1>* — дополнительный узел,
5. *<label_ 2>* — дополнительный узел,
6. *<function str="№">* — значение элемента соответствует имени метки,

7. *<command_1_№_segment str>* — значение элемента соответствует командам ret,jmp,call,
8. *<command_2_№_segment>* — значение элемента соответствует командам loop ... loopz,
9. *<command_3_№_segment>* — значение элемента соответствует командам ja, jnz ... jz,
10. *<command_4_№_segment>* — значение элемента соответствует всем оставшимся командам.

В качестве иллюстрации применения эквивалентного представления исходного текста на ассемблере были написаны построители блок-схем алгоритма (БСА) и слайсинг-преобразования БСА. Для построения графических образов использована утилита визуализации графов dot из пакета graphviz [1], продукт с открытым исходным кодом под лицензией CPL.

Входными данными для утилиты dot служит текстовый файл с расширением .dot, описывающий граф. Реализованный нами обработчик xml создает файлы 1.dot, 2.dot, ... n.dot; имена файлов соответствуют номерам страниц генерируемого документа графического изображения БСА. Все файлы имеет идентичную структуру:

```
digraph G {
rankdir=TB;
{Name_block[shape="Type_block",label="Text_block",URL="file"];rank=
    same;Str;}
In->Out[shape,label,color,fontcolor,dir];
. . . . .
  \{node[shape="plaintext"];
    edge[color="white"];
    1.0->1.1->......1.21->1.22;}}
```

Здесь

1. digraph (directed graphs) — метод построения (ориентированный граф.
2. rankdir — выбор направления графа, TB-сверху вниз (также можно задать направление снизу-вверх, слева на право, справа налево).
3. Name_block — имя блока. Может принимать как числовые значения (номера блоков), так и текстовые (имена, названия

блоков). В квадратных скобках указываются параметры блока, а именно: shape — тип фигуры. labe — текст внутри фигуры, color — цвет фигуры, fontcolor — цвет текста внутри фигуры.URL — переход по ссылке на другую страницу, rank — выравнивание по уровню, same — вырав-нивание по строке.

Для преобразования XML-документа в текст формата .dot использована утилита libxml2 [2] с открытым исходным кодом.

Литература

[1] Graphviz — Graph Visualization Software. http://www.graphviz.org/ Home page.

[2] The XML C parser and toolkit of Gnome libxml. http://xmlsoft.org Home page.

Д. К. Державин
Санкт-Петербург

Проект: Проект «Открытые уроки»:
http://www.avalon.ru/OpenLessons/UNIX/, http://twitter.com/derjavin

Опыт применения видеоуроков в учебно-методической работе Факультета переподготовки специалистов СПбГПУ

Аннотация

На основе опыта работы учебного центра avalon.ru в 2008—2010 годах предлагается набор методических указаний, технических решений и практических советов по применению видеоуроков в учебно-методической работе.

В области информационных технологий ситуация непрерывно меняется, и адаптировать учебные программы к этим изменениям приходится также очень быстро. В частности, важно успевать своевременно готовить учебные пособия и учебно-методические материалы для сопровождения курсов.

Как показал опыт Факультета, видеоуроки, они же «скринкасты», могут быть хорошим выходом из ситуации, так как при определённом

подходе практически не требуют дополнительного времени на подготовку и могут использоваться одновременно как в качестве раздаточного материала для учеников, так и в качестве учебно-методических пособий для преподавателей.

Наиболее полно на ФПС СПбГПУ технология была задействована в чтении учебных курсов, посвящённых архитектуре и администрированию ОС Linux. В частности, все лекции этих учебных курсов практически от начала до конца сопровождались демонстрацией примеров на большом аудиторном экране, куда транслировался экран преподавательского компьютера. Всё происходящее на экране вместе с голосом преподавателя записывалось в файл. В конце лекции записи раздавались студентам, которые с удовольствием пересматривали их сами, передавали «прогульщикам» и использовали как справочный материал во время лабораторных работ. Эти же записи успешно применялись в качестве дополнения к учебно-методическим комплексам учебного курса.

Интересно, что посещаемость лекций после внедрения видеоуроков в повседневную практику не снизилась, а заметно возросла. Также заметно упростились процедура ввода в курс нового преподавателя и обмен знаниями между преподавателями смежных курсов и специальностей.

К сожалению, проследить, как внедрение видеоуроков повлияло на успеваемость, по ряду причин оказалось невозможным. Но определённо можно сказать, что студенты стали относиться к предмету с гораздо большим интересом.

Как показал опыт Факультета, широкому распространению технологии в данный момент мешают в первую очередь следующие проблемы: излишняя требовательность преподавателей к качеству монтажа видеоуроков; боязнь заочной неконструктивной критики; отсутствие готовых технических решений; отсутствие методической базы, позволяющей адаптировать существующий учебный курс к новому формату.

В ходе внедрения технологии на ФПС СПбГПУ были выявлены следующие возможные пути решения этих проблем.

Проблема качества монтажа

Одно из распространённых возражений против записи видеоуроков на лекциях — низкое качество монтажа. Точнее говоря, его отсутствие.

Претензия в данном случае несправедливая, так как целью является не производство высококачественного учебного фильма для демонстрации на выставках, а получение актуальных учебно-методических материалов без дополнительных затрат времени на их изготовление.

Кроме того, преподаватели часто не вполне представляют себе истинную ценность видеоуроков с точки зрения учеников. Мелкие недостатки вроде оговорок по ходу объяснения ничтожны на фоне ценности наглядного руководства, полученного из первых рук. А ошибки преподавателя представляют собой дополнительную ценность за счёт демонстрации путей их решения.

Проблема боязни неконструктивной критики

Часто от введения видеоуроков в повседневную практику преподавателя удерживает боязнь оказаться в ситуации, когда невозможно что-либо возразить удалённым в пространстве и времени слушателям на неконструктивную критику.

Чувства преподавателей в данном случае, видимо, сродни чувствам музыкантов или производителей программного обеспечения. Распространённость ситуации говорит о том, что проблема вполне типична, решаема, и ничего страшного в ней нет.

Краткие методические указания

Хочется отметить, что видеоуроки применимы не только для чтения технических курсов, где большую часть объяснения можно проиллюстрировать демонстрацией работы соответствующего программного обеспечения. Любой иллюстративный материал можно дополнить как минимум примерами эффективного поиска его в Интернете.

Поэтому главное требование к адаптации подачи материала заключается в большей интерактивности: везде, где есть возможность показать действие, можно успешно использовать видеоуроки.

Кроме того полезно специальным образом оформить рабочий стол: выбрать разрешение экрана, убрать лишние, отвлекающие учеников

и отнимающие полезное пространство элементы оформления. Подобрать шрифт. Настроить удобные клавиатурные комбинации, чтобы не отвлекать учеников посторонними зрелищами типа выключения записи видео или выбора приложения из меню.

Технические решения

Предлагается готовое техническое решение на базе ffmpeg. Решение в том числе включает в себя осознанный выбор кодеков, контейнера и параметров видеопотока.

А. И. Бикова
Ижевск, ГОУВПО Удмуртский государственный университет

Психолого-педагогические принципы успешного взаимодействия педагога и ученика при обучении с использованием СПО

Аннотация

Проблемно-эвристическое обучение повышает качество освоения материала. Активное самостоятельное творческое мышление (АСТМ) является основой дальнейшего продуктивного развития и совершенствования ребёнка. Успешность взаимодействия учителя и ученика во многом зависит от принципиальной позиции учителя. При обучении с использованием СПО можно активно применять принципы, показанные в этой статье.

> Железо ржавеет, не находя себе применения, стоячая вода гниёт или на холоде замерзает, а ум человека, не находя себе применения, чахнет
>
> Леонардо да Винчи

Давно известно, что успешность ученика зависит как от собственных усилий, так и от учителя. Поэтому бесспорным элементом является признание необходимости компетентности педагога, нужности программы обучения и воспитания, выбора адекватных способов

Таблица 1: Таблица этапов педагогического процесса [2]

Этапы	1. Подготовительный	2. Основной	3. Заключительный
Функция	Организация	Осуществление	Анализ
Составляющие	Целеполагание, диагностика условий, прогнозирование достижений, проектирование и планирование развития процесса	Педагогическое взаимодействие, организация обратной связи, регулирование и корригирование (улучшение) деятельности, оперативный контроль	Выявление возникающих отклонений, вычленение ошибок, анализ причин отклонений, проектирование мер по устранению ошибок

взаимодействия участников педагогического процесса и многое другое. Особенно это должно учитываться в ситуации обучения основам программирования в школе, где будет заложен фундамент интересов школьника в данной области; но и при дальнейшем освоении специфичных тем в более страшем возрасте это важно. Кроме того, проведение занятия должно включать особенности предмета. В частности, при обучении с использованием СПО достаточно успешно будут применяться принципы проблемно-эвристического обучения для активизиции АСТМ, но и некоторые другие. Давайте рассмотрим всё по-порядку.

Педагогическое взаимодействие

Педагогическое взаимодействие есть преднамеренные контакты педагога с ребёнком, целью которого является взаимное изменение поведения, деятельности и отношения [1].

Педагогическое взаимодействие является частью всего педагогического процесса, одной из его основных составляющих.

Кратко можно описать все этапы педагогического процесса [2].

Основные методологические подходы педагогики

Поскольку наука педагогика является достаточно сложной и многогранно проявляется в практике, предлагается большое количество подходов для реализации педагогических идей.

Можно выделить основные методологические подходы педагогики.

1. *Диалектико-материалистический подход* (основа — философия И.Канта, Р.Декарта и др.);
2. *идеалистический подход* (Платон, Гегель Г.В.Ф, религиозные постулаты, экзистенциализм и др.);
3. *системный подход*, понимающий педагогический процесс как целостную систему, а задачу педагога — учёт всех компенентов системы (Кузьмина Н.В., Якунин В.А., Сластёнин В.А и др.);
4. *деятельностный подход*, понимающий деятельность как основу развития личности, а задачу воспитателя — организовать деятельность (Леонтьев А.Н., Ананьев Б.Г. и др.);
5. *личностный подход* признаёт уникальность личности, а задачу воспитателя — создать условия для развития личности (Мясищев В.Н., Амоношвили Ш.А. и др.);
6. *диалогичный (полисубъектный) подход*, полагающий, что важна не только деятельность, но и отношения, которые закрепляют личность, а задачу воспитателя — организовать среду общения ребёнка для правильного развития личности (Бубер М., Библер В.С. и др.).

   4, 5 и 6 являются основой гуманистического подхода. А так же важны:

7. *культурологический подход*, рассматривающий воспитание как целенаправленное приобщение к культурному потоку (Выготский Л.С., Бандаревская Е.В. и др.);
8. *этнопедагогический подход* подчёркивает значение традиций народа, а задачу воспитателя — набрать самое лучшее для лучшего развития личности ребёнка (Хотинец В.Ю. и др.);
9. *компетентностный подход* учитывает важность способности применять знания (Хутарской А.В., Трофимова Г.С. и др.);

10. *антропологический подход* говорит о системном использовании всех знаний о человеке в педагогическом процессе (Ушинский К.Д., Бим-Бат Б.М. и др.).

Особенности проблемно-эвристического обучения и АСТМ

Разнообразие подходов позволяет педагогу выбрать вполне определённый путь, поэтому от его позиции зависит многое, в том числе, фомирование творческой личности ученика.

В связи с этим стоит отметить важность проблемно-эвристического обучения.

Проблемное обучение есть процесс обучения, детерминированный системой проблемных ситуаций, в основе которых лежит особый вид взаимодействия учителя и ученика, характеризуется систематичностью, самостоятельностью учебно-познавательной деятельностью ученика по освоению новых знаний и способов действия решения учебных проблем.

Пробленое обучение состоит в том, что в процессе решения специально разработанной системы проблем и проблемных задач происходит овладение опытом творческой деятельности, творческое усвоение знаний и спопобов деятельности, формирование активной, творчески относящейся к своей деятельности, сознательной личности (см., например, [3, 4, 5, 6, 7, 8, 9, 10, 11, 12]).

Проблемная ситуация — осознанное субъектом затруднение, преодоление которого требует творческого поиска новых знаний, новых способов и действий.

Проблема — проблемная ситуация, принятая к решению, не огрничивающая и не указывающая направление её решения.

Проблемная задача — проблема, в условии которой даны какие-либо параметры решения. Человек решает только проблемные задачи: когда возникает проблема, человек в фонде знаний находит некоторые исхоодные параметры для её решения.

Важен ещё *проблемный вопрос*. Это входящий в состав проблемной задачи или отдельно взятый учебный вопрос, требующий ответа на него посредством мышления.

Вопрос же требующий работу памяти проблемным не является.

Проблемный вопрос может начинаться: «Почему?», «Чем можно объяснить?», «Отчего?», «Как вы это понимаете?», «Чем можно доказать?» и т.п.

Как для того, чтобы создать проблемную ситуацию, вызывающий интеллектуальную активность, самостоятельную мыслительную деятельность обучающегося, нужно учитывать ряд условий, так и для того, чтобы этим управлять.

Средством овладения опытом творческой деятельности является система творческих задач, решение которых должны искать сами учащиеся. Формирование такого опыта осуществляется постепенно на протяжении всего обучения. Постепенность овладения и определяет различие методов, организующих и обеспечивающих его усвоение.

Таких методов три: *исследовательский, эвристический* и *метод проблемного изложения.*

Кратко охарактеризуем их в таблице.

Можно сделать вывод, что сочетание и чередование всех видов методов способствует наилучшему развитию в области творческой и иных видах деятельности человека.

Также выделяют *четыре уровня проблемного обучения:*

1. несамостоятельная (обычная) активность,
2. полусамостоятельная активность,
3. самостоятельная активность,
4. творческая активность.

Как можно заметить, чем выше уровень, тем в большей степени активизируется АСТМ. Кроме того, эти уровни могут давать представление об уровне освоения и использования СПО.

Так же следует отметить достоинства и недостатки проблемного обучения.

Достоинства:

1. Учит мыслить логически, научно, творчески;
2. развивает самостоятельность;
3. делает учебный процесс более доказательным и основательным;
4. вызывает эмоциональное отношение к знаниям;
5. развивает волю, внимание, память.

Недостатки:

1. Большие затраты времени;
2. темп передачи информации зависит от индивидуальной работы ученика;
3. проблемное обучение не универсально.

Таблица 2: Таблица методов проблемного обучения [2]

Методы	Деятельность учителя	Деятельность ученика
Исследовательский	Составление и предъявление проблемных задач для поиска решения, контроль за ходом решения	Восприятие задания, составляющие части задания осмысление условий задачи, актуализация знаний о путях решения сходных задач, самостоятельное решение части задачи, самокнтроль в процессе решения и проверка его результатов, преобладание непроизвольного запоминания материала, связанного с заданием, воспроизведение хода решения и его самостоятельная мотивировка
Эвристический	Постановка проблем, составление и предъявление заданий на выполнение отдельных этапов решения познавательных и практических проблемных задач, планирование шагов решения, руководство деятельностью учащихся (корректировка и создание промежуточных проблемных ситуаций)	Восприятие проблемы или самостоятельное усмотрение проблемы, осмысление условий задачи, планирование этапов исследования, планирование способов исследования на каждом этапе, самоконтроль в процессе исследования и его завершения, преобладание непроизвольного запоминания, воспроизведение хода исследования, мотивировка его результатов
Проблемного изложения	Постановка проблемы и раскрытие доказательного пути её решения	Восприятие знаний, осознание знаний и проблемы, внимание к последовательности и контроль над степенью убедительности решения проблемы, мысленное прогнозирование очередных шагов логики решения, запоминание

Заключение

Учитывая особенности обучения с использованием СПО, педагогу необходимо использовать и свои знания, и способности к подаче материала, организации урока и прочее. А поскольку педагог обязан быть подготовлен по своему материалу на порядок выше ученика (за исключением особых и выдающихся случаев) и не имеет права просто сказать «гугл в помощь» (как временами бывает это при самостоятельном освоении СПО), то разработка собственной методики обучения с максимальным учётом условий (качество подготовки учителя и ученика, оборудование, время и т.д.) имеет значение. Конечно, определённая доля материала должна усваиваться самостоятельно учеником, однако многое вкладывает сам педагог. Таким образом, от реализации урока частью зависит и мотивация учащегося. Способность вдохновить на дальнейшие поиски есть сила педагога. Тем более в пространстве СПО.

Здесь можно вспомнить статью «Использование FreePascal при обучении школьников основам программирования» В. Пупышева [13], где он проводит анализ критериев возможности применения его методики и успешно справляется с поставленными задачами, а также рассматривает пути усовершенствования.

Проблемное обучение в грамотной его реализации в русле курсов освоения СПО (в контексте перехода школ и других учреждений на СПО это актуально) может поспособствовать наилучшему развитию как в области задаваемых тем, так и любых других видов деятельности участников педагогического процесса. Если помнить слова Н. Н. Непейводы [14], что одной из главных *компетенций* информатика высокого уровня является способность быстро входить в любую конкретную предметную область (задачи поступают из непредсказуемых областей) и осваивать её в той мере, как это необходимо для поставленной цели, то на такой почве благодатно осваивается проблемно-эвристическое обучение и активизируется АСТМ.

Литература

[1] Научно-педагогический глоссарий / Автор-составитель В.И. Тузликова. http://alimni.iubip.ru/Sokolova/lexicography/glossary.htm.

[2] Этапы педагогического процесса. http://psylist.net/hpor/ped027.htm.

[3] *Брушлинский А.В.* Психология мышления и проблемное обучение. — М.: «Знание», 1983. — 96 с.

[4] *Кудрявцев В.Т.* Проблемное обучение: истоки, сущность, перспективы. — М.: «Знание», 1991. — 80 с.

[5] *Лернер И.Я.* Дидактические основы методов обучения. М.: «Педагогика», 1981. — 185 с.

[6] *Лернер И.Я.* Проблемное обучение. — М.: «Знание», 1974. — 64 с.

[7] *Матюшкин А.М.* Актуальные вопросы проблемного обучения // Оконь В. Основы проблемного обучения. Пер. с польск. — М.: «Просвещение», 1968. — С. 186—203.

[8] *Махмутов М.И.* Организация проблемного обучения в школе. Книга для учителей. — М.: «Просвещение», 1977. — 240 с.

[9] *Оконь В.* Основы проблемного обучения. Пер. с польск. — М.: «Просвещение», 1968. — 208 с.

[10] *Поспелов Д.А., Пушкин В.Н., Садовский В.Н.* К определению предмета эвристики // Проблемы эвристики. — М., 1969.

[11] *Хуторской А.В.* Дидактическая эвристика. Теория и технология креативного обучения. — М.: Изд-во МГУ, 2003. — 416 с.

[12] Сайт научной школы А.В. Хуторского «Дидактическая эвристика». `http://khutorskoy.ru/science/concepts/theories/didactic_heuristics.htm`.

[13] *Пупышев В.* Использование FreePascal при обучении школьников основам программирования. // Тезисы докладов V Конференции разработчиков свободных программ на Протве 21—23 июля 2008 г. — Москва, Институт логики, 2008. С. 22—23.

[14] *Непейвода Н.Н.* О системе компетенция для подготовки информатиков высокого уровня // Современные информационные технологии и ИТ-образование. / Сборник докладов научно-практической конференции: учебно-методическое пособие. / Под ред. проф. В.А. Сухомлина. — М.:ИНТУИНТ.РУ, 2009. С. 43—52.

Р. М. Биков
Ижевск, УдГУ

Проект: SEvents `http://sourceforge.net/projects/sevents`

Дальнейшее развитие системы событийного программирования SEvents

Аннотация

SEvents — система событийного программирования с фактором случайности. Создана для обучения студентов основам событийного программирования. В новой версии системы переработан способ загрузки динамических библиотек, добавлена поддержка плагинов. Также написана игра на основе данной системы.

Введение

Событийное программирование — стиль программирования, при котором вычисления активизируются в результате некоторого события в системе и являются реакцией на происшедшее событие, изменяя характеристики системы [1].

По одной из классификаций [1], событийное программирование можно разделить на программирование от событий и программирование от приоритетов. *SEvents* — система событийного программирования, в некоторой степени реализующая концепцию программирования от приоритетов [2], создана для обучения студентов событийному стилю программирования.

Изменения

Для обеспечения кроссплатформенности SEvents написана на языке Java, однако эта система имеет возможность использовать динамические библиотеки в качестве объектов, способных реагировать на изменение состояния системы и участвовать в происходящих событиях. И если раньше динамические библиотеки загружались средствами самой Java, то теперь эта функция переложена на отдельную программу libseventsloader, написанную на языке C. Благодаря этому теперь вместе с системой можно использовать не только библиотеки,

написанные на языках C/C++ (что являлось ограничением JNI[1]), но действительно на любом языке программирования, поддерживающем создание динамических библиотек. Также теперь можно прерывать исполнение зависших модулей[2]. Кроме того, это введение позволило избежать зависимость системы от JDK[3] и упростить интерфейс взаимодействия между динамическими модулями и системой.

Ещё одно из нововведений — использование плагинов, которые имеют возможность создавать свои события и назначать им обработчики во время исполнения сценария.

Обработчики событий в системе выполняются псевдопараллельно, при этом могут всплывать некоторые эффекты совместного доступа к ресурсам, для обострения этих эффектов в систему введена вероятность мгновенного обновления значений переменных.

SEvents-Symplegades

SEvents-Symplegades — это антагонистическая игра[4], написанная на основе SEvents, позволяющая создавать сценарии управления игроком. Цель игры — провести свой корабль на противоположную сторону, минуя препятствия в виде следа противника и случайным образом всплывающих скал — симплегадов.

Размер игрового поля — 11 на 11 ячеек, каждая ячейка — правильный шестиугольник. По всему полю случайным образом расставлены симплегады — скалы, которые в любой момент могут всплыть и потопить корабль, находящийся в той же ячейке. За всплытие и погружение симплегадов отвечают соответствующие события. Корабли игроков совершают ход раз в 10 тактов времени, в каждый такт времени скалы могут как погрузиться, так и всплыть. У игроков также есть в запасе по 10 камней. Во время совершения хода игрок может либо бросить камень в соседнюю ячейку, либо перейти в соседнюю ячейку, либо остаться на месте. Если камень брошен в ячейку с погруженной

[1] JNI — Java Native Interface. Интерфейс, позволяющий вызывать платформозависимые функции из Java.

[2] Выполнение зависших потоков прервать практически невозможно без ущерба для основного процесса, а выполнение зависших процессов прерывается.

[3] Для того, чтобы выгрузить динамическую библиотеку из Java, нужно выгрузить classloader, загрузивший тот класс, который загрузил библиотеку. Чтобы так не делать, раньше для каждого модуля генерировался свой загружающий его класс и приходилось перекомпилировать программу.

[4] Идея создания игры принадлежит профессору Н.Н.Непейвода.

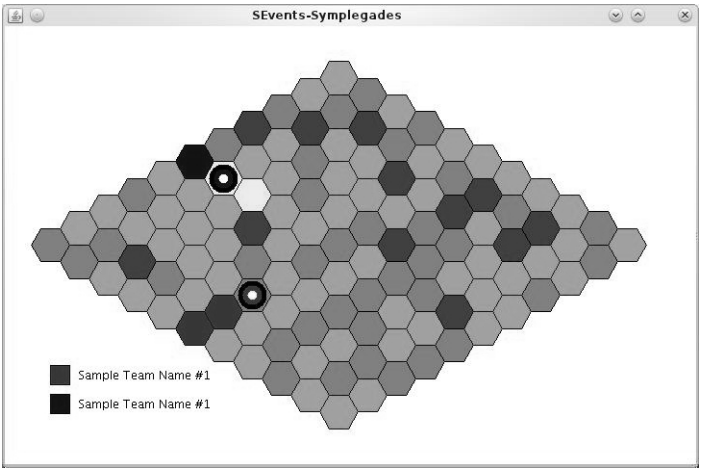


Рис. 1: Окно SEvents-Symplegades

под воду скалой, скала в этой ячейке больше не может всплыть. При переходе корабля из ячейки в соседнюю, первая помечается следом корабля и в случае, если в этой ячейке находится погруженная скала, она также больше не может всплыть, также в эту ячейку не может перейти корабль противника. Игра заканчивается в случае, если один из кораблей потоплен, либо добрался до противоположной стороны поля.

За выбор направления движения и действия игроком отвечают скрипты, написанные на языке SEvents. Для анализа ситуации на игровом поле им могут помогать динамические библиотеки путём использования общих переменных и аудита происшедших событий.

Выводы

SEvents предоставляет простую возможность создания событий и их обработчиков, что позволяет легко познакомить студентов с событийным стилем программирования и научить мыслить в рамках событий. Также эта система снимает рамки использования предопределённых событий и позволяет создавать свои события, которые возникают в определённых пользователем ситуациях.

Последние версии SEvents доступны в репозиториях ALTLinux Sisyphus и branch 5.1.

Литература

[1] *Н.Н.Непейвода.* Стили и методы программирования, Москва: Интернет-Университет Информационных Технологий, 2005.

[2] *Ринат Биков*, Система событийного программирования SEvents // Пятая международная конференция разработчиков свободных программ на Протве. Тезисы докладов. М., 2008. С. 51—53

Александр Гороховский
Донецк, Донецкий национальный технический университет
Проект: Chemical cocktail `http://peooc.donntu.edu.ua/chemistry`

Возможности Chemistry::Harmonia для определения степеней окисления химических элементов

Аннотация

Рассматривается концептуальный алгоритм и интерфейс `oxidation_state` — подпрограммы perl-модуля `Chemistry::Harmonia`, позволяющей определять степени окисления химических элементов в формулах неорганических соединений. Его применение разнообразно: консольные и Internet приложения, возможная составляющая Kalzium — пакета образовательных программ KDE Education Project.

Разработка выполнена благодаря дистрибутиву ОС ALT Linux Desktop 4.x, языку с раздвоением личности Perl и его модулям из единой Архивной Сети — CPAN.

Не секрет, что для постигающих основы химии в средней и высшей школе, одним из самых загадочных химических понятий является понятие степень окисления элементов (СОЭ) в соединениях [**?**]. Хотя это понятие и является условной численной величиной электрического заряда, приписываемого атому элемента в молекуле в предположении, что электронные пары, осуществляющие связь, полностью смещены в сторону более электроотрицательных атомов, оно служит

фундаментом и отправной точкой для определения состава молекулярных (H_2O, CO_2, NH_3, ...) и простых ионных неорганических соединений (NaCl, K_2SO_4, ...). Наконец, оно помогает *правильно уравнивать* окислительно-восстановительные реакции стехиометрическими коэффициентами по методу электронного баланса и выстраивать простую логику в объяснениях проявлений многочисленных химически периодических свойств атомов.

В настоящее время накоплено достаточно знаний о возможных степенях окисления, которые могут проявлять атомы в различных соединениях [?] и основная сложность заключается в запоминании огромного объёма фактической информации, а также исключений из общих правил[1].

На удивление, в проприетарном мире и свободно распространяемого программного обеспечения не оказалось алгоритма или, тем более, программ позволяющих по заданной в классическом виде формуле неорганического соединения определить наиболее вероятные степени окисления составляющих его атомов. В некоторой мере, исправить такое печальное положение удалось автору с помощью языка Perl и аккумулированного труда массы энтузиастов Единой Архивной сети Perl — CPAN.

Итак, концептуально алгоритм определения СОЭ основан на следующих положениях. В качестве количественного показателя СОЭ было выбрано понятие *электроотрицательность* атома[2]. В формуле простого неорганического соединения число отрицательно заряженных атомов ограничено одним элементом, а более сложного, как правило — двумя. Элементов заряженных положительно в формуле может быть сколько угодно.

Чтобы логика подпрограммы `oxidation_state` была примитивной и устойчивой, знания о электроотрицательностях и степенях окисления элементов были собраны в блоке данных.

Вначале с помощью подпрограммы `parse_formula` модуля CPAN `Chemistry::File::Formula` из формулы неорганического соединения готовится фарш (хеш) отдельных атомов и их количеств.

Затем в соответствии с адаптированной шкалой электроотрицательностей Полинга атомы ранжируются в список по убыванию этой

[1] Говорят, что дьявол проявляет себя в частностях.

[2] Электроотрицательность — фундаментальное химическое свойство атома, выражаемое количественной характеристикой его способности в молекуле притягивать к себе общие электронные пары.

величины. Первому и наиболее электроотрицательному атому в этом списке присваивается из его множества допустимых СОЭ только отрицательные значения. Второму атому списка — все возможные, т. е. отрицательные и положительные его СОЭ. Остальным атомам — только положительные СОЭ.

Полученный новый хеш из атомов и множеств их СОЭ с помощью подпрограммы `variations_with_repetition` модуля CPAN `Algorithm::Combinatorics` подвергается комбинаторному анализу. Анализ считается законченным, когда будет найдено такое сочетание степеней окисления всех атомов соединения, которое в сумме даст ноль, т. е. электронейтральную молекулу.

Конечно, в этом описании фрагмента алгоритма опущены некоторые специфичные детали, позволяющие учитывать общеизвестные химические исключения. В таком виде он практически полностью справляется со своей задачей для неорганического соединения, формула которого сконструирована не более чем из 3–4 произвольно расположенных атомов (даже с отступлением от принятой в химии записи). Например, для любой записи формулы серной кислоты: H_2SO_4, H_2O_4S, SO_4H_2, SH_2O_4 и т. д. будет получен единственно правильный результат — H^{+1}, S^{+6}, O^{-2}.

Всё было бы идеально замечательно, если бы на практике не встречались достаточно сложные химические формулы, состоящие более чем из 4-х элементов, среди которых возможно более 2-х отрицательно заряженных.

Вот, что для такого случая предусмотрено в следующей части алгоритма. Здесь на помощь приходит мощь шаблонов Perl, основанная на регулярных выражениях. Благодаря им `oxidation_state` распознаёт и выделяет в формуле неорганического соединения «знакомые» ему фрагменты — ионы и молекулы с известными заранее СОЭ. Понятно, что от числа таких знакомых фрагментов, в конечном итоге, зависит и возможность правильного определения СОЭ для очень длинно запутано, но правильно с точки зрения химии написанных формул неорганических соединений.

Интерфейс вызова подпрограммы: `HASH = oxidation_state( )`

Возвращаемый результат — `HASH` со сложной структурой:

`HASH{element}{num}[0..n-1]` — количество элемента `element` по каждому 1..n атому;

`HASH{element}{OS}[0..n-1][..]` — степень(и) окисления элемента `element` по 1..n атому.

Подкрепить описание интерфейса `oxidation_state` можно следующим классическим примером:

```
use Chemistry::Harmonia qw(oxidation_state);
use Data::Dumper;

my $chem_sub = '[Cr(CO(NH2)2)6]4[Cr(CN)6]3';
my $os = oxidation_state($chem_sub);

print Dumper($os);

%%% OUTPUT %%%

$VAR1 = {
         'H' => { 'num' => [ 96 ],
                  'OS' => [ [ 1 ] ]
                },
         'O' => { 'num' => [ 24 ],
                  'OS' => [ [ -2 ] ]
                },
         'N' => { 'num' => [ 48, 18 ],
                  'OS' => [ [ -3 ],
                            [ -3 ] ]
                },
         'C' => { 'num' => [ 24, 18 ],
                  'OS' => [ [ 4 ],
                            [ 2 ] ]
                },
         'Cr' => { 'num' => [ 4, 3 ],
                   'OS' => [ [ 3 ],
                             [ 2 ] ]
                 }
       };
```

В завершении следует отметить, что рассмотренная здесь подпрограмма `oxidation_state` является составляющей более крупного проекта *Chemical cocktail* реализуемого автором в рамках модуля `Chemistry::Harmonia`. В ближайшем будущем, после тестирования, следующими разработками будут подпрограммы:

1) `stoichiometry` — расчёт стехиометрических коэффициентов и химических уравнений для произвольно заданной смеси веществ;

2) `redox_test` — определение окислителей и восстановителей в уравнениях ОВР.